Implementing the Gaia Astrometric Solution

William O’Mullane

Universitat de Barcelona

Implementing the Gaia Astrometric Solution

prepared by:	William O'Mullane
supervised by:	Xavier Luri, Lennart Lindegren
reference:	GAIA-C3-CP-ESAC-WOM-007-02
issue:	02
revision:	0
date:	2012-05-15
status:	Issued

Abstract:

An in depth look at the Astrometric Global Iterative Solution (AGIS). This document details the algorithms in AGIS and examines them from several perspectives. It is argued that they are adequate and probably the best, if not only, approach to making an astrometric solution for Gaia data. Translation of the system of equations to an actual computer infrastructure is a non trivial task. This is the area the author has most experience in and is dealt with in some detail forming the main original content of this thesis. Some initial results using simulation data are presented to demonstrate how the system operates and removes specific effects from the data set within an acceptable time frame.

For my pal who ran around the world at my side.
My wife, my support, my love Ranpal.

Resumen de la tesis: Implementación de AGIS

Esta tesis presenta el marco numérico y computacional para la solución astrométrica Gaia. También cubre las consideraciones astrofísicas relativas a la solución y los aspectos relacionados con la gestión de la implementación de un sistema tan complejo. Cada uno de los siguientes apartados es un resumen del correspondiente capítulo de la tesis.

La misión Gaia y su procesamiento de datos

La astrometría es uno de los objetivos más antiguos de la ciencia. La medición de las posiciones y subsecuentes movimientos de los cuerpos celestes nos ha ocupado durante milenios. El primer catálogo se lo debemos a Hiparco de Nicea ($\sim$ 190 aC - 120 aC) cuyo nombre, por supuesto, se honra en la misión Hipparcos (AEE, 1997). Gaia continúa esta antigua tradición utilizando las técnicas más modernas.

En 2013, la AEE lanzará el satélite Gaia, de unas dos toneladas de peso, en un cohete Soyuz-Fregat. Consta de dos instrumentos astrométricos, además de instrumental fotométrico y de velocidad radial que le permitirán crear un mapa del espacio fásico de nuestra galaxia. Se podría trivializar Gaia afirmando que no es más que una segunda parte de Hipparcos, pero en realidad es mucho más que eso. Hipparcos observó con precisión ciento cuarenta mil fuentes, mientras que Gaia observará más de mil millones de fuentes galácticas y extragalácticas. La exactitud que se espera de Gaia tampoco tiene precedentes: en el rango de microarcosegundos, observará fuentes menos luminosas que Hipparcos hasta una magnitud de G=20 (en la que G es la banda pasante del instrumento astrométrico). La inclusión del espectrógrafo de velocidades radiales subsana una deficiencia de la misión Hipparcos y permitirá calcular las velocidades tridimensionales de los objetos. Los potenciales beneficios de Gaia para la ciencia son casi innumerables, sin embargo, la capacidad de procesamiento de datos necesaria para producir el catálogo Gaia a partir del que se extraerán estos beneficios científicos no es trivial. Este documento se centra sólo en la solución astrométrica, aunque en las áreas espectrográfica y fotométrica serán necesarias labores de procesamiento igualmente complejas.

Gaia se ubicará en una órbita de tipo Lissajous alrededor del punto de Lagrange L2 en el sistema Sol-Tierra, donde orbitará durante al menos cinco años, observando todo el cielo y llevando a cabo un censo de mil millones de fuentes, cada una de las cuales será observada unas ochenta veces en promedio. En la figura 1 la cobertura de cielo estimada para los telescopios astrométricos de Gaia, mostrando un mínimo de cincuenta tránsitos para la mayor parte del cielo. En el contexto de Gaia, tránsito se refiere a cada vez que una fuente cruza el plano focal.

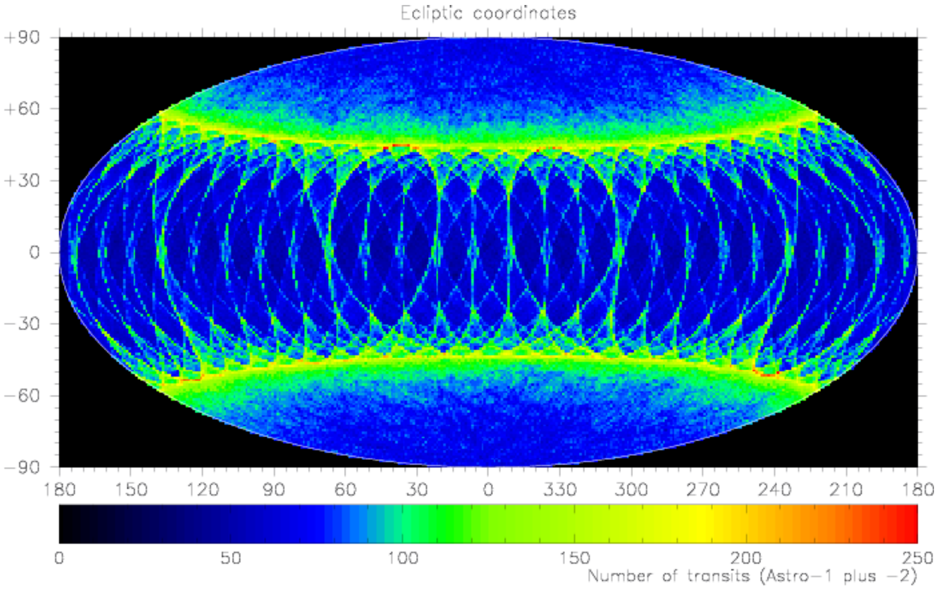


Figure 1: Durante su vida operativa, Gaia escaneará el cielo continuamente, aproximadamente en grandes círculos, siguiendo una ley de escaneo predefinida y elegida cuidadosamente. Las características de esta ley, en combinación con la dimensión AC (across-scan) de los campos astrométricos, produce el patrón que aparece en la imagen para a distribución del número predicho de tránsitos en el cielo en coordenadas eclípticas. **(Imagen J. De Bruijne)**

El autor se ha interesado desde hace mucho tiempo por la solución global de procesamiento (O'Mullane & Lindegren, 1999) y ha contribuido con una sección al libro blanco de Gaia sobre el tema (ESA-SCI(2000)4). El sistema de procesamiento de Gaia difiere del de otras misiones astronómicas por el propio diseño de la misión; es decir, porque proporciona astrometría global absoluta. Para conseguir la exactitud en microarcosegundos de arco necesaria para obtener la recompensa científica que se esboza en (ESA-SCI(2000)4) es necesario llevar a cabo un procesamiento de los datos bastante complicado, tal y como se describe en el capítulo 2.

Principios de la astrometría Gaia

El catálogo Hipparcos (ESA, 1997, Volumen 3 Capítulo 23) formula el principio general de una misión de astrometría global de manera sucinta en el siguiente problema de minimización:

$$\min_{\mathbf{a},\mathbf{n}} ||\mathbf{g^{obs}} - \mathbf{g^{calc}}(\mathbf{a},\mathbf{n})||_M \tag{1}$$

Aquí **a** es el vector de incógnitas que describe el movimiento baricéntrico de una estrella. Estos parámetros se extraen a partir de un número de observaciones de una estrella. Las observaciones de una estrella se expresan en las coordenadas del detector denominadas G y H. Por lo tanto, la observación en el tiempo t_k se representa mediante el vector de mediciones $\mathbf{g_k} = (G_k, H_k)'$ y sus estadísticas asociadas. $\mathbf{g^{obs}}$ representa el vector de todas las mediciones y $\mathbf{g}^{CALC}$ representa el vector de coordenadas de detector calculadas a partir de los parámetros astrométricos. **n** es un vector de los parámetros [1] que no resultan de interés para el problema astronómico, pero necesarios para dar forma a los datos de manera realista. La norma se calcula en una métrica M definida por las estadísticas de los datos, lo que se denomina comúnmente ponderación de errores. En el capítulo 2 se analizan los detalles matemáticos de la solución iterativa por bloques para este problema en el caso de Gaia.

Implementación de AGIS

La implementación eficaz en algún tipo de código de la solución iterativa por bloques resulta todo un reto y durante el procesamiento de datos de Hipparcos se abandonó un primer intento de alcanzar dicha solución. En (O'Mullane & Lindegren, 1999). se presentó una prueba básica del concepto, más bien una pseudoimplementación, en la que se utilizaron de nuevo los datos y el sistema de gestión de base de datos de Hipparcos. Se realizó un gran esfuerzo para adaptarlo a las dimensiones de Gaia hasta que el grupo ESAC (O'Mullane et al., 2006) lo consiguió. Este último marco de trabajo es el que se presenta en el capítulo 3.

AGIS no es sino una de las muchas partes que componen el sistema de procesamiento de datos de Gaia, sin duda una parte fundamental, pero tan solo una parte. En la Fig. 2 se presenta una descripción general aunque simplificada de AGIS. Cada uno de los componentes de la imagen puede ejecutarse en cualquier máquina independiente, salvo el `Attitude Update Server` que requiere un poco más de memoria. El `DataTrain`, como mediador, se puede observar en el centro del cuadro de la izquierda. Los algoritmos y compiladores se describen en la sección 5 del capítulo 3.

El sistema AGIS se despliega en una máquina multiprocesador local dedicada a Gaia. El marco utiliza funciones de lenguaje Java como RMI (Remote Method Invocation, Invocación Remota de Métodos). Por supuesto, el acceso a los datos es uno de los principales problemas para AGIS y en la sección 2 del capítulo 3 se analiza con bastante detalle. Se trata de un sistema de software único, diseñado y optimizado para realizar la reducción de datos astrométricos de Gaia y constituye la principal contribución original del autor.

[1] Llamados parámetros de perturbación.

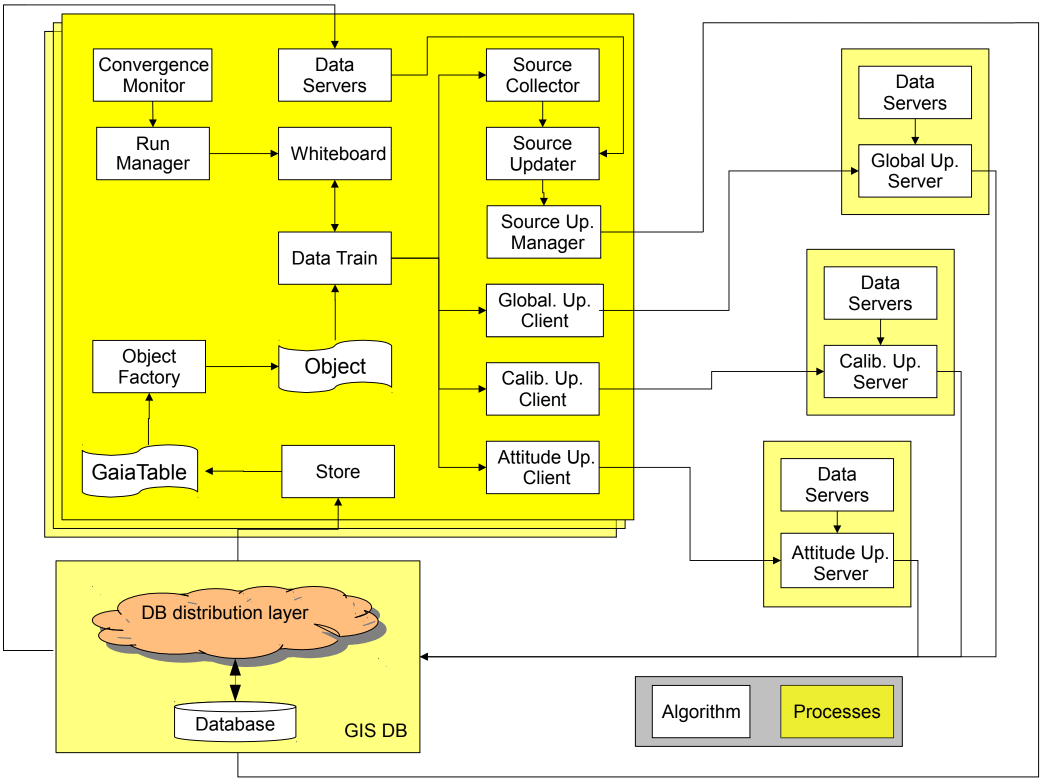


Figure 2: Este diagrama simplificado pretende mostrar una perspectiva de la lógica general de AGIS. AGIS ejecuta el mismo número de procesos en muchas máquinas, que no se muestran aquí. El cuadro grande de la izquierda representa el tren de datos, aunque puede haber un gran número de ellos ejecutándose. A la derecha se encuentran los servidores de actualización, de los que puede haber solo uno ejecutándose en todo el sistema. Todos estos procesos están basados en una base de datos.

Consideraciones astrofísicas

En el capítulo 4 se tratan algunos de los fenómenos físicos y astrofísicos sobre los que AGIS debe rendir cuenta. Lo que aquí se presenta, por supuesto, es solo un subconjunto de esos fenómenos. En efecto, se trata de un amplio rango de efectos cuya descripción en el catálogo final ocupará, sin duda, varios volúmenes. El propio AGIS solo se ocupa de una parte de todos los fenómenos conocidos que tienen efecto en el ámbito de μas. Muchos de estos temas se han tratado ya en profundidad en (Lindegren, 2005) donde se describen los principios astrométricos de Gaia. No cabe duda de que el simposio de París (Turon et al., 2005) constituyó un foro incomparable para todas ramas de la astrofísica relacionadas con Gaia.

Resultados

En el capítulo 5 se explican algunos de los resultados experimentales que nos hacen confiar en que AGIS es capaz de abordar los fenómenos astrofísicos. El software AGIS ha estado funcionando con datos simulados desde 2005. De hecho, los datos simulados proporcionan la única prueba posible de que AGIS no introduce errores sistemáticos en la solución, ya que se conocen las fuentes y observaciones auténticas y se pueden realizar comparaciones. Típicamente, la simulación añade ruido, que AGIS debería eliminar; su eficacia en esta tarea se puede juzgar con precisión comparando la astrometría resultante con los valores reales. Muchos efectos en las observaciones de Gaia no son distinguibles en los datos reales (p.ej. una aberración en un espejo o un leve desplazamiento de un CCD se calibran de la misma manera), por lo que debemos estar seguros de que AGIS no introduce errores sistemáticos propios.

Los conjuntos de datos de simulación han mejorado AGIS y han demostrado que puede recuperar o suavizar la clase de efectos esperada en el instrumento astrométrico de Gaia. Y lo que es, quizás, de más importancia, los tests han demostrado que es viable una solución iterativa global y que ésta resulta de utilidad para la astrometría de Gaia. Con el sistema AGIS, tal y como ha funcionado en 2011, ejecutando una iteración de más de 50×10^6 uentes en veinticuatro horas en un modesto sistema de multiprocesador sería posible procesar 10^8 fuentes en unos tres meses, lo que se puede deducir sin riesgo, dado que el tiempo de ejecución para AGIS es lineal respecto al número de fuentes incluidas. No cabe duda de que los algoritmos se irán haciendo más complejos, pero también los ordenadores serán cada vez más rápidos.

Enfoque de desarrollo

El desarrollo de AGIS se llevó a cabo con bastante rapidez utilizando técnicas *Agile*[2], eXtreme Programming (1999) en particular. El enfoque seguido se explica brevemente en el capítulo 6, ya que se considera que reunir y dirigir el equipo para crear el software es un logro importante. El enfoque adoptado se nutre de numerosas fuentes y de la experiencia personal obtenida a lo largo de muchos años de desarrollo de software operativo.

Durante un año, el autor realizó un estudio sobre numerosos grandes avances científicos (WOM-003), que resultó de lo más interesante y concluyó que el desarrollo del software científico es diferente al del software tradicional debido a la estructura de financiación y al enfoque general respecto al liderazgo. En cualquier caso, hay una gran diferencia respecto a un proyecto de desarrollo de un satélite. Pese a este estudio y a múltiples interacciones con ESA, esta parece decidida a imponer una estructura de gestión de proyecto mucho más tradicional para el DPAC, una postura lamentable que hará que se desperdicien esfuerzos. El enfoque de desarrollo dentro del DPAC consistente en el uso de ciclos y en tener siempre algún software en funcionamiento se acerca mucho más a las técnicas *Agile* aquí mencionadas que a la gestión de proyectos tradicional.

Los ciclos de planificación cortos del enfoque XP parecen ideales para la programación científica que deberemos asumir en Gaia. Sin embargo, está claro que para emplear dichas técnicas, el líder del equipo debe estar centrado y tener confianza en que la técnica funciona. Cada equipo tendría que establecer su propia manera de trabajar de este modo. Se haga como se haga, el principio de un sistema en evolución que pase siempre de ‘funcionando ’ a ’funcionando mejor’ es un buen enfoque para el DPAC.

Conclusión

AGIS (Astrometric Global Iterative Solution, Solución Iterativa Global Astrométrica) constituye una parte fundamental en el procesamiento de datos de Gaia. En esencia, se puede representar mediante unas pocas ecuaciones y, sin embargo, se han necesitado muchos años para llegar a ver un sistema de software operativo capaz de resolver estas ecuaciones en un tiempo razonable. La creación de un AGIS funcional ha sido uno de los problemas de la astronomía que requieren una buena comprensión de la ciencia, además de la informática necesaria para llegar a ella. Los intentos de concentrarse únicamente en los aspectos computacionales e intentar hacer que AGIS se ajustase a un marco ya existente han demostrado ser inútiles, por lo que se ha creado un marco más sencillo pero potente, especialmente diseñado para el problema de AGIS.

Los capítulos iniciales de este documento intentan mostrar hasta qué punto es necesario comprender en profundidad el problema de AGIS para poder llevar a cabo una imple-

[2] Agile es una palabra colectiva que se utiliza en la actualidad par alas técnicas de desarrollo iterativo e incremental (o desarrollo no en cascada) `http://en.wikipedia.org/wiki/Agile_development`

mentación eficiente de AGIS. El sistema, tal y como se encuentra en la actualidad, se presenta en el capítulo 3. La creación de dicho sistema no es el trabajo de una persona, sino de todo un equipo. Los aspectos de construcción de equipos necesarios para organizar un grupo de esas características no se presentan con detalle aquí, pero quizás merecerían que se les dedicase un volumen completo. No es necesario decir que la construcción y dirección de un equipo para que cree el sistema tampoco es una tarea trivial. El enfoque *Agile* utilizado para AGIS, descrito en el capítulo 6, ha demostrado ser un gran éxito y se ha ampliado al DPAC en general. El autor, por lo tanto, no ha escrito todas las líneas de código de la implementación, pero ha su influencia está presente, sin lugar a dudas, en cada una de las líneas escritas para AGIS y en muchas de las utilizadas en todo el consorcio.

Para el autor, el viaje hacia un AGIS funcional comenzó en 1998, a partir su primer contacto con la noción de una solución iterativa por bloques para datos astronómicos y con el equipo científico de Gaia, en particular con el Dr. Lindegren. Del mismo modo se familiarizó al equipo científico con partes de programas modernos, como el dominio orientado a objetos y el lenguaje Java. No fue hasta 2005 que un esfuerzo coordinado por el autor se centró en lo que entonces se denominó, y así continúa llamándose, AGIS y que se ha presentado aquí tal y como cristalizó en 2005 y se ha venido desarrollando desde entonces. El trabajo del autor ha dado lugar a muchas publicaciones internas en DPAC y a varias publicaciones en revistas internacionales con *referee*, que se incluyen en el apéndice C como referencia para el lector.

Es posible que otro sistema sustituya a AGIS en el largo plazo que nos separa de 2021 y la producción del catálogo final, pero el trabajo en el marco de AGIS ya ha dejado su influencia en muchos otros sistemas de procesamiento de Gaia. En la actualidad VARI, IDT y *First Look* utilizan una versión modificada de AGIS en la ejecución de sus tareas. Si AGIS fuese reemplazado, cosa muy poco probable, cualquier sistema que lo sucediese ya se habría beneficiado del marco de trabajo instituido por AGIS, por lo que, en cualquier caso, se deben tratar todas las observaciones de manera global para obtener la astrometría absoluta que Gaia requiere.

No nos encontramos ante el final del camino de AGIS, pero sin duda es momento de tomar un breve respiro. Se ha demostrado que AGIS es capaz de reducir observaciones similares a las de Gaia para proporcionar el marco referencial necesario para los datos de Gaia. Se ha desplegado un nuevo enfoque en la solución de un problema único y aquí se han presentado las contribuciones del autor. Ha resultado difícil y gratificante por igual. ¡Esperamos ansiosos los datos de Gaia!

Contents

Chapter 1

Introduction and Background

> Great discoveries and improvements invariably involve the cooperation of many minds. I may be given credit for having blazed the trail but when I look at the subsequent developments I feel the credit is due to others rather than to myself.
>
> **Alexander Graham Bell (1847-1922)**

1.1 Layout and Organisation

As is the way with books in general this document is presented in the form of chapters (seven in number) devoted to individual topics relating to the overall topic of Gaia astrometric data processing. We progress logically from the satellite to the equations for the astrometry to the implementation of a software system to process Gaia observations. After this we look at a few key astrophysical issues for Gaia and explain tests which have been carried out, using the implementation, concerning these effects. A few appendices provide additional information.

Here an overview paragraph is provided for each of the chapters:

This introductory chapter Section 1 provides an overview of the work as well as an overview of the satellite hardware for the reader unfamiliar with Gaia.

In Section 2 the equations underpinning the astrometric solution are explained and developed toward the algorithms actually coded in the system.

Section 3 provides details of the Java software framework which hosts the equations previously described. The framework itself has been tuned to effectively process Gaia observations and is the main original content of the thesis. This system is known as the Astrometric Global Iterative Solution or AGIS.

Having looked at the implementation, a few of the astrophysical effects and design decisions which influence the solution are described in Section 4.

Although no data has yet been received from Gaia, extensive simulations have been performed in the Gaia community. Some of the AGIS tests relating to the astrophysical phenomena described in Section 4 are reported in Section 5. In this manner a demonstration of the effectiveness of AGIS is presented.

A discussion and overview of the development approach adopted for AGIS is presented in Section 6. The eXtreme programming approach is particularly suited to science development and worked well for this project in the form presented.

Brief conclusions are drawn in Section 7.

Appendix A provides a primer on Quaternions which are used for attitude modelling.

A complete list of the mind boggling acronyms used in this document appear in Appendix B.

Finally some published papers are included in Appendix C.

1.2 The original thought and acknowledgement

It should be clear in reading this thesis the Astrometric Global Iterative Solution (AGIS) as presented is and has always been the work of Dr. L. Lindegren. The author has had the good fortune to work with Dr. Lindegren over many years and hopes to have at least sometimes aided the evolution of AGIS. A certain grasp of what AGIS needs to do is required to actually implement the system and the Gaia community in general has always been willing to explain what the intention is without which there would have been no advancement on the implementation. Here again Dr. Lindegren has been endlessly patient to what, at times, must appear to be bumbling attempts of computer scientists. Dr. X. Luri was encountered within a similar time frame during the early years of the Gaia project. Some initial collaborations on simulations in the University of Barcelona and continued Gaia interaction led to the notion of submitting a PhD thesis with Dr. Luri as supervisor. Many thanks are due to Dr. Luri for encouragement, explanation and suggestions. None of this would have come about and none of these people would have been encountered were it not for Dr. M. Perryman. He laid down the challenge to understand and make AGIS work, many thanks to him for explaining astrometry in very simple terms[1] to the author on arrival at ESTEC.

A thesis needs an original thought and this is something the reader may find difficult to pin down. The Gaia project is a highly collaborative undertaking with thoughts fixed firmly on the common long term goal of the catalogue rather than near term developments. The words of Bell (at the start of the chapter) ring true completely for the author. Hence it is not always clear if an idea has come from the author or from interaction with

[1] Reciprocation was in Java programming tutorials, the author got the better deal by far.

another. This is not something which the author has held as important. For example the notion of executing all AGIS algorithms in parallel was arrived at seemingly simultaneously - it does not advance our goal to know who came up with this idea, it simply must be done.

It is a little more clear on the implementation side of AGIS. The framework for running the equations in a distributed manner has always been the realm of the author. The use of Java for AGIS, and its use in Gaia processing in general, are solely attributable to the author for better or for worse. Even here however after 2005 with a team working on this in ESAC the system has grown and changed not always due to the influence of the author, although all involved would certainly still identify a single architect of the original system. These days the single architect is more likely to be seen as Dr. U. Lammers.

Hence it is hoped the reader will accept that a key and important role in designing and bringing AGIS to operation has been played by the author. That the original contribution of the author is definitely in the area of implementation and has greatly influenced AGIS and indeed Gaia processing in general. Finally it is hoped the reader grasps the aesthetic of AGIS and understands that it is a collaboration where all have played an important role and trying to pick the originality of one from the collective is not always easy.

In this vein the author would like to acknowledge the work of DPAC and especially the work of the ESAC team without whom AGIS would not function. In particular the work of John Hoar, Jose Hernandez, Paul Parsons and most of all Uwe Lammers was vital to the initial version of AGIS.

1.3 The Gaia Mission

Astrometry is one of the oldest pursuits in science. The measurement of positions and later motions of celestial bodies has been an occupation for millennia. The first catalogue came from Hipparchus ($\sim$190 BC - 120 BC) whose name is of course echoed in the Hipparcos mission (ESA, 1997). Gaia continues in this ancient tradition using the most modern of techniques.

ESA is due to launch the $\sim$2000kg Gaia satellite in 2013 on a Soyuz-Fregat rocket. It consists of two astrometric instruments as well as photometric and radial velocity instruments allowing it to build a phase space map of our galaxy. One may trivialise Gaia saying it is simply Hipparcos II, yet it is so much more. Hipparcos accurately observed one hundred and twenty thousand sources whereas Gaia will observe in excess of one thousand million galactic and extra-galactic sources. The accuracy predicted for Gaia is also unprecedented, in the microarcsecond range, it will observe fainter sources than Hipparcos down to G=20 magnitude (where G is the passband of the astrometric instrument). The addition of the radial velocity spectrograph addresses a shortcoming[2]of the Hipparcos mission allowing correct velocities in all three dimensions to be calculated. The potential scientific benefits of Gaia are practically innumerable. The data processing required to produce a Gaia Catalogue from which these scientific benefits may be reaped is, however, non trivial. This document looks only at the astrometric solution - equally difficult processing tasks exist within the spectrographic and photometric areas.

Gaia will be injected into a Lissajous orbit around the Sun-Earth Lagrange point L2, where it shall spin for at least five years observing the whole sky, conducting a census of one thousand million sources, observing each approximately eighty times. Fig. 1.1 plots the predicted sky coverage for Gaia's astrometric telescopes showing a minimum of fifty transits over most of the sky. A transit for Gaia means a source crossing the focal plane.

[2] Radial velocities were measured from the ground for Hipparcos sources.

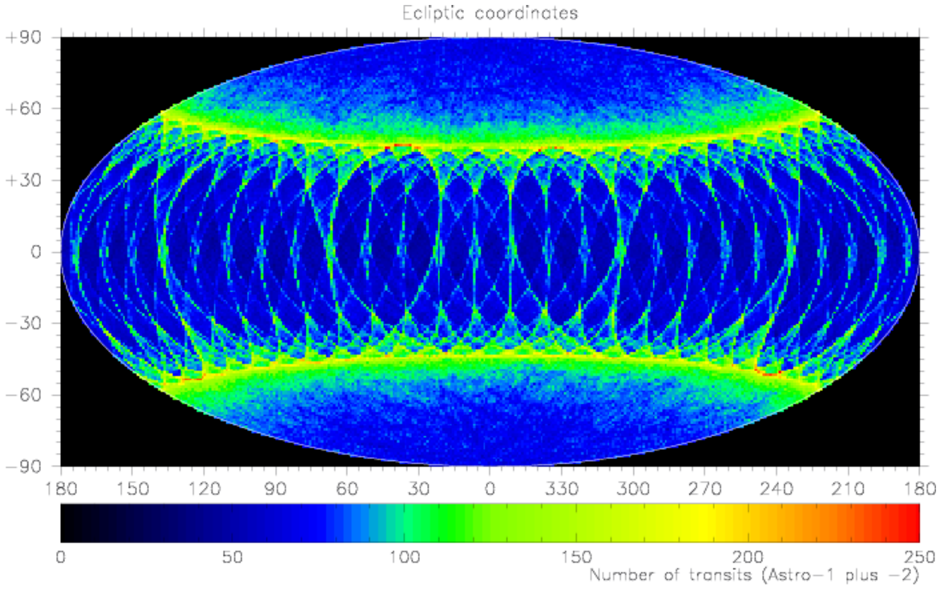


Figure 1.1: During its operational lifetime, Gaia will continuously scan the sky, roughly along great circles, according to a carefully selected pre-defined scanning law. The characteristics of this law, combined with the across-scan dimension of the astrometric fields of view, result in the above pattern for the distribution of the predicted number of transits on the sky in ecliptic coordinates **(Image J. De Bruijne)**

1.3.1 Astrometric instruments

The satellite contains two astrometric telescopes [3] with a fixed angle of 106.5 degrees between them. The viewing directions of both telescopes overlap on a common focal plane. The entrance pupil is 1.4 m x 0.5 m and the focal length is 35 m [4] for each Telescope. Fig. 1.2 shows the mirrors and the focal plane they eventually reflect onto. The focal plane [5] functionally consists of five CCD strips (see Fig. 1.3): the Basic Angle Monitor (BAM) and Wave Front Sensor (WFS), the Sky Mapper (SM), the Astrometric Field (AF) and the Blue and Red Photometers (BP, RP) and the Radial Velocity Spectrograph (RVS). The mosaic contains 106 Charge Coupled Devices (CCDs) with pixels of 10 micrometers along scan x 30 micrometers across scan size (59 mas x 107 mas). The first column contains the BAM and WFS CCDs, the next two columns of CCDs form the SM, which works out the transits of sources crossing the focal plane, thus allowing efficient read-outs of the CCDs in the main focal plane. The main astrometric measurements are made in the AF in the next nine columns of CCDs. The next two columns of CCDs are for the red and blue photometry, these are in fact spectrographs, providing spectrophotometry, and not standard photometers. Finally, a little in front of the rest of the CCDs, is a block of four by three CCDs for the radial velocity spectrograph. Fig. 1.3

[3] http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_astrometric_instrument.pdf

[4] http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_gaia_telescope.pdf

[5] http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_gaia_focal_plane.pdf

Table 1.1: Predicted end of mission astrometric accuracies for Gaia (ESA website)

	B1V	G2V	M6V
$V - I_C$ [mag]	-0.22	0.75	3.85
Bright stars	5-14 μas (6 mag< V < 12 mag)	5-14 μas (6 mag< V < 12 mag)	5-14 μas (8 mag< V <14 mag)
$V = 15$ mag	26 μas	24 μas	9 μas
$V = 20$ mag	330 μas	290 μas	100 μas

depicts the Astro focal plane.

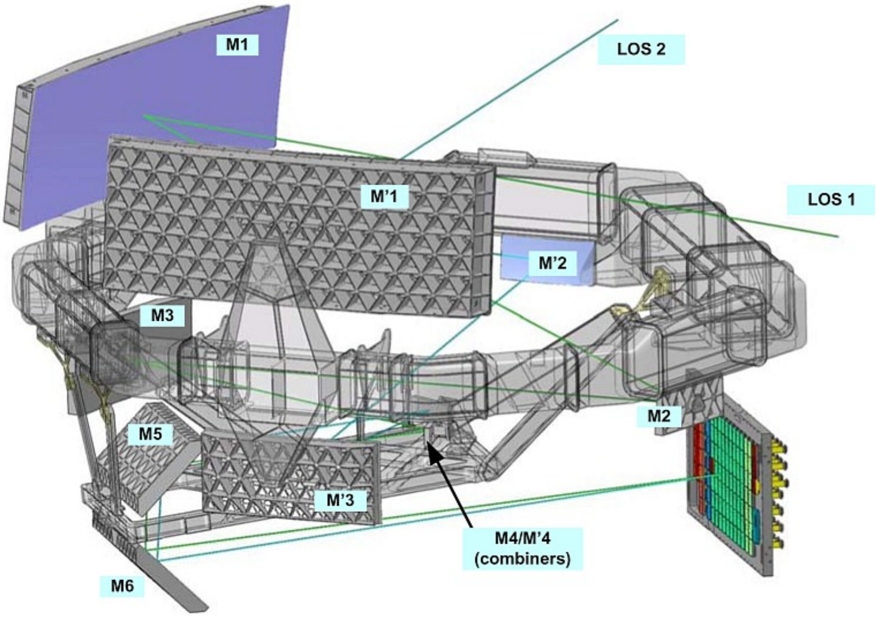


Figure 1.2: The Gaia science payload consists of two telescopes separated by a basic angle of 106.5 degrees and sharing a focal plane. A beam combiner combines the images from the two telescopes after the third of the six mirrors in the system. The focal plane (see Fig. 1.3) is split between astrometric, photometric and radial velocity spectrograph CCDs. (Image Astrium)

The predicted end of mission accuracies for Gaia astrometry are shown in Table 1.1. We see it is a few microarcseconds to G≈12 dropping to 20-25 microarcseconds to G=15 and ranging to a few hundred microarcseconds at G=20. These accuracies are only possible given the statistical effect of having ∼80 measurements per source, the focal plane itself will not make such an accurate individual measurement.

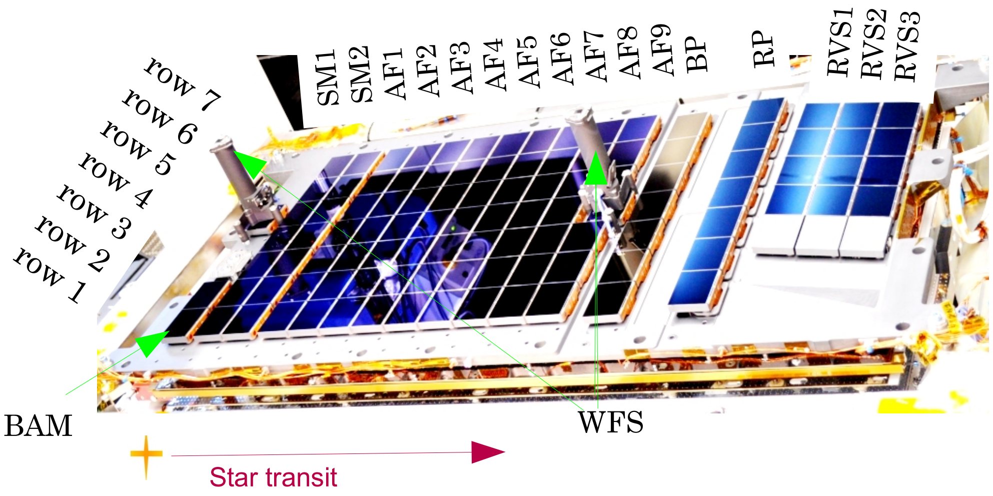


Figure 1.3: The two telescopes share the same focal plane with a mask in the beam combiner ensuring that only light from Astro 1 falls on SM1 and only light from Astro 2 on SM2. Apart from the SMs, light from both telescopes crosses the entire focal plane, the main part of which is dedicated to astrometry (AF1 - AF9 CCDs). After crossing the AF CCDs light crosses the BP (Blue Photometer) and RP (Red Photometer). Occupying a smaller area only some sources will cross the Radial Velocity Spectrograph (RVS). In the picture we also see the Wave Front Sensors (WFS) and Basic Angle Monitoring (BAM) CCDs in the focal plane. **(Photograph Astrium)**

1.3.2 Photometric measurements

As mentioned above, following the AF CCDs is a column of CCDs for the Blue Photometer and another for the Red Photometer. Light from all sources seen in the AF will be seen in the RP and BP giving ∼80 photometric observations of every Gaia source. The light falling on these CCDs is first dispersed along scan by a fused-silica prism, hence these are more like spectrographs than photometers. The BP disperser operates in the 330-660 nm range while the RP is in the 650-1000nm range producing spectrophotometry. The photometric measurements are of interest in themselves and are also essential for correcting systematic chromatic effects in the astrometric measurements.

1.3.3 Radial velocity spectrograph

Certain sources are selected in the RP for observation in the Radial Velocity Spectrograph (RVS). There are three CCD strips and four CCD rows for the RVS, hence not all sources crossing the focal plane cross the RVS, at end of mission the RVS coverage is estimated to be about 40 transits per source - considerably less than for the astrometric instrument. The RVS CCD block is slightly in front of the other CCDs in the focal plane. In its optical path is an optical module with grating and six spherical fused-silica lenses to create the spectra which fall on the red enhanced CCDs. The RVS is operating in the near-infrared (847-874 nm) [6].

RVS will provide radial velocities and about 40 individual spectra in the narrower 847-874nm band for up to 150 million stars. This alone will make Gaia one of the largest sources of spectra (Sloan will only have 1 million when it is finished).

1.3.4 Scientific Benefits

Gilmore et al. (2000) summarise the prime scientific possibilities with a thousand million source photometric and kinematic survey:

- The history of our galaxy: test hierarchic formation theories, inner bulge/bar dynamics, disk/halo interactions, dynamical evolution, what is the warp, star cluster disruption, weigh spiral structure, star formation history, chemical evolution, link to high redshifts.
- Stellar evolution: detect rapid evolutionary phases, quantify pre-main sequence evolution, complete census of local neighbourhood.
- Stellar formation: dynamics of star forming regions, luminosity function for pre-main sequence stars.

[6] `http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_RVS_instrument.pdf`

- Brown dwarfs: census of brown dwarfs in binaries.
- Planetary systems: complete census of (Jupiter size) planets around 3×10^5 stars.
- The Local Group: rotational parallaxes for Local Group galaxies, kinematic separation of stellar populations, galaxy orbits to give cosmological history.
- Beyond the Local Group: parallax calibration of distance scale, zero proper motion QSO survey, photometry of 108 galaxies.
- The nature of matter: galactic rotation curve, disk mass profile from velocity dispersions, internal dynamics of Local Group dwarfs.
- Fundamental physics: determine the space-curvature parameter γ to 10^{-6}.
- Reference frames: define the local metric.
- Serendipity: the first all-sky, multi-colour, multi-epoch phase-space map.

A comprehensive scientific case (100 pages) for Gaia is laid out in the Concept and Technology Study report (the white book ESA-SCI(2000)4). This is the document which culminated Phase A of Gaia before it was chosen as ESA's sixth cornerstone mission. A more concise science case is available in (Perryman et al., 2001). Within the science community the proceedings from the symposium *The Three-Dimensional Universe with Gaia* (Turon et al., 2005) provide in-depth considerations of the scientific benefits of Gaia.

1.4 General Gaia Data processing

The author has long been interested in the global processing solution (O'Mullane & Lindegren, 1999) and contributed a section to the white book (ESA-SCI(2000)4) on the topic. Gaia processing differs from other astronomy missions because of the mission design i.e. optimised to provide absolute global astrometry. To achieve the microarcsecond accuracies required to provide the scientific bounty outlined in (ESA-SCI(2000)4) a rather involved statistical processing must be carried out on the data.

Consider that the field of view for Gaia is about 0.7 degrees [7] consisting of pixels 59x177 milliarcseconds in size. With centroiding capability of 100th of a pixel this gives a rough accuracy of 1 milliarcsecond for the instrument. Add the positional uncertainty of the satellite itself to this and it becomes clear there are issues to be addressed. Gaia offers the community accuracies of about 24 microarcseconds for sources G=15 and brighter. Each source will be observed about 80 times[8] on average. By constructing a model for the stars position based on the multiple positions observed over the mission

[7] http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_gaia_telescope.pdf

[8] http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_astrometric_instrument.pdf

and the satellite pointing (and other factors) a more accurate estimation of the observational parameters may be computed. In a similar way the positions of the multiple sources observed by the pair of astrometric telescopes over time may be used to build a more accurate representation of the satellite attitude. This is a feedback system, where improving one measurement improves the other; the intention is to iterate over these solutions until convergence is reached. This is of course an iterative and distributable solution for the otherwise intractable problem of solving millions of equations for the astrometric parameters of millions of stars.

The iteration of the algorithms requires access to the data in the spatial and time domains which is discussed further in Section 3.2.1. Some of the algorithms require access to all observations of a given source (e.g. astrometry and photometry calculations), other algorithms require all the data in time series (e.g. to reconstruct the satellite pointing accurately or to calculate chromaticity [9] or other calibration values over time). This process is referred to as the Astrometric Global Iterative Solution (AGIS).

After, or simultaneously with, the calculation of the astrometry the other parameters must also be calculated. Expert groups around Europe will deal with photometry, bright stars, variable stars and spectrometry to name but a few. Many of these tasks in turn rely on output from the AGIS and each other. The organisation of this gargantuan task is laid out in the response to ESA's announcement of opportunity (FM-030) and a brief overview was presented by the author at ADASS 2005 (O'Mullane et al., 2006). Fig. 1.4 shows the notion of a processing hub at ESAC in Spain which receives all new data as well as updates from the other processing centres. AGIS is just one of many processes occurring within one of many DPCs.

[9] http://www.rssd.esa.int/SA/GAIA/docs/info_sheets/IN_chromaticity.pdf

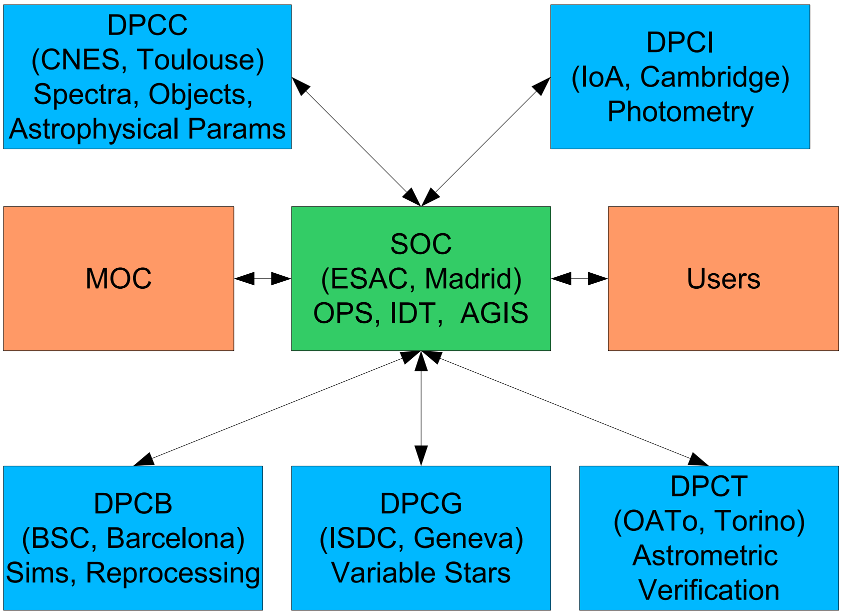


Figure 1.4: ESAC forms a hub for the overall iterative processing of the Gaia data. New data arrives to ESAC where Initial Data Treatment (IDT) and AGIS run. The processed data is sent out to the other DPCs for their parts of the processing. The results are then sent back to ESAC. This process will be repeated until the results meet the requirements of the catalogue.

Chapter 2

Gaia Astrometric Processing

> The new reduction of the *Hipparcos* data, as presented in this book, differs in a fundamental way from the old reduction.
>
> **Floor van Leeuwen, Member NDAC, Hipparcos and Gaia Science teams, DPAC CU5 leader (van Leeuwen, 2007, Chapter 1.4).**

2.1 Introduction

Most of the material in this section comes from technical notes written by Lennart Lindegren, as cited in the text. The aim of the section is to present the reader with a complete overview of the AGIS equations which was not available in any other single place at the time of writing - a co-authored paper is now published on this topic (Lindegren et al., 2012). This paper is included in Appendix C for the reader's convenience and provides a more recent formulation of the astrometric problem.

Section 2.2 goes back to the very high level and basic concepts of the global solution as hinted at in the Hipparcos catalogue. The remaining sections introduce the AGIS models and discuss the Global Iterative Solution. The quote from van Leeuwen at the beginning of this chapter holds true for this text also. The Global Iterative solution is more in line with van Leeuwen's processing than the original great circle approach of Hipparcos.

In this section many math variables are used, to assist the reader they are summarised in Table 2.1.

2.2 Principles of Gaia Astrometry

A prescient view of the Gaia principle is provided in the Hipparcos catalogue (ESA, 1997, Volume 3 Chapter 23). With the great knowledge accumulated on Hipparcos a

Table 2.1: A handy reference table of math variables used in the astrometric solution description.

Var.	**Description**	Ref.
$\mathbf{A}$	Observation matrix	Eq. 2.4
$\mathbf{a}$	The astrometric parameters	Section 2.4
$\mathbf{B}$	Element of standard SVD	Section 2.4.3
$\mathbf{b}$	Residuals, observed minus calculated	Eq. 2.4
$\mathbf{e}$	Additional data such as ephemeris of satellite	Eq. 2.3
f	Function calculating position based on current estimate	Eq. 2.5
$\mathbf{f}$	vector of field angles	Eq. 2.17
G	Detector coordinate	Section 2.2
H	Detector coordinate	Section 2.2
h	Set of uncorrelated observation equations	Eq. 2.31
$\mathbf{g}$	The measurement vector, field angles of observation	Eq. 2.1
$_i$	Subscript typically denoting some source	Section 2.4
$\mathbf{n}$	Nuisance parameters	Eq. 2.1
$\mathbf{q}$	Attitude quaternion	Appendix A
$\mathbf{R}$	Element of standard SVD	Section 2.4.3
$\mathbf{S}$	Element of standard SVD	Section 2.4.3
t_k	An instant of time $_k$	Section 2.4
$\mathbf{U}$	Element of standard SVD	Section 2.4.3
$\mathbf{u}$	Calculated direction of star CoMRS	Eq. 2.3
$\mathbf{V}$	Element of standard SVD	Section 2.4.3
$\mathbf{W}$	Down weight matrix	Eq. 2.7
$\mathbf{y}$	Observed field angles of star	Eq. 2.2
α	Azimuthal angle of observation	Section 2.4
δ	Angular distance from the celestial equator	Section 2.4
ϖ	Annual parallax	Section 2.4
$\mu_{\alpha*}$	Proper motion in α scaled by δ	Section 2.4
μ_δ	Proper motion in delta	Section 2.4
μ_r	Motion in the line of sight direction	Section 2.4
η	Along scan local plane coordinate	Section 2.6
ζ	Across scan local plane coordinate (elevation)	Section 2.6
λ	Variable used for Lagrange multiplier	Section 2.5.1
ω	Angular velocity vector	Eq. 2.25
σ	Typically an error value	Eq. 2.4

view was cast to the future and the general principle of a global astrometric mission is succinctly formulated as the following minimisation problem:

$$\overset{min}{\mathbf{a},\mathbf{n}} \|\mathbf{g}^{\mathbf{obs}} - \mathbf{g}^{\mathbf{calc}}(\mathbf{a},\mathbf{n})\|_M \tag{2.1}$$

Here **a** is the vector of unknowns describing a star's barycentric motion (see Section 2.4). These parameters are extracted from a number of observations of the star. The observations of a star are expressed in detector coordinates denoted G and H. The observation at time t_k is thus represented by the measurement vector $\mathbf{g_k} = (G_k, H_k)'$ and associated statistics. $\mathbf{g}^{\mathbf{obs}}$ represents the vector of all measurements and $\mathbf{g}^{\mathbf{calc}}$ represents the vector of detector coordinates calculated from the astrometric parameters. **n** is a vector of parameters[1] which are not of interest for the astronomical problem but are required for realistic modelling of the data. The norm is calculated in a metric M defined by the statistics of the data, this is classically referred to as error weighting.

2.3 The Basic Gaia Problem

Fig. 2.1 formulates the problem very simply: which set of astrometric parameters best predict the positions of the star images in the Gaia focal plane at the given observation times? Therefore the derived question: what is the mapping (model, transformation or relation) between astrometric parameters and the position in the focal plane at t_{obs}?

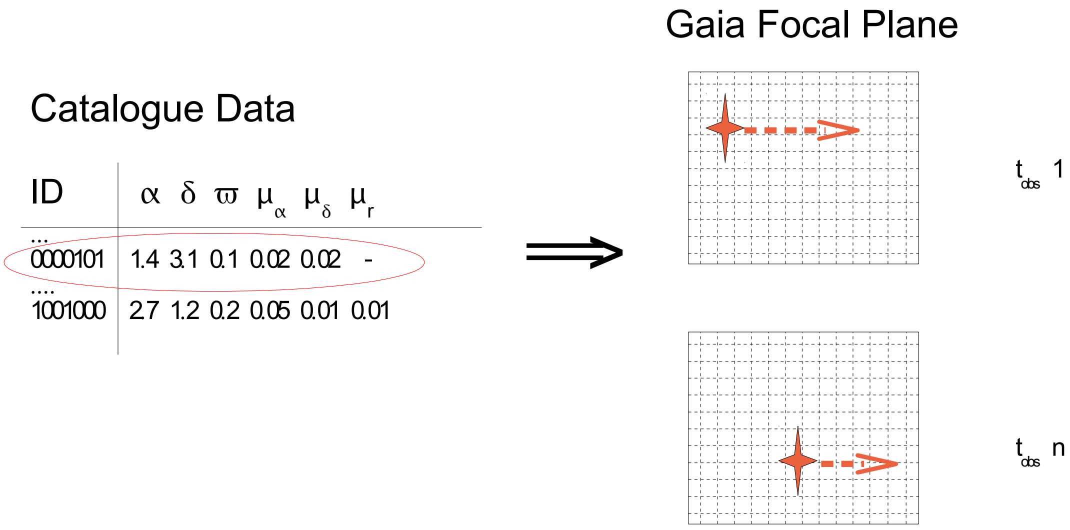
The main goal of AGIS is the determination of the astrometry and the parameters defining this mapping that best reproduce the ensemble of Gaia observations.

It is pointed out in (ESA, 1997) that **n** in Eq. 2.1 is quite dependent on the instrument in use. Like Hipparcos, Gaia is a scanning satellite. The mapping or modelling of the observables **g** is done by three successive transformations:

1. from astrometric parameters to the celestial directions of a star at the instant of observation, using an astrometric model (see Section 2.3.1);
2. from celestial to instrument frame directions using an attitude model (see Section 2.3.2);
3. and finally from instrument directions to detector coordinates using an instrument model (see Section 2.3.3).

This remains true for Gaia today - although the measurement vector is perhaps a little more complex than previously considered. Gaia will operate in Time Delay Integration (TDI) mode in a manner more complex than Hipparcos. However according to (Bastian & Biermann, 2005) we may still use the same astrometric model, which assumes

[1] Termed nuisance parameters.



ID	α δ ϖ μ_α μ_δ μ_r
...	
0000101	1.4 3.1 0.1 0.02 0.02 -
....	
1001000	2.7 1.2 0.2 0.05 0.01 0.01

Figure 2.1: The basic Gaia problem is to find the astrometric parameters best predicting the focal plane observations of the source.

an instantaneous measurement, for Gaia provided we properly define and interpret the quantities in the equations. Below, these models are expanded for Gaia but first we take a higher level look at the scheme.

2.3.1 Astrometric parameters to apparent sky position

The six astrometric parameters (α δ ϖ μ_α μ_δ μ_r see Section 2.4) - catalogue data - describe the positions and velocity of a source at a given reference time in the standard reference system, the ICRS. This position is thus referred to as the barycentre of the solar system.

This six parameter model implies a linear motion of the star in the ICRS and is intrinsically a 3D framework. When projected on to the 2D apparent sky it translates to an angular quasi-linear motion in equatorial coordinates. This is quasi-linear if the so called “perspective acceleration”, due to the fact that the motion is actually 3D with a radial component, is ignored. This is depicted in Fig. 2.2.

The source is not, however, observed from the barycentre of the solar system but from Gaia. The location of the spacecraft at L2 introduces a perspective effect into the apparent motion of the star (parallactic effect) so that the motion is no longer quasi-linear but reflects the global motion of Gaia around the Sun (see Fig. 2.3). In addition to this effect, the apparent position of the source as seen from Gaia depends on other additional phys-

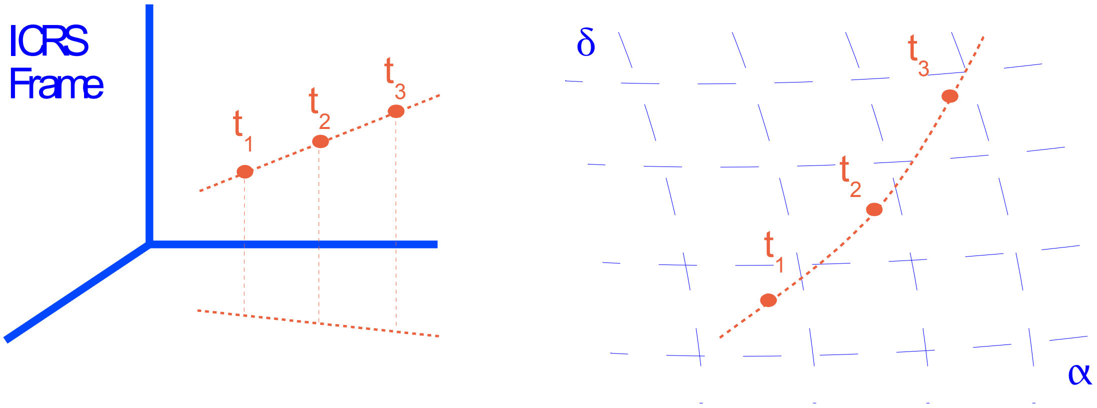


Figure 2.2: A star's motion projected into the 2D apparent sky translates to an angular quasi-linear motion in the usual equatorial coordinates (the departure of the linearity, only noticeable for very close and very fast stars, is the so-called perspective acceleration). The left diagram depicts motion in 3D ICRS while the right shows the quasi-linear motion in angular coordinates (proper motion).

ical effects: the finite velocity of light (stellar aberration) and the light bending by the solar system bodies (relativistic corrections), etc. Taking into account these additional effects the apparent position of the star as seen from Gaia ($\alpha_{app}, \delta_{app}$) is obtained.

The full mapping chain combining all these effects maps the reference astrometric parameters into the apparent angular position at observation time. This is noted in this thesis as Eq. 2.3. This is discussed further in Section 2.4.

2.3.2 Mapping celestial directions to the instrument frame

The second mapping takes into account the need to rotate the apparent position of the source ($\alpha_{app}, \delta_{app}$) to the view point of the Gaia instruments. The instruments (and indeed the spacecraft itself) are not permanently aligned to the ICRS, Gaia is tumbling and revolving in space and this dynamic behaviour must be captured. Consider the view from inside Gaia, if we sat in there observing we would see objects passing by while in fact the movement would be our own. This viewpoint is captured by the Scanning Reference System (SRS). To translate from this to ICRS we would need to translate and rotate our positions. The Centre of Mass Reference System (CoMRS) (BAS-003), aligned with ICRS but centred on Gaia makes this a two step and simpler process. By fixing Gaia as the center of the system we need only worry about the instantaneous rotation of the SRS (instruments) relative to the CoMRS (almost ICRS).

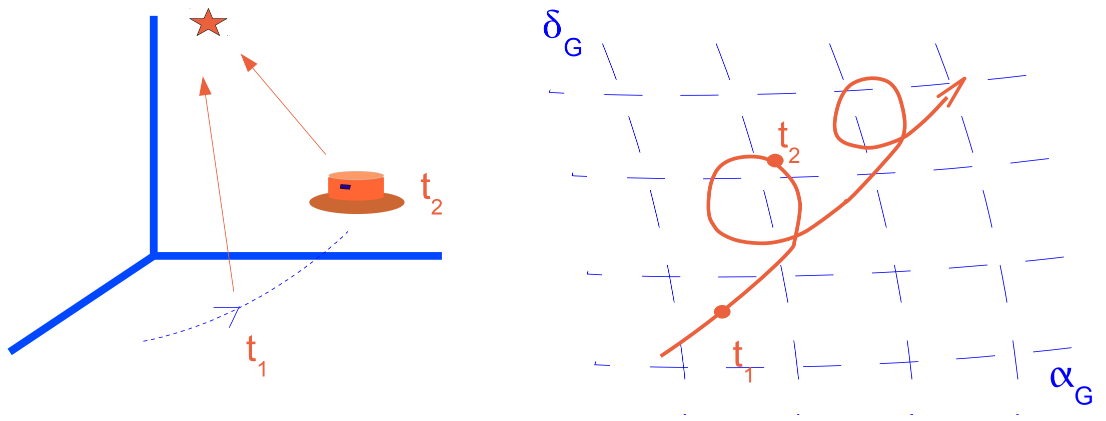


Figure 2.3: Observations at different times taken from Gaia contain parallactic effects due to the motion of the satellite.

This rotation is given by the orientation of the spacecraft in CoMRS as a function of time, that is the Gaia attitude (see Section 2.5). The attitude provides the transformation (rotation) between the two reference systems at any time as depicted in Fig. 2.4.

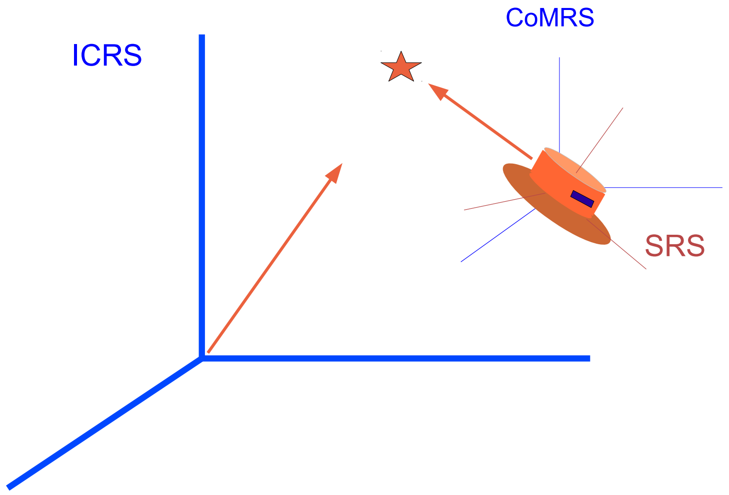


Figure 2.4: The apparent position of a source in the CoMRS may be transformed to a SRS (Scanning Reference System) position through a rotation.

Thus the attitude allows the apparent coordinates of a source to be mapped into angular coordinates in the Scanning Reference System (see Fig. 2.4):

$$(\alpha_{app}, \delta_{app}) \xrightarrow{attitude} (\eta, \zeta).$$

This is expressed mathematically in Section 2.5 specifically in equation Eq. 2.17. in the equations below the field angles of a specific observation i are referred to using $\mathbf{y}_i$:

$$\mathbf{y}_i = (\eta_i, \zeta_i). \tag{2.2}$$

while $\mathbf{u}_i$ is used for the apparent coordinates.

2.3.3 Mapping SRS coordinates to pixel coordinates

The final mapping step involves translating the angular position in the SRS into the fiducial position in the focal plane or, more accurately, the pixel coordinates of the image in the focal plane CCDs.

This mapping is expressed in terms of *calibrations* (see Section 2.6) these calibrations that vary with time summarise all the instrument effects determining the position of the source image on the CCDs: optical projection, position of CCDs in the focal plane, intra CCD effects, etc. The mapping that these calibrations describe is noted in Section 2.6.1 and expressed in equation Eq. 2.49. Fig. 2.5 depicts the projection of the source photons on to the CCD within Gaia.

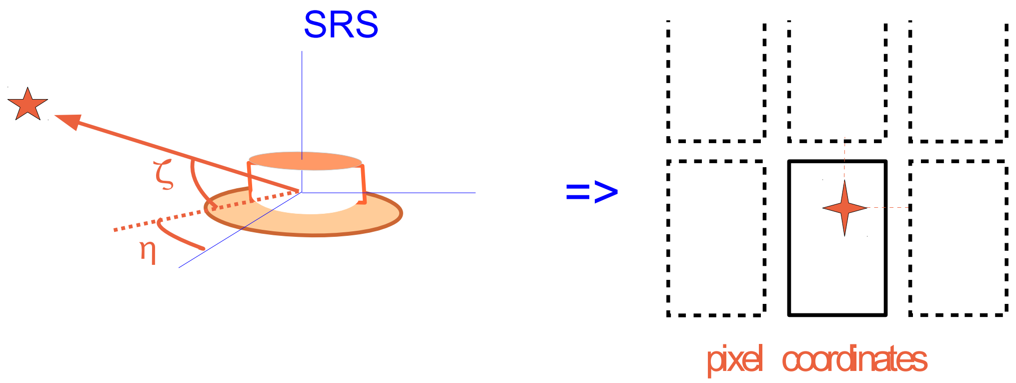


Figure 2.5: The apparent position of a source in the SRS must be transformed to pixel coordinates on the CCD. The optical path is modelled by a set of time varying calibrations.

2.4 Astrometric model

In the Gaia model the satellitocentric direction to star i at time t_k depends on six parameters:

α_i	azimuthal angle of observation a.k.a. right ascension
δ_i	angular distance from the celestial equator (north or south) a.k.a. declination
ϖ_i	annual parallax, the apparent shift in angular position when viewed from opposite sides of the sun
$\mu_{\alpha *i}$	$\mu_{\alpha *i} = \mu_\alpha \cos\delta$ proper motion in alpha
$\mu_{\delta i}$	proper motion in delta
μ_{ri}	radial proper motion, motion in the line of sight direction

It should be noted that these simple parameters may only be used to model a subset of Gaia sources. These sources, which shall be used in the global iterative solution, are termed *primary sources* and are typically single stars with sufficient observations. The addition of μ_r is not always necessary as discussed in Section 4.1.

Calculation of the observable direction (in CoMRS) to the star also requires some given data e which includes for example the barycentric ephemeris of the satellite and is written symbolically as:

$$\mathbf{u_{ik}} = u(\mathbf{a_i}|t_k, e) \tag{2.3}$$

2.4.1 Source Update - Observation Equations

For Gaia each source transiting the focal plane is *observed* by each of the AF CCDs (see Fig. 1.3), for the purposes of astrometry we may think of these observations as a time and pixel coordinate on the CCD. To derive or update the astrometric parameters we must fit them to these observations. This may be done with a robust least squares fit of the n astrometric parameters where $n = 5$ *or* 6 as outlined in Section 2.4 to the multiple observations (or focal plane transits) of the source during the mission. By mission end most of the observed sources will have crossed the focal plane about 80 times which implies an observation in the SM and each of the nine AF strips giving typically $m \approx 800$ observations. We are mainly interested in the centroid timings of the AF1-9. This may be expressed in a standard least squares manner (also in GAIA-LL-055):

$$\mathbf{Ax} \sim \mathbf{b} \pm \sigma \tag{2.4}$$

Where $\mathbf{A}$ is the design matrix (observation matrix in our case) of dimension m x n, $\mathbf{b}$ is an $m-$vector of residuals (observed minus calculated) and σ is an $m-$vector of measurement uncertainties (or noise), $\mathbf{x}$ is an $n-$vector of updates to the astrometric parameters. Put more simply $\mathbf{A}$ contains a row for each observation with an evaluation of the basis function for that observation (see Eq. 2.6) based on the observed timings while $\mathbf{b}$ contains the difference from our current model for this source (a measure of how well or not we are doing). Adding updates ($\mathbf{x}$ on the LHS) and noise (σ on the RHS) LHS and RHS should more or less agree but will never be exactly the same. Both sides approach the noise level which we can never know precisely.

Put more mathematically this is an approximation ($\sim$) since the system is very much overdetermined i.e. $m \gg n$, hence general agreement is sought. As stated $\mathbf{b}$ is a matrix of observed minus calculated such that i^{th} element is:

$$\mathbf{b}_i = \mathbf{y}_i - f_i(\mathbf{a}, \mathbf{q}) \tag{2.5}$$

Where $\mathbf{y}_i$ is the measured ('observed') field angle of the source and f_i is the calculated field angle based on the current best estimate of the astrometric parameters $\mathbf{a}$ and the attitude $\mathbf{q}$ (see also Eq. 2.17). It should be noted here that f_i is not quite the same thing as $\mathbf{g}^{calc}(\mathbf{a}, \mathbf{n})$ in Eq. 2.1, a little more work remains to be done.

The elements of $\mathbf{A}$ provide a measure of the sensitivity of b_i to changes in parameter a_j and are given as follows:

$$A_{ij} = \frac{\partial f_i(\mathbf{a}, \mathbf{q})}{\partial a_j} \tag{2.6}$$

Lindegren (GAIA-LL-055) points out several factors to be noted about the source update in this form:

1. The m measurements are in general heteroscedastic (σ_i are not all the same), so a weighted least-squares solution should be used.

2. A small fraction of outliers must be expected, so the solution should be robust against errors of arbitrary size.

3. The model $f_i(\mathbf{a}, \mathbf{q})$ is non-linear, so the updating process must be iterated until convergence. Iteration is also needed to cope with outliers.

4. Depending on the type of source and on the number and distribution of observations, it may be desirable to update less than the maximum set of $n = 6$ astrometric parameters. It is always desirable to solve for fewer parameters and perspective acceleration is an important factor in very few solutions. In fact $n = 5$ will be the normal case, but fewer may also be used (e.g. when there are insufficient observations for parallax determination we may solve for positions only). This corresponds to making solutions that are constrained to zero updates in some parameters.

5. On some sources the measurements may be degenerate, e.g. because there are too few observations, resulting in a non-unique least-squares solution. The algorithm should in such cases still produce a sensible update, if possible, and indicate that it is unreliable.

6. The calculation of $f_i(\mathbf{a}, \mathbf{q})$ and its derivatives is time-consuming and recomputing them should be avoided when not strictly necessary.

7. Apart from the updated astrometric parameters, the algorithm should also supply their variances and correlations, together with suitable statistics and diagnostics.

8. The estimated variances should take into account error sources not included in the formal observation uncertainties σ.

2.4.2 SVD solution

A singular value decomposition (SVD) may be used to solve for $\mathbf{x}$ in the above equations. First let us introduce the m x m weight matrix:

$$\mathbf{W}_0 = diag(\sigma_1^{-2}, \ldots \sigma_m^{-2}) \tag{2.7}$$

which may be used to obtain the normalised observation matrix:

$$\hat{\mathbf{A}} = \mathbf{W}^{1/2}\mathbf{A} \tag{2.8}$$

and the normalised right hand side (residuals):

$$\hat{\mathbf{b}} = \mathbf{W}^{1/2}\mathbf{b} \tag{2.9}$$

This leads to the weighted least squares solution, minimise $|\hat{\mathbf{A}}\mathbf{x} - \hat{\mathbf{b}}|$. We do not need to solve the normal equations but may use an SVD approach using the null space.

The SVD of $\hat{\mathbf{A}}$, following the notation of (Riley et al., 2006), is:

$$\hat{\mathbf{A}} = \mathbf{USV}\dagger \tag{2.10}$$

Where:

- $\mathbf{S}$ is a diagonal n x n matrix with the singular values of $\hat{\mathbf{A}}$ in descending order ($\mathbf{S} = diag(s_1 \ldots s_n); s_1 \geq s_2 \geq \ldots \geq s_n \geq 0$) as its entries - the singular values are the square root of the eigenvalues of the normal equations matrix.
- $\mathbf{U}$, an m x n matrix, is the basis for the data (observations) or the left singular vectors of $\hat{\mathbf{A}}$.
- $\mathbf{V}$, an n x n matrix, forms the basis for the model containing the right singular vectors of $\hat{\mathbf{A}}$.
- $\mathbf{U}$ and $\mathbf{V}$ are orthogonal.

The hermitian conjugate † in Eq. 2.10 will simply become ^{T}, the transpose in the case that $\mathbf{V}$ is composed of real numbers (which it is in our case).

Now define $\mathbf{S}^+ = diag(s_1^+ \ldots s_n^+)$ where:

$$s_j^+ = \begin{cases} s_j^{-1} & if\ s_j\ \geq\ \varepsilon s_1 \\ 0 & otherwise \end{cases} \tag{2.11}$$

Here ε is a preselected tolerance level for the ratio between the smallest and largest singular value (s).

2.4.3 Using a basis of null space

Null spaces are used widely in physics and engineering for singular problems of this nature from electronics (Castillo et al., 2005) to guidance and control (Psiaki, 2006).

The null space, or kernel, of $\mathbf{B}$ is the subspace $\mathbf{N}$ for which any vector $\mathbf{n} \in \mathbf{N}$ satisfies:

$$\mathbf{Bn} = 0 \tag{2.12}$$

The null space solutions are homogeneous solutions. They may be used to express all solutions for $\mathbf{B}$ since it may always be added to any solution i.e.

$$\begin{aligned} \text{For some } \mathbf{B}(\mathbf{x}) &= \mathbf{h} \text{ where } \mathbf{x} \notin \mathbf{N} \qquad (2.13)\\ \mathbf{B}(\mathbf{x}+\mathbf{n}) &= \mathbf{B}(\mathbf{x}) + \mathbf{B}(\mathbf{n}) \qquad (2.14)\\ &= \mathbf{B}(\mathbf{x}) + 0 = \mathbf{h} \qquad (2.15) \end{aligned}$$

For such solutions to exist $\mathbf{B}$ must be rank deficient, some columns must be liner combinations of others. In this case we shall have ≈ 800 observations and wish to extract only six astrometric parameters.

The required solution is then given by (Riley et al., 2006, Section 8.19):

$$\mathbf{x} = \mathbf{U}\mathbf{S}^{+}\mathbf{U}^{T}\hat{\mathbf{b}} \tag{2.16}$$

Such a solution is equivalent to solving the normal equations for the source. In the case where the matrix is rank deficient ($r < n$ where r, the rank, is equivalent to the number of non zero rows in $\mathbf{S}^{+}$, a diagonal matrix defined in (2.11))[2] Eq. 2.16 gives the unique solution with the minimum norm $|\mathbf{x}|$.

Such a solution will need to be iterated several times to convergence since $\mathbf{f_i}(\mathbf{a},\mathbf{q})$ is non linear. This means an estimation is made and plugged back in to the original equation (e.g. Eq. 2.16) to find the residual - this is used to find an even better approximation. This manner of updating is dealt with in a little more detail in the attitude section (Section 2.5). Experience shows that such a system always converges in 4 or 5 iterations for *primary* stars.

2.5 Continuous attitude model

The attitude specifies the instantaneous orientation of Gaia in the Centre of Masses Reference System (CoMRS) Frame. CoMRS is derived from ICRS and defined in the same

[2] $\mathbf{S}^{+}$ is almost the same as $\bar{\mathbf{S}}$ in standard texts such as (Riley et al., 2006, Section 8.19) but with some singular value suppression.

manner but with the centre of mass of Gaia as the origin of the coordinate system instead of the solar system barycentre, it is the local rest frame of Gaia (see Fig. 2.4). The axes of CoMRS point in the same direction as the ICRS axes, for a detailed explanation see (BAS-003).

Since Gaia is a scanning satellite we describe the attitude as a continuous function of time; let $\mathbf{c_j}$ represent this continuous attitude function for a time interval τ_j, during which the attitude is stable. Then the field angles, usually denoted (η,ζ), for a source at $t_k \in \tau_j$ are given by:

$$\mathbf{f_{ik}} = f(u_{ik}, c_j | t_k) \tag{2.17}$$

where $\mathbf{f_{ik}}$ is the vector of field angles and u_{ik} is the observed direction as discussed above in section 2.4.

The attitude may be modelled in a number of ways. In the case of Hipparcos the attitude was described differentially to the Hipparcos Nominal Scanning Law by the NDAC consortium, and to the great circles by the FAST consortium, using three Euler angles (ESA, 1997, Vol. 3 Chapter 7). Within the Gaia community it is felt this differential approach should not be taken due to problems it introduced in later processing stages for Hipparcos [3]. Hence we shall model the attitude directly.

Euler's theorem on rotation (Kane et al., 1983) tells us that we need only three angles to cover an arbitrary rotation. In flight dynamics these angles are generally termed roll (α, about X), pitch (β, about Y) and yaw (γ, about Z) [4]. The vector $\mathbf{c}_j$ in (2.17) could for instance be the polynomials describing these three angles giving the rotation between SRS and CoMRS as functions of t in the time interval τ_t.

The three Euler angles however have singularities (about the poles). If we expand it to a rotation matrix removing the singularities we have nine parameters to model. However only three degrees of freedom remain, this leaves six difficult constraints to be respected. William Rowan Hamilton introduced a fourth dimension, with the quaternion (Hamilton, 1847), alleviating these problems. Hamilton noted quaternions may be used to model rotations in three dimensional space very efficiently and in many texts the four components of quaternions used in this manner are referred to by the term Euler parameters (Kane et al., 1983). For some further details on quaternions see Appendix A.

The quadratic sum of its components are constrained to unity since there remains only three degrees of freedom. If we let $\mathbf{e}$ represent a unit vector (i.e. in ICRS for us) and ϕ represent an angle of rotation around that vector the quaternion which may be used to represent the instantaneous attitude (of Gaia) would be defined as follows:

[3] Problems with the differential modelling approach have been discussed in private communications and have not apparently been documented.

[4] It is noted that various other symbols are used for the Euler angles (Korn & Korn, 1961)

$$\mathbf{e} \equiv \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \tag{2.18}$$

$$\mathbf{q} = \begin{bmatrix} e_1 \sin(\phi/2) \\ e_2 \sin(\phi/2) \\ e_3 \sin(\phi/2) \\ \cos(\phi/2) \end{bmatrix} \tag{2.19}$$

The direction cosine matrix, representing the attitude (Kane et al., 1983), in terms of this quaternion is:

$$\mathbf{A} = \begin{bmatrix} 1-2q_2^2-2q_3^2 & 2(q_1q_2+q_3q_4) & 2(q_1q_3-q_2q_4) \\ 2(q_1q_2-q_3q_4) & 1-2q_1^2-2q_3^2 & 2(q_2q_3+q_1q_4) \\ 2(q_1q_3+q_2q_4) & 2(q_2q_3-q_1q_4) & 1-2q_1^2-2q_2^2 \end{bmatrix} \tag{2.20}$$

It should be noted that we still have a constraint on our quaternion that the sum of the squares is unity hence **q** in (2.20) should be normalised or a normalisation factor should be applied to the matrix.

Over time then we would be interested in $\mathbf{A}(t)$ and $\mathbf{q}(t)$. So in fact we would need to model $\mathbf{q}(t)$ which is essentially the attitude curve with respect to time. To model such a curve we need to choose an appropriate basis function, for Gaia the choice is to use a B-spline[5]. B-splines afford a flexible modelling for continuous functions with some history between nodes but also allowing discontinuities to be correctly represented. Splines are a standard way of tackling attitude modelling in spacecraft dynamics.

Hence in (2.17) c_j would in fact be the four coefficients of the four spline knots representing $\mathbf{q}(t)$; these coefficients may be determined by least squares fitting of the predicted values to the observed ones. $\mathbf{q}(t)$ may be calculated using:

- a set of known directions e.g. **u** (2.3)
- the CCD observations of these directions e.g. η, ζ of the observations (2.17)
- a calibration of the instrument parameters e.g. d (2.49)

Each measurement associates a time instant t with certain field angles (η, ζ) in the instrument frame (SRS) which relate to a direction **u**. Hence returning to (2.17) with our quaternion we may write it as:

$$\mathbf{f_{ik}} = f(u_{ik}, q(t_k|c_j)) \tag{2.21}$$

[5] the B stands for basis and was coined by Isaac Jacob Schoenberg

Where c_j are now the B-spline coefficients. The field angles ($\mathbf{f}$) are known here for given t, for the iterative process we need the differential w.r.t. $\mathbf{q}$.

$$\dot{\mathbf{f}} = \frac{df}{dq}\dot{\mathbf{q}} \Leftrightarrow \frac{df}{dt} = \frac{df}{dq}\frac{dq}{dt}$$

Since $\mathbf{q}$ is non linear we must formulate a set of linear equations from which we may derive an approximation of it. Consider $\mathbf{q_0}$ to be the current attitude estimate and $\mathbf{f_o}$ to be the observed field angles such that:

$$\mathbf{q} = \mathbf{q_0} + \Delta\mathbf{q} \tag{2.22}$$

yielding the next best estimate and:

$$\Delta\mathbf{f} = \mathbf{f_o} - \mathbf{f_c} \tag{2.23}$$

the observed minus calculated field angles. Then we may consider:

$$\Delta\mathbf{f} = \frac{df}{dq}\Delta\mathbf{q} \tag{2.24}$$

At this point it is worthwhile to introduce the angular velocity or inertial rotation vector ω (Kane et al., 1983, Sect. 1.11), also written as ${}^{A}\omega^{B}$. This may be seen as an 'operator' which when operating on any vector fixed in B produces the time derivative of that vector in A. Now ω may be expressed in terms of $\mathbf{q}$ (Kane et al., 1983, Sect. 1.11):

$$\omega = 2\dot{\mathbf{q}} \begin{bmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ -q_2 & q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} \tag{2.25}$$

if we multiply this out we find:

$$\omega = 2*(\dot{q}_1\dot{q}_2\dot{q}_3\dot{q}_4) \begin{bmatrix} q_4 & -q_3 & q_2 & q_1 \\ q_3 & q_4 & -q_1 & q_2 \\ -q_2 & q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} = 2* \begin{bmatrix} q_4\dot{q}_1 + q_3\dot{q}_2 - q_2\dot{q}_3 - q_1\dot{q}_4 \\ -q_3\dot{q}_1 + q_4\dot{q}_2 + q_1\dot{q}_3 - q_2\dot{q}_4 \\ q_2\dot{q}_1 - q_1\dot{q}_2 + q_4\dot{q}_3 - q_3\dot{q}_4 \\ q_1\dot{q}_1 + q_2\dot{q}_2 + q_3\dot{q}_3 + q_4\dot{q}_4 \end{bmatrix} \tag{2.26}$$

But our quaternion is constrained $|\mathbf{q}| = 1$ and hence:

$$q_1\dot{q}_1 + q_2\dot{q}_2 + q_3\dot{q}_3 + q_4\dot{q}_4 = \frac{1}{2}\frac{d}{dt}({q_1}^2 + {q_2}^2 + {q_3}^2 + {q_4}^2) = 0 \tag{2.27}$$

If we then take (2.26) it is (in agreement with SAG-LL-030, eq 9):

$$\omega = 2* \begin{bmatrix} q_4\dot{q}_1 + q_3\dot{q}_2 - q_2\dot{q}_3 - q_1\dot{q}_4 \\ -q_3\dot{q}_1 + q_4\dot{q}_2 + q_1\dot{q}_3 - q_2\dot{q}_4 \\ q_2\dot{q}_1 - q_1\dot{q}_2 + q_4\dot{q}_3 - q_3\dot{q}_4 \\ 0 \end{bmatrix} = 2* \begin{bmatrix} q_4 & -q_3 & -q_2 & -q_1 \\ q_3 & q_4 & q_1 & -q_2 \\ -q_2 & -q_1 & q_4 & -q_3 \end{bmatrix} \dot{\mathbf{q}} \tag{2.28}$$

We may then rewrite (2.24), including a conversion matrix $\mathbf{C}$ to convert from the unit vector to actual field angles, as:

$$\Delta\mathbf{f} = \begin{bmatrix} \Delta\eta\cos\zeta \\ \Delta\zeta \end{bmatrix} = 2\mathbf{C} \begin{bmatrix} q_4 & -q_3 & -q_2 & -q_1 \\ q_3 & q_4 & q_1 & -q_2 \\ -q_2 & -q_1 & q_4 & -q_3 \end{bmatrix} \Delta\mathbf{q} \tag{2.29}$$

where:

$$\mathbf{C} = \begin{bmatrix} \sin\zeta\cos\eta & \sin\zeta\sin\eta & -\cos\zeta \\ -\sin\eta & \cos\eta & 0 \end{bmatrix} \tag{2.30}$$

Taking (2.29) and normalising by the detection errors $\sigma_\eta \cos\zeta$ and σ_ζ we have two uncorrelated observation equations for each direction observed at the same time. The attitude model ties the two telescopes together, both are observing simultaneously and it is the large angel between the telescopes which allows us to calculate absolute parallaxes since observations are made relative to different sources in different parts of the sky during the mission. Here we must treat all observations in the same time period (typically between 5 and 120 seconds) from both telescopes. Another way to think of this is in terms of simple triangulation - we have two viewing directions with sources at *known positions* in both directions, we use these simultaneously to work out the orientation of Gaia. The observation equations represent these. Let:

$$\mathbf{h} = \mathbf{B} \cdot \Delta\mathbf{q} \tag{2.31}$$

be the full set of such equations. Then we wish to minimise $|\mathbf{h} - \mathbf{B} \cdot \Delta\mathbf{q}|^2$ subject to the constraint $\mathbf{q_0} \cdot \Delta\mathbf{q} = 0$, the length of $\mathbf{q}$ must be preserved hence any change in $\mathbf{q}$ must be orthogonal to it. Here the $\cdot$ denotes the scalar product[6](Korn & Korn, 1961). The scalar product of two vectors is in fact the equivalent of the product of the transpose of the first times the second hence we may also write $\mathbf{q_0}^T\Delta\mathbf{q} = 0$ for the constraint. In the text below X^T will be used to denote the transpose of the matrix X which is in agreement with the conventions document (BAS-003). Here two ways to minimise such a set of functions are presented, the method of Lagrange multipliers Section 2.5.1 and using a basis of null space Section 2.5.2.

2.5.1 The method of Lagrange multipliers

The method of Lagrange[7] Multipliers is an appropriate way to minimise such a set of functions with constraints (Korn & Korn, 1961, Chpt. 11.5-2). This involves adding an unknown (denoted by λ) times the constraint to our equation as follows:

$$S(\Delta\mathbf{q};\lambda) = |\mathbf{h} - \mathbf{B}\Delta\mathbf{q}|^2 + 2(\mathbf{q_0}^T\Delta\mathbf{q})\lambda \tag{2.32}$$

[6] See also http://mathworld.wolfram.com/DotProduct.html

[7] Also known as Lagrangian

and then performing the unconstrained minimisation of this wrt. $\Delta\mathbf{q}$ and λ. The Lagrange multiplier (λ) then tells us how much the constraint should be emphasised in the solution.

$$\begin{aligned} S &= (\mathbf{h}-\mathbf{B}\Delta\mathbf{q})(\mathbf{h}-\mathbf{B}\Delta\mathbf{q})+2\mathbf{q_0}^T\Delta\mathbf{q}\lambda && (2.33)\\ &= \mathbf{h}^T\mathbf{h}-2\mathbf{h}^T\mathbf{B}\Delta\mathbf{q}+\mathbf{B}\Delta\mathbf{q}\mathbf{B}\Delta\mathbf{q}+2\mathbf{q_0}\Delta\mathbf{q}\lambda && (2.34)\end{aligned}$$

also note:

$$S^T = \mathbf{h}^T\mathbf{h}-2\Delta\mathbf{q}^T\mathbf{B}^T\mathbf{h}+\Delta\mathbf{q}^T\mathbf{B}^T\mathbf{B}\Delta\mathbf{q}+2\Delta\mathbf{q}^T\mathbf{q_0}\lambda \qquad (2.35)$$

To minimise this we may set the partial derivatives w.r.t. $\Delta\mathbf{q}$ and λ to zero:

$$\begin{aligned} \frac{d(S^T)}{d(\Delta\mathbf{q})} &= -2\mathbf{B}^T\mathbf{h}+2\mathbf{B}^T\mathbf{B}\Delta\mathbf{q}+2\mathbf{q_0}^T\lambda=0 && (2.36)\\ \frac{d(S)}{d(\lambda)} &= 2\mathbf{q_0}\Delta\mathbf{q}=0 && (2.37)\end{aligned}$$

giving a system with five unknowns ($\Delta\mathbf{q}$ a quaternion, λ):

$$\begin{aligned} \mathbf{B}^T\mathbf{B}\Delta\mathbf{q}+\mathbf{q_0}\lambda &= \mathbf{B}^T\mathbf{h} && (2.38)\\ \mathbf{q_0}\Delta\mathbf{q} &= 0 && (2.39)\end{aligned}$$

Pre-multiplying by $(\mathbf{B}^T\mathbf{B})^{-1}$:

$$\begin{aligned} \Delta\mathbf{q}+(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}\lambda &= (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{h} && (2.40)\\ => \Delta\mathbf{q} &= (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{h}-(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}\lambda && (2.41)\end{aligned}$$

and using (2.41) in:

$$\begin{aligned} \mathbf{q_0}^T\Delta\mathbf{q} &= 0 && (2.42)\\ => \mathbf{q_0}^T((\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{h}-(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}\lambda) &= 0 && (2.43)\\ \mathbf{q_0}^T(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{h}-\mathbf{q_0}^T(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}\lambda &= 0 && (2.44)\\ \frac{\mathbf{q_0}^T(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{h}}{\mathbf{q_0}^T(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}} &= \lambda && (2.45)\end{aligned}$$

inserting this in (2.41):

$$\Delta\mathbf{q} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}\mathbf{h}-(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}\frac{\mathbf{q_0}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}\mathbf{h}}{\mathbf{q_0}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}} \qquad (2.46)$$

letting $\mathbf{a} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}\mathbf{h}$ and $\mathbf{b} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{q_0}$ thus an equation similar to SAG-LL-030, eq 15 restated here in similar notation to the above:

$$(\mathbf{B}^T\mathbf{B})[\mathbf{a}\ \mathbf{b}] = [\mathbf{B}^T\mathbf{h}\ \mathbf{q_0}] \tag{2.47}$$

may be solved to find **a** and **b** and then [8]:

$$\Delta\mathbf{q} = \mathbf{a} - \mathbf{b}\lambda, \text{ where } \lambda = \frac{\mathbf{q_0}^T\mathbf{a}}{\mathbf{q_0}^T\mathbf{b}} \tag{2.48}$$

One problem with this method is the introduction of an extra unknown parameter λ.

2.5.2 Using a basis of null space

Null spaces have already been discussed in Section 2.4.3. This approach is preferred over the Lagrange Multipliers Section 2.5.1 since Lagrange Multipliers introduce another variable to the system and require inversion of large or incomputable matrices. Here for attitude it must be ensured the time steps are large enough to have many more observations than unknowns. This is may be formulated in a manner analogous to (2.16) and solved using SVD as described in Section 2.4.2 (LL-071).

2.6 Instrument model

Finally we must convert from field angles **f** to detector coordinates **g** i.e.G, H:

$$\mathbf{g_{ik}} = g(\mathbf{f_{ik}}, \mathbf{d_l}|t_k) \tag{2.49}$$

where d is the instrument parameter vector, geometric calibration parameters for the CCDs such as orientation, scale and mechanical distortions. These parameters will be defined on timescales of hours or months as needed or could be continuously modelled. This transformation for Gaia is quite involved (Bastian & Biermann, 2005), yet for our purposes we may consider an instantaneous position for the source in the field of view.

2.6.1 Calibration Model

Taking a further look at the d parameters of (2.49) we consider the calibration model. The calibration model discussed here is the so called “geometric” calibration model i.e. the mapping of CCD coordinates (pixels) to field angles (η, ζ). Position shifts may be introduced by a range of physical features of the instrument and the sources themselves. Some of these are discussed in Section 4 and are not considered here, but

[8] SAG-LL-030, eq 16 should have $\mathbf{q_0}$ as in (2.47)

may be dealt with in a similar way. Calibration is considered to occur over different time periods known as "calibration units". Calibration units take into consideration all observations during their time period, they may overlap and indeed many shorter units may be contained inside a unit covering a longer period. In these cases of overlapping an observation is considered in all units in which it falls.

The following index notation, as outlined in (LL-063), is used:

- l identifies the observation (the transit of one source across a single CCD chip)
- i identifies the source
- f identifies the FOV (Astro-1 or Astro-2)
- n identifies the CCD chip
- m identifies the pixel column on the chip
- j identifies a 'short' calibration time interval (e.g., one month)
- k identifies a 'long' calibration time interval (e.g., the whole mission)

Within the astrometric global solution all other indexes are known for any given observation (l). The nominal geometric calibration is given by the nominal along-scan coordinate η_n^0 associated with each CCD chip (both for SM and AF), together with the nominal basic angle, and the nominal across-scan coordinate ζ_n^0 of each SM chip. These nominal values are never changed. The actual calibration is expressed as small corrections to these nominal angles.

2.6.1.1 Calibration parameters

As outlined in (LL-063) there are three types of calibration parameter to be considered.

1. Large-scale, along-scan geometric correction, $\Delta\eta$. There is one such value for each CCD in the SM and AF fields. Moreover, the values are in general different in Astro-1 and Astro-2 (for the same CCD) due to optical distortion. The values are assumed to be constant only on a short time scale (1 month). $\Delta\eta$ thus depends on indices f, n and j.
2. Small-scale, along-scan geometric correction, $\delta\eta$. There is one such value for each pixel column and CCD. For a given CCD and pixel column, however, the same value applies in Astro-1 and Astro-2, since its physical origin is assumed to be on the CCD chip itself. The values are also assumed to be constant on a long time scale (whole mission). $\delta\eta$ thus depends on indices n, m and k.
3. Large-scale, across-scan geometric correction, $\Delta\zeta$. There is one value per CCD in the SM area, and different values apply in each field. Although the physical

origin is the same as that of $\Delta\eta$, the across-scan component is assumed to be constant on a long time scale, since the calibration requirement is much relaxed in the across-scan direction. $\Delta\zeta$ thus depends on f, n and k.

Hence the field angles f of (2.49) are more precisely for some l:

$$\eta_l = \eta_n^0 + \Delta\eta_{fnj} + \delta\eta_{nmk} \tag{2.50}$$

$$\zeta_l = \zeta_n^0 + \Delta\zeta_{fnk} \tag{2.51}$$

There are a number of constraints enforced on these equations to ensure the parameters are uniquely determinable, these are:

$$\langle\Delta\eta_{fnj}\rangle_j = 0 \qquad \text{(for all } j\text{)} \tag{2.52}$$

$$\langle\delta\eta_{nmk}\rangle_{nk} = 0 \qquad \text{(for all } n,k\text{)} \tag{2.53}$$

$$\langle\Delta\zeta_{fnk}\rangle_{nk} = 0 \qquad \text{(for all } n,k\text{)} \tag{2.54}$$

Here $\langle X_{abc}\rangle_{ab}$ is the unweighted mean of all values of X_{abc} over the whole applicable range of the index c.

Constraint 2.52 concerns the origin of the along scan field angle per time interval j. The different FOVs are not differentiated in this case rather the basic angle is used to differentiate between them. This also means any detectable basic angle variation would show up in this term.

Constraint 2.53 implies the mean small scale correction is zero. This ensures independence between small and large scale calibrations.

Constraint 2.54 concerns the across scan field angle origin. Again this is for each time interval j and each FOV f.

2.7 Resolving the astrometric measurements

Now we may put the three models together. Starting with the instrument model in Eq. 2.49 and putting in attitude we get:

$$g_{ik} = g(f(u_{ik}, c_j|t_k), d_l|t_k) \tag{2.55}$$

Next expanding the astrometric model we have:

$$g_{ik} = g(f(u(a_i|t_k, e), c_j|t_k), d_l|t_k) \tag{2.56}$$

Which we may write as:

$$g_{ik} = h(a_i, c_j, d_l|t_k, e) \tag{2.57}$$

Hence the expanded version of the observation minimisation Eq. 2.1 looks like this:

$$\overset{min}{a,c,d}\|g^{obs} - h(a,c,d|t,e)\|_M \tag{2.58}$$

The indices are dropped since the norm is over all indices.

2.7.1 Rotation of the solution

The Astrometric solution is self consistent and decoupled from real space. Hence the entire solution may be rotated about an arbitrary axis relative to ICRS. This rotation is clearly visible when looking at AGIS results (Section 5.6.1). Hence any solution must be rotated back to the ICRS, the rotation will be worked out using a least squares fitting of some list of reference sources, in principle a quasar catalogue.

2.7.2 Direct Solution

A direct solution for this data would be computationally intensive. Assuming the c_j and d_l terms could be eliminated we would be left with a dense set of normal equations for the astrometric parameters. Assuming that not all sources were used but only some well behaved subset of one hundred million or so we would still have 6×10^8 unknowns. The solution would require about $n^3/3 \ = 2 \times 10^{24}$ FLOPs the matrix would occupy about $n^2/2 \ = 3 \times 10^{16}$ doubles which is around 30 Petabytes of storage. For Hipparcos in its time the direct solution was considered infeasible, for Gaia, even in the future, a direct solution is most improbable. Bombrun has made an excellent study of this infeasibility in (Bombrun et al., 2010).

2.7.3 Global Iterative Solution

The heart of the global iterative solution is to take each of the three models described in the previous sections and treat the dependencies as given. Hence to solve for the astrometry of a star we assume some attitude, calibration and global parameters, then for calibration we assume globals, attitude and astrometry and so on. The order in which this is done should be arbitrary although solving for the astrometry for the individual sources first seems intuitively correct. Hence the solution would involve four relatively independent blocks of equations where each takes on the form of the general minimisation problem of Eq. 2.1.

The astrometry would be solved for each source and require as input all of the observations of the source as well as the relevant calibration, attitude and global parameters. It would then require a minimisation of the observed less the expected parameters for each source. Essentially the equation is not very different to Eq. 2.1:

$$\overset{min}{a_i}\|g_{ik} - h(a_i,c_j,d_l|t_k,e)\|_M \tag{2.59}$$

In which the index i denotes a single source i and g_{ik} denotes all observations of the source i. The calibration parameters c and instrument parameters d are taken as known to some accuracy for this equation hence there are only $dim(a_i)$ unknowns to solve for, in this case the six astrometric parameters. In fact the instrument and calibration parameters could be combined in this equation to n and it also does not contain the global parameters such as general relativistic light bending which we shall denote γ. Hence the astrometric equation would look like:

$$\overset{min}{a_i} \|g_i k - h(a_i, n_j, \gamma | t_k, e)\|_M \tag{2.60}$$

Assuming some values for astrometry and globals we may solve for the nuisance parameters:

$$\overset{min}{n_j} \|g_i k - h(a_i, n_j, \gamma | t_k, e)\|_M \tag{2.61}$$

The system for global parameters is:

$$\overset{min}{\gamma} \|g_i k - h(a_i, n_j, \gamma | t_k, e)\|_M \tag{2.62}$$

The attitude modelling may be broken into several intervals. Realistically it will have to be broken as continuous real attitude for the satellite will not always be available. But in principle we may solve for attitude in the same way as for the other parameters above.

$$\overset{min}{e} \|g_i k - h(a_i, n_j, \gamma | t_k, e)\|_M \tag{2.63}$$

The solution of these equations would mean the organising or accessing of the observational data in both position order (all observations for source i) and time order (all observations in time period j or k). The global parameters simply require all observations. Each solution requires iteration to convergence itself before its results may be used in another step. The full set of solutions must also be iterated to convergence. This iterative process is well described in (LL-071) and displays convergence after 40 or 50 iterations with very small updates after initial large updates. This is explained in terms of a shift in the dominating error contribution. The larger error contributors are quickly corrected for leaving the smaller error contributors to become dominant, and these take longer to correct, it may take more than 100 iterations to really beat down the correlations. Hence we typically see an initial rapid convergence tailing off in our convergence plots as seen for parallax in the top of Fig. 3.13.

2.7.4 The number of sources in the solution

If we were to use all one thousand million Gaia sources even this global iterative solution would prove quite difficult. Organising the 10^9 source and their 10^{12} observations for access in both spatial and time order, although possible, would prove prohibitive in terms of hardware cost today. This solution will work best for so called ‘well behaved’ stars which are apparently single. The process of selecting these stars is described in detail in Section 4.8.1. In principle a few stars could be iterated in this manner and the resulting reference frame applied to all remaining sources in a single pass. This is the intended approach - a discussion of the precise number of sources needed to make the frame is given in (LL-093).

2.8 Conclusion regarding the astrometric solution

The astrometric reduction of the Gaia data is essential for the overall data reduction task. This section has taken us through the fundamentals of the astrometric solution and why we believe the Global Iterative solution is the best approach to follow. In this section we have examined some of the mathematics involved in AGIS. In the following chapters this chosen solution will be examined from the astronomical (Section 4) and computational (Section 3) perspectives.

Chapter 3

Implementation

> Everything should be made as simple as possible, but not simpler.
>
> **Uwe Lammers quoting Albert Einstein, Nobel Laureate.**

The contents of this chapter regarding AGIS have been published in a slightly modified form as (O'Mullane et al., 2011a). In addition the discussion of Java as a language was extended and published as (O'Mullane et al., 2011b). These papers are included in Appendix C for the reader's convenience.

The efficient implementation in some form of code of the block iterative solution, described in Section 2, is challenging. A first attempt for such a solution during the Hipparcos data processing was abandoned. A basic proof of concept, more a pseudo implementation, using again Hipparcos data and a database management system was presented in (O'Mullane & Lindegren, 1999). A good deal of effort went into scaling this up to Gaia dimensions until finally a degree of success was gained by the ESAC group (O'Mullane et al., 2006). It is this later framework which is presented here and which will continue in development up to the Gaia launch. Hence at whichever point one reads this summary the system will have evolved. The core ideas which have not changed in the many years of development to date are presented here, this should give a good understanding of the system as a whole. In our working solution we have taken to heart the Einstein quote at the start of the section stripping complexity from our code wherever possible - hence although what follows may look complex it is really as simple as possible.

Terms such as *DataTrain*, *Taker* and other concepts presented in this chapter are believed to be uniquely associated with the author. The *Store* concept was introduced for Planck and Gaia missions in ESTEC in 1998 and was first published in (O'Mullane & Lindegren, 1999). The interface approach was also postulated in tech notes as a means for defining sub contracts for specific code modules. The *DataTrain* framework was first introduced for Gaia in 2005 and first mentioned in public at ADASS 2006 (O'Mullane et al., 2006).

In this section the reader should bear in mind that the word *Object* will be used in the computer science or object oriented programming sense. It should not be confused with astronomical object for which, in general, we use the term *Source*.

3.1 Overview of the AGIS system

AGIS is just one of many parts of the Gaia processing, a central or core part certainly but still a part. In the overall design of the Gaia Processing System (JSH-004) the Main Database (JH-004; JH-001) is the central repository of all information. Fig. 3.1 depicts AGIS is this broader context with the Main database.

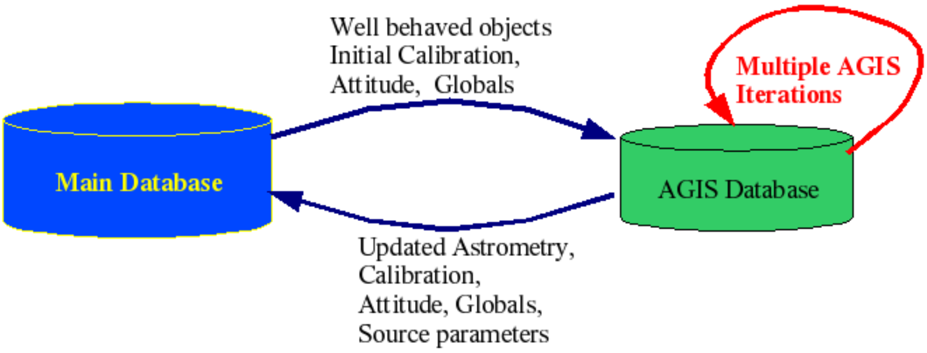


Figure 3.1: AGIS, like other Gaia processing systems, extracts data from the Main database. Updated results are fed back to the main database for merger with results coming from their processing systems.

A simplified overall AGIS picture is presented in Fig. 3.2. Each of the components in the picture may run on any machine apart from the Attitude Update Server which requires a little more memory. The DataTrain, as mediator, is seen in the middle of the left box and is explained in some detail in Section 3.2.2.

It should be noted that the Oracle RAC (Real Application Cluster) may also run on several machines (or nodes) in a cluster to improve data access performance. The data access and storage is abstracted through the *Store* interface which is described in Section 3.2.3.

The algorithms and collectors are described in Section 3.4.

The AGIS system is deployed on a local Gaia dedicated multi-processor machine. All the classes are available on each node but objects will be run on specific nodes according to the configuration specified in the agis.properties file. Objects on different hosts communicate through Remote Method Invocation (RMI). In general a class with the name SomeServer will only have one instance on the cluster while the DataTrains will be numerous with one or several running on each node depending on the number of processors/cores available.

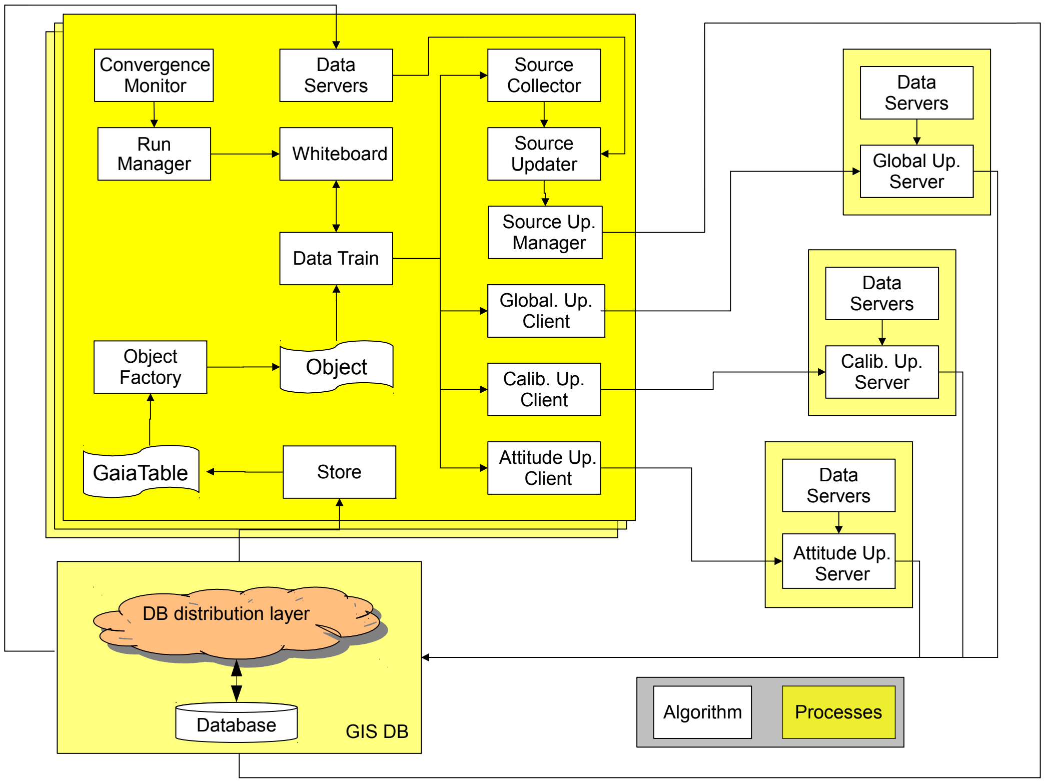


Figure 3.2: This simplified diagram attempts to show a logical overview of AGIS. AGIS runs as many processes on many machines, the machines are not shown here. The large box on the left represents the DataTrain of which there may be a great number running. On the right are the update servers of which there may be only one running in the entire system. An Oracle database underpins all of these processes.

3.2 Data Access and Access Patterns

The key to an efficient implementation of the global solution is in data access. Even with today's machines accessing a large volume (tens of terabytes) in both spatial and temporal order is demanding.

Looking at the four main blocks of the global solution we see each has a seemingly unique data access pattern. These are the access patterns:

Source	All observations of a given source - spatial.
Attitude	All observations within a given time period - temporal.
Calibration	All observations within a given time period falling on a given CCD - temporal/spatial.
Global	All observations any order.

The naive approach would be to go through the data once for each block, iterate them internally, and then perform the next outer iteration. This is indeed the mathematical formulation and is depicted in Fig. 3.3.

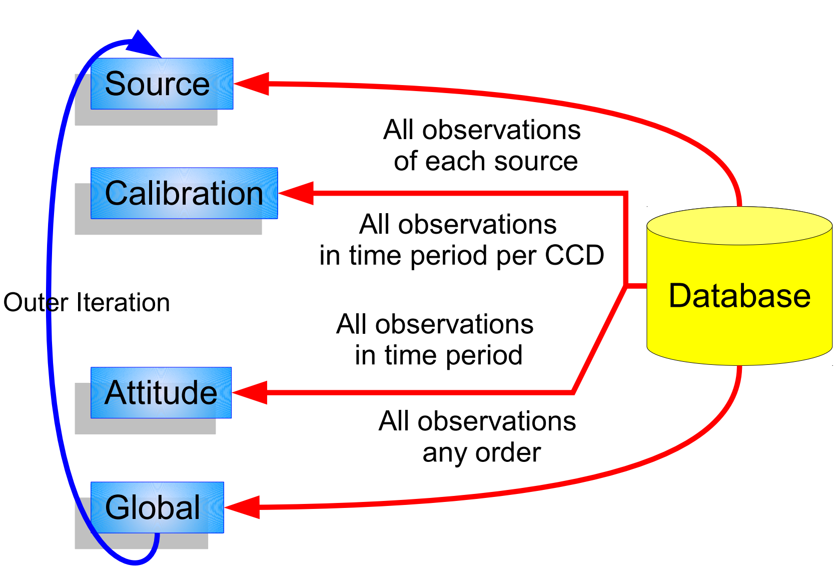


Figure 3.3: Each block of the global solution has a slightly different data access requirement. This could cause four passes through the data for each outer iteration. It is immediately clear that some of these could be combined e.g. Calibration and Attitude together and Source and Global.

Running through the approximately ten terabytes of data four times per iteration is rather daunting. Consider even ten iterations to convergence would yield thirty extra reads of this large volume of data.

Immediately though we see that Calibration and Attitude are similar enough and perhaps could be combined. Global is order independent and as such could be combined with the data access of any of the other blocks for example Source. Indeed this was already remarked in (O'Mullane & Lindegren, 1999) where the prototype made just two passes through the data for each iteration rather than four.

The question after this is could one pass through the data per iteration be enough?

3.2.1 A Question of order

Let us assume that this could be executed in one pass then what would be the impact of the ordering. There are two primary orderings we may choose: spatial or temporal.

3.2.1.1 Temporal Ordering

If we assume a temporal ordering then what would happen in each block? Clearly for the Attitude we may read the data once and break it in time chunks suitable for attitude processing - we may then process a time interval and finish with it. With a small buffer we may accumulate the observations required for the Calibration and also finish with calibrations in a timely manner during the same pass through the data. For Global the order is immaterial.

The problem here comes with Source. Since all sources are observed many times over the entire mission if we process in time order we must accumulate the data for each source until we have all observations of it. This will not happen until we have seen all of the data - only then can we be certain that no more observations of a given source will show up. This would effectively mean all observation data would end up in memory which is undesirable if not completely infeasible. The alternative is another pass through the data in spatial order. Since we must wait until the end of the pass the updated Calibration, Attitude and Globals could already be used for the Source update.

3.2.1.2 Spatial Ordering

If we assumed all observations of a source were clustered together, in a spatial ordering, the story is quite different. Now we may process each source to find its new astrometric solution which we may immediately write out to disk. Since that source is finished with, the updated parameters may be used to find its contributions to the globals. The situation for Attitude and Calibration is however that all contributions from all observations must be accumulated until the end of the pass through the data - only then may the calibration and attitude updates be calculated. It is important to note it is not the observations which must be held but their contribution to the matrices of attitude and calibration which is much smaller than the accumulation of source matrices in Section 3.2.1.1. Hence in this manner one pass may be made though the data for each outer iteration as depicted in Fig. 3.4.

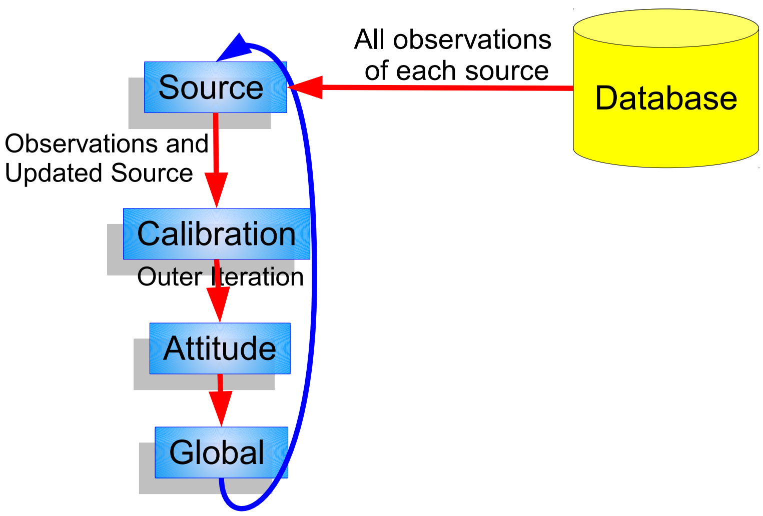


Figure 3.4: With a little accumulation in Calibration, Attitude and Global a complete iteration may be made in one pass though the data. Hence the optimal ordering is spatial. Furthermore the updated source may already be used in the other blocks.

This clearly represents a better approach to the ordering from a technical point of view. Additionally, somewhat holistically perhaps, it is more natural to keep astronomical data of the same part of the sky together and easily accessible. Hence the AGIS database has observations of the same source sequentially grouped together on disk.

3.2.2 Getting data to algorithms, the *DataTrain*

Throughout the Gaia processing there are choices to be made concerning data access patterns such as those outlined in Section 3.2.1. The ideal approach, for efficiency, is a data driven approach whereby data is accessed in the sequential order in which it is stored. Hence rather than algorithms requesting data they should be presented with data by a *mediator*. The mediator pattern (Gamma et al., 1994) is a very powerful tool for decoupling software modules. The implementation of the mediator for the astrometric solution is called the *ElementaryDataTrain*.

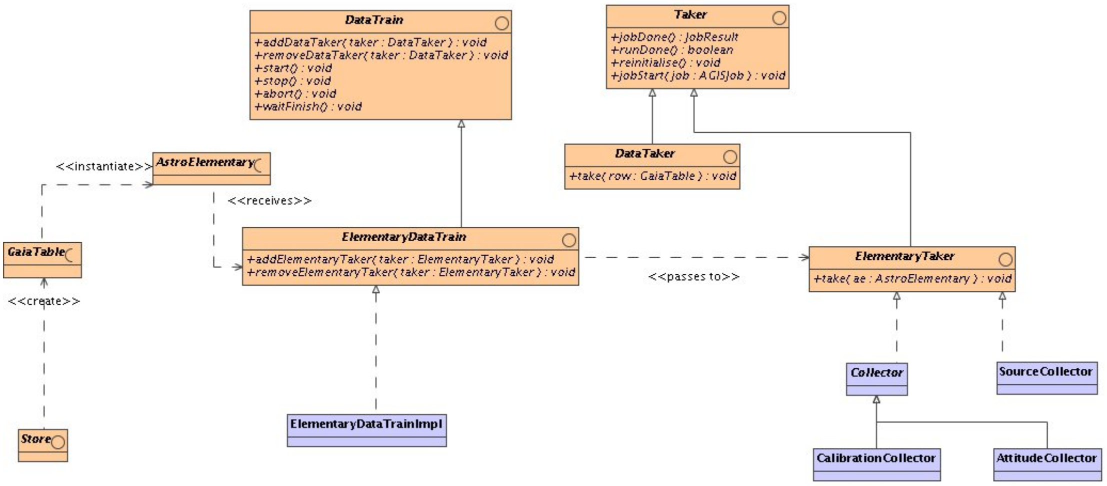


Figure 3.5: The *DataTrain* acts as a mediator between algorithms and data access (the *Store*) thus leading to a less coupled system. The *ElementaryDataTrain* accesses *AstroElementarys* in the fastest possible manner for the AGIS algorithms. The participating Algorithms must implement the *Taker* interface O'Mullane et al. (2006).

The generic notion of a *DataTrain* (see Fig. 3.5) is to access data in the fastest possible manner, usually meaning sequentially, and call a given set of algorithms passing them the data. The concept and code are quite simple. To enable the calling of the algorithms in a generic manner they must implement the *Taker* interface. As one may see in Fig. 3.5 the *Taker* is an interface with a method to "take" some data. By implementing this the algorithm has its input when the implementation is called.

More specifically for AGIS the *ElementaryDataTrain* accesses *AstroElementary* data (see Section 3.2.5). The train decides which data to access by taking a *Job* (see Section 3.3.2). It uses the *Store* to access a set of *AstroElementarys* each of which is then passed to each registered *ElementaryTaker* e.g the Source, Calibration, Global and Attitude algorithms. Each algorithm (see Section 3.4) must implement the *ElementaryTaker* interface to allow the *DataTrain* to interact with it. The *ElementaryDataTrain* has a method for registering the algorithms (see *addElementaryTaker* in Fig. 3.5). The algorithms must then accumulate observations until it may process a source or a time interval. An abstraction such as this forces the algorithms to accept data in the order it is stored allowing the infrastructure to be built without fixing the data storage order. Choosing spatial order Section 3.2.1.2 means all of the elementaries for a given source are sequential. Any given train accesses complete sets of elementaries wrt. to sources. The cartoon in Fig. 3.6 depicts how the *AstroElementary* is constructed by the *ObjectFactory* from a *GaiaTable* which results from a query to the Database through the *Store* interface. The *AstroElementary* is then passed to the algorithms attached to the *DataTrain*.

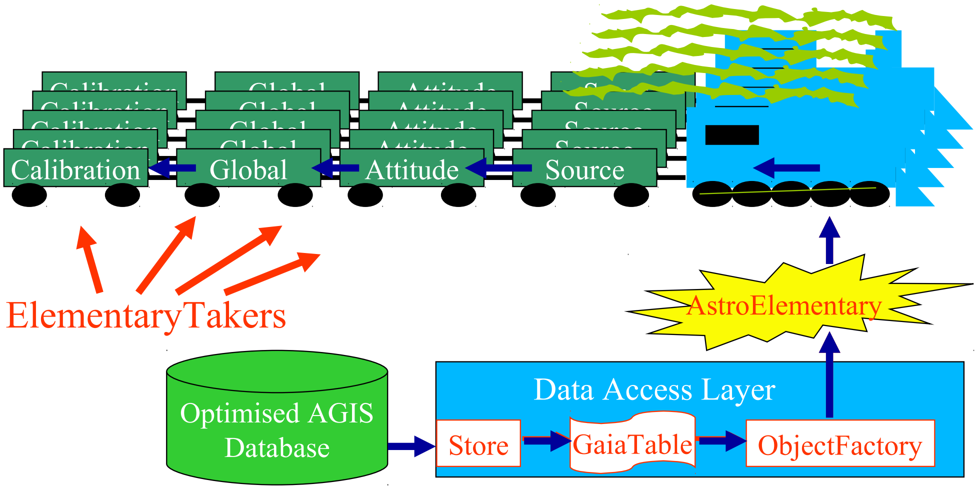


Figure 3.6: Here the blue arrow shows the flow of data from the Database through the Store and ObjectFactory to the algorithms attached to the ElementaryDataTrain. We may think of the ElementaryDataTrain as driving through the database passing observations to the algorithms. We may have as many trains in parallel as we wish, typically one per core or processor on the machine running AGIS.

3.2.3 Abstraction of Data Storage, the *Store*

To give a level of independence from the physical storage mechanism used it is normal to use some abstraction. Java interfaces provide an excellent approach to provide such insulation. Creation of an interface is a small coding overhead while in usage one gets a real implementation i.e. there is no overhead. It is very important to realise that a Java interface is a contract binding the using class and the providing class but does no translation of any kind.

No algorithm code in the system interacts directly with the DBMS (DataBase Management System) rather a query interface to the data is provided through the *Store* interface (see Fig. 3.7). The implementation of the Store is hidden behind the interface, thus the data store may be implemented in files or any database management system.

An implementation of the Store is requested from the *AGISFactory*, the actual implementation of the store is configured using the *gaia.tools.dal.Store* property in the `agis.properties` file and thus can be changed at run-time (rather than at compile-time). The store interface includes an explicit range query which returns all objects within a certain id range, which is required to support the *DataTrain*.

As depicted in Fig. 3.7 there are multiple implementations of the *Store*. The *FileStore* does not support the same level of querying as the *JDBCStore* but is sufficient for running the testbed on a laptop.

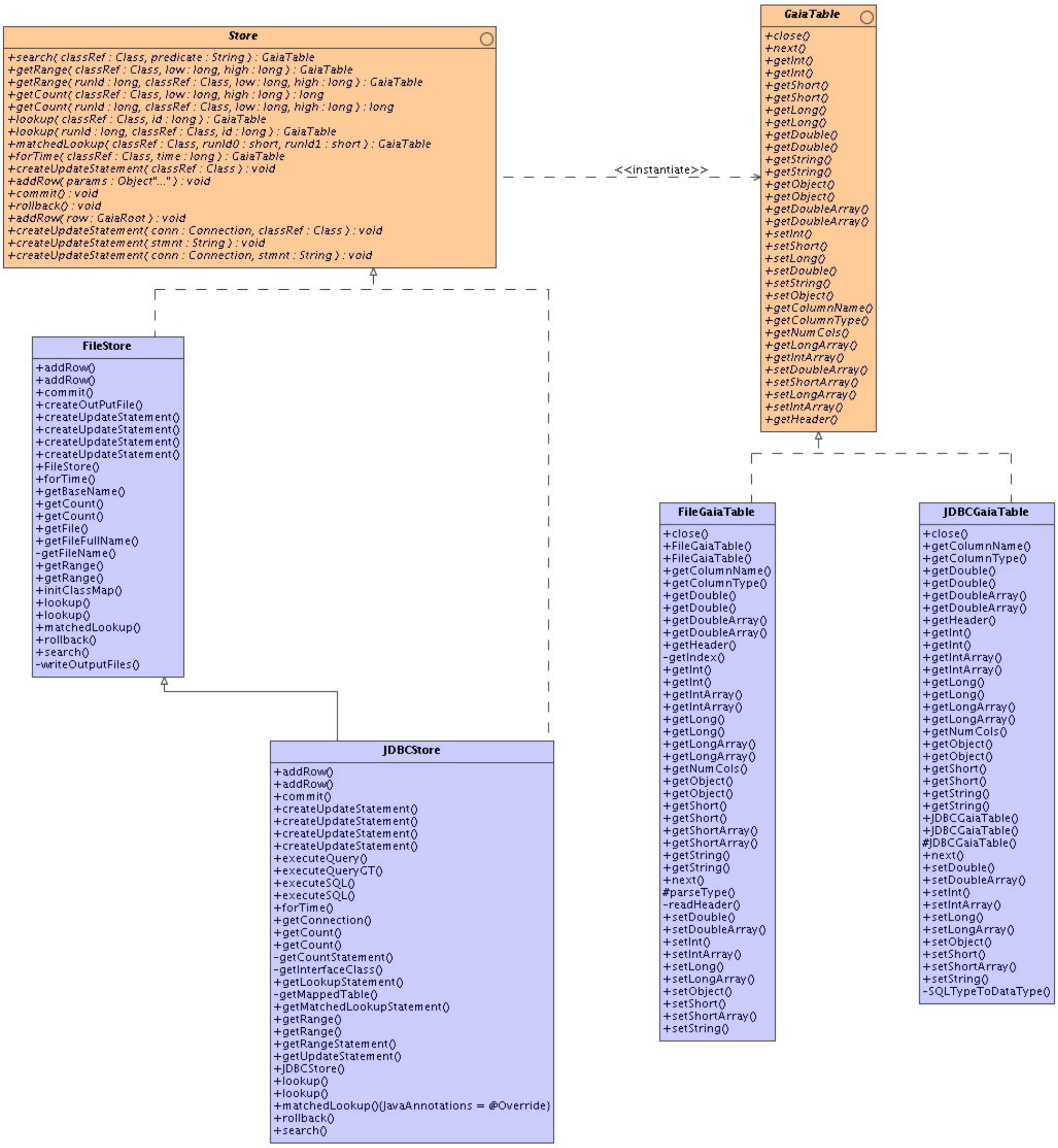


Figure 3.7: The *Store* provides an interface for data access, in this way many Store implementations may exist. In the figure we may see a *JDBCStore* and *FileStore* both of which implement *Store*. With these implementations AGIS code may be switched between FITS files or an Oracle Database for storage in a seamless manner.

GaiaTable (in Fig. 3.7) represents an interface to tabular data. The assumption of dealing only with tabular data is a major simplification for AGIS. This is a fair assumption since use of an object oriented database is unlikely. Both files (FITS, ASCII etc.) and Relational Database tables may be represented as a GaiaTable. The interface defines methods for retrieving the next row and for getting columns by name or index. The whole row may be passed to the Algorithm or *ObjectFactory* and it may extract the columns it requires. The *DataTrain* loads the entire row.

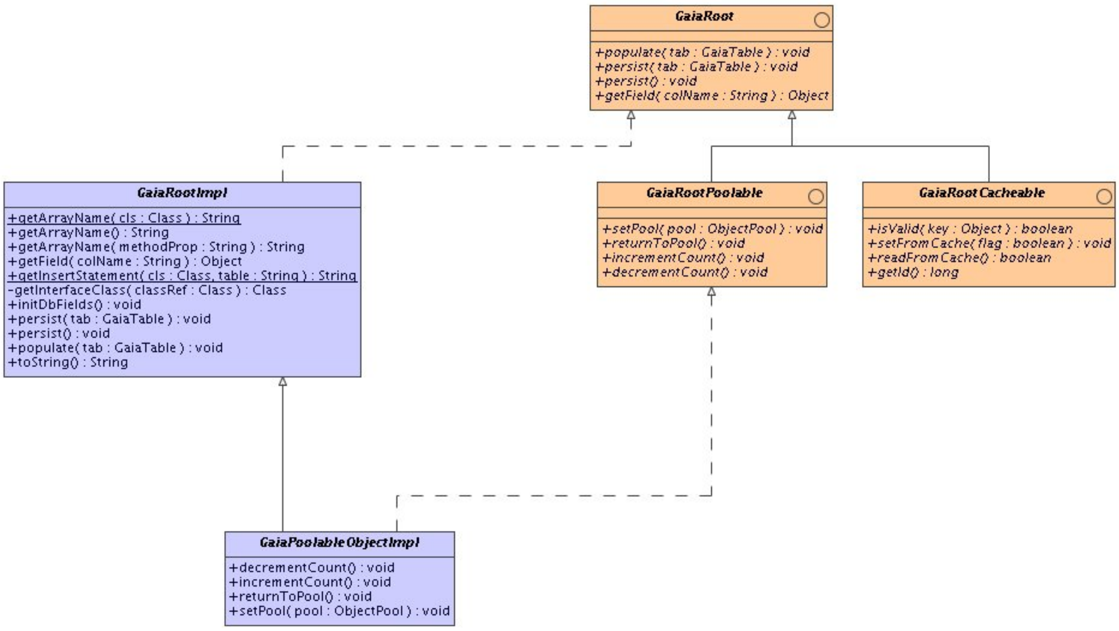

GaiaRoot
+populate(tab : GaiaTable) : void
+persist(tab : GaiaTable) : void
+persist() : void
+getField(colName : String) : Object
GaiaRootImpl
+getArrayName(cls : Class) : String
+getArrayName() : String
+getArrayName(methodProp : String) : String
+getField(colName : String) : Object
+getInsertStatement(cls : Class, table : String) : String
-getInterfaceClass(classRef : Class) : Class
+initDbFields() : void
+persist(tab : GaiaTable) : void
+persist() : void
+populate(tab : GaiaTable) : void
+toString() : String
GaiaRootPoolable
+setPool(pool : ObjectPool) : void
+returnToPool() : void
+incrementCount() : void
+decrementCount() : void
GaiaRootCacheable
+isValid(key : Object) : boolean
+setFromCache(flag : boolean) : void
+readFromCache() : boolean
+getId() : long
GaiaPoolableObjectImpl
+decrementCount() : void
+incrementCount() : void
+returnToPool() : void
+setPool(pool : ObjectPool) : void

Figure 3.8: All data objects implement *GaiaRoot* which makes certain methods available to the Store. All data objects are interfaces not real classes - this allows them to be easily replaced by different implementations.

The *GaiaRoot* UML is given in Fig. 3.8. If you are viewing this in colour, interfaces are brown (they also are marked with a ○) while implementations are blue. Any objects in the Gaia data model which use the *Store* (see also Fig. 3.7) and *ObjectFactory* must implement this interface. A basic implementation is provided which most classes may inherit from but in some cases, due to single inheritance in Java, this may not be possible.

3.2.4 Access to Objects - *ObjectFactory*

The store deals essentially with tables but some code will require objects. The object factory sits on top of the *Store* and returns objects implementing the data model interfaces. The object-from-table method of the interface is also exposed allowing code to do this conversion exactly when it requires it. We do need to take care that not too many pieces of code perform such a transformation - preferably it would be done once by the *DataTrain*. Splitting this out allows for very direct measurement of the performance.

This is implemented as a *Generic* class. The Factory is instantiated for a specific data model interface and then provides a method returning that class of object only. Java Generics are very nice for this and, although similar to C++ templates, should not be confused as being the same. Generics provide type checking and safety but they do not generate extra code with new types.

The factory relies on the *populate* method of the *GaiaRoot* to populate the fields of the object from a *GaiaTable*. A generic implementation of this using a mapping from the config file is provided in the *GaiaRootImpl* class. This provides a convenient mechanism to read the data from the Store into a Java Object that can be used elsewhere in the system.

The *ObjectFactory* also has caching capabilities. Whenever an Object is read from the *Store* it may be cached in memory in order to avoid new reads when it is requested again. Any object which is created by the Object Factory can be made cacheable just by implementing the *gaia.tools.dm.GaiaRootCacheable* interface. The caching can also be disabled by adding an entry to the property file. The interface contains a method to determine the “validity” of the Object.

The factory also has the possibility to implement object pooling. The notion here is to reuse objects by filling them with new data rather than reconstructing new ones. This technique was very popular in early Java implementations to reduce garbage collection time. Tests with the new JDK (1.5 and 1.6) show this is no longer beneficial. Still by having all data object creation done through one class the possibility to change the way it works later remains available.

3.2.5 The Data Model

As stated in Section 3.2.4 the algorithm’s work in terms of Java Objects such as *Source* and *AstroElementary*. These objects form what is generally termed a Data Model for the system. The AGIS Data Model is depicted in UML form in Fig. 3.9.

The data used for AGIS will be about 10% of the mission data; this will be selected and put in the special AGIS database (see for example Fig. 3.6).

Each of the interfaces in Fig. 3.9 are described below. The data model is made in terms of interfaces to allow easy substitution of multiple implementations. The *ObjectFactory* (see Section 3.2.4) and *Store* (see Section 3.2.3) are used to construct real implementations of these interfaces but all code refers only to the interface. Hence all client code may be compiled without any implementation if necessary. This is a technique used throughout AGIS and indeed GaiaTools.

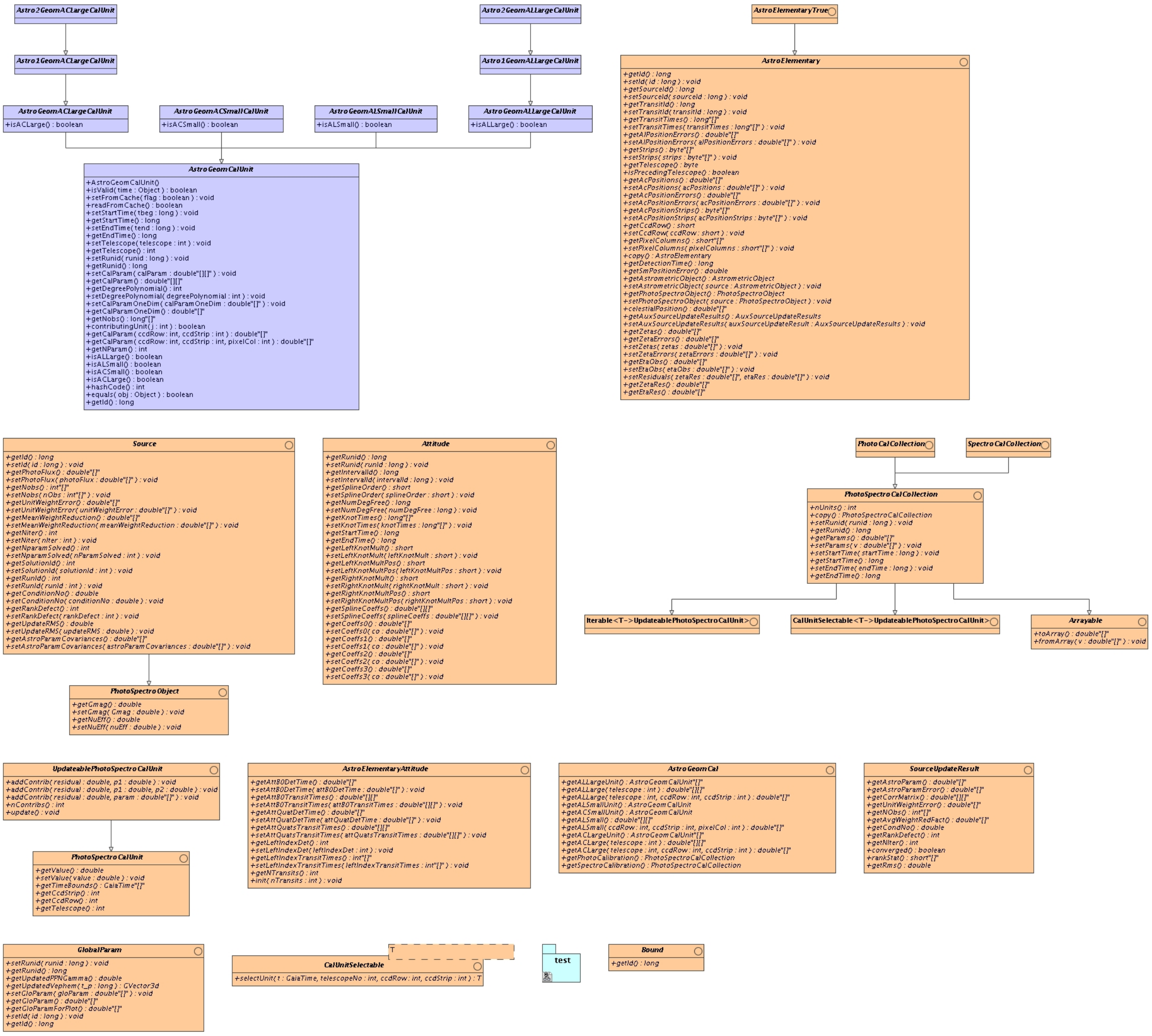

Astro2GeomACLargeCalUnit
Astro1GeomACLargeCalUnit
Astro2GeomALLargeCalUnit
Astro1GeomALLargeCalUnit
AstroGeomACLargeCalUnit
+isACLarge() : boolean
AstroGeomACSmallCalUnit
+isACSmall() : boolean
AstroGeomALSmallCalUnit
+isALSmall() : boolean
AstroGeomALLargeCalUnit
+isALLarge() : boolean
AstroGeomCalUnit
+AstroGeomCalUnit()
+isValid(time : Object) : boolean
+setFromCache(flag : boolean) : void
+readFromCache() : boolean
+setStartTime(tbeg : long) : void
+getStartTime() : long
+setEndTime(tend : long) : void
+getEndTime() : long
+setTelescope(telescope : int) : void
+getTelescope() : int
+setRunid(runid : long) : void
+getRunid() : long
+setCalParam(calParam : double"[][]") : void
+getCalParam() : double"[][]"
+getDegreePolynomial() : int
+setDegreePolynomial(degreePolynomial : int) : void
+setCalParamOneDim(calParamOneDim : double"[]") : void
+getCalParamOneDim() : double"[]"
+getNobs() : long"[]"
+contributingUnit(j : int) : boolean
+getCalParam(ccdRow : int, ccdStrip : int) : double"[]"
+getCalParam(ccdRow : int, ccdStrip : int, pixelCol : int) : double"[]"
+getNParam() : int
+isALLarge() : boolean
+isALSmall() : boolean
+isACSmall() : boolean
+isACLarge() : boolean
+hashCode() : int
+equals(obj : Object) : boolean
+getId() : long
AstroElementaryTrue
AstroElementary
+getId() : long
+setId(id : long) : void
+getSourceId() : long
+setSourceId(sourceId : long) : void
+getTransitId() : long
+setTransitId(transitId : long) : void
+getTransitTimes() : long"[]"
+setTransitTimes(transitTimes : long"[]") : void
+getAlPositionErrors() : double"[]"
+setAlPositionErrors(alPositionErrors : double"[]") : void
+getStrips() : byte"[]"
+setStrips(strips : byte"[]") : void
+getTelescope() : byte
+isPrecedingTelescope() : boolean
+getAcPositions() : double"[]"
+setAcPositions(acPositions : double"[]") : void
+getAcPositionErrors() : double"[]"
+setAcPositionErrors(acPositionErrors : double"[]") : void
+getAcPositionStrips() : byte"[]"
+setAcPositionStrips(acPositionStrips : byte"[]") : void
+getCcdRow() : short
+setCcdRow(ccdRow : short) : void
+getPixelColumns() : short"[]"
+setPixelColumns(pixelColumns : short"[]") : void
+copy() : AstroElementary
+getDetectionTime() : long
+getSmPositionError() : double
+getAstrometricObject() : AstrometricObject
+setAstrometricObject(source : AstrometricObject) : void
+getPhotoSpectroObject() : PhotoSpectroObject
+setPhotoSpectroObject(source : PhotoSpectroObject) : void
+celestialPosition() : double"[]"
+getAuxSourceUpdateResults() : AuxSourceUpdateResults
+setAuxSourceUpdateResults(auxSourceUpdateResult : AuxSourceUpdateResults) : void
+getZetas() : double"[]"
+getZetaErrors() : double"[]"
+setZetas(zetas : double"[]") : void
+setZetaErrors(zetaErrors : double"[]") : void
+getEtaObs() : double"[]"
+setEtaObs(etaObs : double"[]") : void
+setResiduals(zetaRes : double"[]", etaRes : double"[]") : void
+getZetaRes() : double"[]"
+getEtaRes() : double"[]"
Source
+getId() : long
+setId(id : long) : void
+getPhotoFlux() : double"[]"
+setPhotoFlux(photoFlux : double"[]") : void
+getNobs() : int"[]"
+setNobs(nObs : int"[]") : void
+getUnitWeightError() : double"[]"
+setUnitWeightError(unitWeightError : double"[]") : void
+getMeanWeightReduction() : double"[]"
+setMeanWeightReduction(meanWeightReduction : double"[]") : void
+getNiter() : int
+setNiter(nIter : int) : void
+getNparamSolved() : int
+setNparamSolved(nParamSolved : int) : void
+getSolutionId() : int
+setSolutionId(solutionId : int) : void
+getRunId() : int
+setRunId(runId : int) : void
+getConditionNo() : double
+setConditionNo(conditionNo : double) : void
+getRankDefect() : int
+setRankDefect(rankDefect : int) : void
+getUpdateRMS() : double
+setUpdateRMS(updateRMS : double) : void
+getAstroParamCovariances() : double"[]"
+setAstroParamCovariances(astroParamCovariances : double"[]") : void
Attitude
+getRunid() : long
+setRunid(runId : long) : void
+getIntervalId() : long
+setIntervalId(intervalId : long) : void
+getSplineOrder() : short
+setSplineOrder(splineOrder : short) : void
+getNumDegFree() : long
+setNumDegFree(numDegFree : long) : void
+getKnotTimes() : long"[]"
+setKnotTimes(knotTimes : long"[]") : void
+getStartTime() : long
+getEndTime() : long
+getLeftKnotMult() : short
+setLeftKnotMult(leftKnotMult : short) : void
+getLeftKnotMultPos() : short
+setLeftKnotMultPos(leftKnotMultPos : short) : void
+getRightKnotMult() : short
+setRightKnotMult(rightKnotMult : short) : void
+getRightKnotMultPos() : short
+setRightKnotMultPos(rightKnotMultPos : short) : void
+getSplineCoeffs() : double"[][]"
+setSplineCoeffs(splineCoeffs : double"[][]") : void
+getCoeffs0() : double"[]"
+setCoeffs0(co : double"[]") : void
+getCoeffs1() : double"[]"
+setCoeffs1(co : double"[]") : void
+getCoeffs2() : double"[]"
+setCoeffs2(co : double"[]") : void
+getCoeffs3() : double"[]"
+setCoeffs3(co : double"[]") : void
PhotoCalCollection
SpectroCalCollection
PhotoSpectroCalCollection
+nUnits() : int
+copy() : PhotoSpectroCalCollection
+setRunid(runid : long) : void
+getRunid() : long
+getParams() : double"[]"
+setParams(v : double"[]") : void
+setStartTime(startTime : long) : void
+getStartTime() : long
+setEndTime(endTime : long) : void
+getEndTime() : long
Iterable<T->UpdateablePhotoSpectroCalUnit>
CalUnitSelectable<T->UpdateablePhotoSpectroCalUnit>
Arrayable
+toArray() : double"[]"
+fromArray(v : double"[]") : void
PhotoSpectroObject
+getGmag() : double
+setGmag(Gmag : double) : void
+getNuEff() : double
+setNuEff(nuEff : double) : void
UpdateablePhotoSpectroCalUnit
+addContrib(residual : double, p1 : double) : void
+addContrib(residual : double, p1 : double, p2 : double) : void
+addContrib(residual : double, param : double"[]") : void
+nContribs() : int
+update() : void
PhotoSpectroCalUnit
+getValue() : double
+setValue(value : double) : void
+getTimeBounds() : GaiaTime"[]"
+getCcdStrip() : int
+getCcdRow() : int
+getTelescope() : int
AstroElementaryAttitude
+getAtt80DetTime() : double"[]"
+setAtt80DetTime(att80DetTime : double"[]") : void
+getAtt80TransitTimes() : double"[][]"
+setAtt80TransitTimes(att80TransitTimes : double"[][]") : void
+getAttQuatDetTime() : double"[]"
+setAttQuatDetTime(attQuatDetTime : double"[]") : void
+getAttQuatsTransitTimes() : double"[][]"
+setAttQuatsTransitTimes(attQuatsTransitTimes : double"[][]") : void
+getLeftIndexDet() : int
+setLeftIndexDet(leftIndexDet : int) : void
+getLeftIndexTransitTimes() : int"[]"
+setLeftIndexTransitTimes(leftIndexTransitTimes : int"[]") : void
+getNTransits() : int
+init(nTransits : int) : void
AstroGeomCal
+getALLargeUnit() : AstroGeomCalUnit"[]"
+getALLarge(telescope : int) : double"[][]"
+getALLarge(telescope : int, ccdRow : int, ccdStrip : int) : double"[]"
+getALSmallUnit() : AstroGeomCalUnit
+getACSmallUnit() : AstroGeomCalUnit
+getALSmall() : double"[][]"
+getALSmall(ccdRow : int, ccdStrip : int, pixelCol : int) : double"[]"
+getACLargeUnit() : AstroGeomCalUnit"[]"
+getACLarge(telescope : int) : double"[][]"
+getACLarge(telescope : int, ccdRow : int, ccdStrip : int) : double"[]"
+getPhotoCalibration() : PhotoSpectroCalCollection
+getSpectroCalibration() : PhotoSpectroCalCollection
SourceUpdateResult
+getAstroParam() : double"[]"
+getAstroParamError() : double"[]"
+getCorrMatrix() : double"[][]"
+getUnitWeightError() : double"[]"
+getNObs() : int"[]"
+getAvgWeightRedFact() : double"[]"
+getCondNo() : double
+getRankDefect() : int
+getNIter() : int
+converged() : boolean
+rankStat() : short"[]"
+getRms() : double
GlobalParam
+setRunid(runid : long) : void
+getRunid() : long
+getUpdatedPPNGamma() : double
+getUpdatedVephem(t_p : long) : GVector3d
+setGloParam(gloParam : double"[]") : void
+getGloParam() : double"[]"
+getGloParamForPlot() : double"[]"
+setId(id : long) : void
+getId() : long
T
CalUnitSelectable
+selectUnit(t : GaiaTime, telescopeNo : int, ccdRow : int, ccdStrip : int) : T
test
Bound
+getId() : long

Figure 3.9: The AGIS Data Model comprises several objects and is defined in terms of Java interfaces. The interfaces allow multiple implementations which is sometimes useful e.g. for testing purposes.

3.2.5.1 AstroElementary

An object of this kind represents the transits of a celestial source over the first 11 strips of the focal plane, namely, SM1/SM1 and AF1-9. Each AstroElementary in AGIS is associated with a single *Source*.

3.2.5.2 Source

An object of this kind represents a celestial sources that follows the standard 6-parameter astrometric model and are eligible for AGIS source processing.

3.2.5.3 Attitude

An object of this kind represents an interval of continuous attitude data. Attitude is parametrised as quaternions and each of the four components is fitted by a B-spline of a given order.

3.2.5.4 AstroGeomCalUnit

Calibrations are calculated in different time orders and for AC and AL (see Section 3.4.3). This class hierarchy represents all of these calibrations.

3.3 Distributed Processing

Regardless of the ordering chosen in Section 3.2.1 the access of the data does not need to be done serially. Indeed we require the data to be sequential on disk but multiple parts of that sequence may be read simultaneously. In the case of sources we may simultaneously process each source, in terms of distributed computing this is “Embarrassingly parallel” (Wilkinson & Allen, 1999) [1]. We may theoretically gain a speed up of N by using N processors where N is the number of sources to be processed. We say theoretically with reason as the data must still be read from disk and we are unlikely to actually put in place 10^8 processors. Still tests have shown that the number of processors added to the pool for AGIS indeed linearly decrease the amount of time spent processing. Some numbers are given in Table 3.1.

[1] http://en.wikipedia.org/wiki/Embarrassingly_parallel

Table 3.1: Evolution of AGIS performance during the development of the processing framework. Data volumes are indicated by the number of observations (*AstroElementaries*), depending on the number of sources and the length of the observation period. The time is the processing time per AGIS iteration for the given number of processors. The last column shows the throughput, in observations per processor per hour, as an indication of the real performance.

Date	Observations	Processors	Time (hr)	Normalized Rate
2005	1.6×10^7	12	3	0.9×10^6
2006	8.0×10^7	36	5	0.5×10^6
2007	8.0×10^7	24	3	1.3×10^6
2008	8.0×10^7	31	1	3.2×10^6
2009	2.6×10^8	50	1.8	2.8×10^6
2010	4.0×10^9	68	9.5	6.2×10^6

Concerning Table 3.1 one should also note that the 2006 test for 60 months took longer than expected i.e. the runtime did not remain the same. This required further optimisation - attitude appeared to be a bottleneck, in 2007 after several optimisations the runtime was reduced by a factor of 2 such that the system was running one outer iteration on 24 processors in 3 hours for 5 years data. In 2005 it took 3 hours on 12 processors to do just one third of the data volume i.e. 18 months of simulated data.

3.3.1 Distributed Processing Frameworks

Many different approaches exits for distributed processing. The different approaches are usually embodied in some library such as the Globus Toolkit [2]. However since we have an "Embarrassingly parallel" problem we also have little need for such a complex and heavy library. In fact all that we require is already available within the standard Java library namely:

- Communication Between Processing Nodes: The Remote Method Invocation (RMI) framework in Java provides this.
- Access to a database or databases: The Java Data Base Connectivity (JDBC) framework provides this.
- Some form of graphics library for GUIs: Java Swing library provides this.

Additionally, in this age of the web, Java provides easy support for dynamic web site generation using Java Server Pages (JSP).

[2] `http://www.globus.org/`

Hence an early feeling was to use the tools of Java directly rather than try to fit the problem into one of the many distributed programing libraries each with their own assumptions and problems. The modern programming languages of the day such as Java are very sophisticated in the feature set and tools they provide. Looking for example at JJPF (Danelutto & Dazzi, 2006) it provides some reliability on top of these tools while also taking a much more process oriented view - each worker has a `getData` call to pass back results. JJPF is also more coordinator oriented with a single server eliciting support from available nodes to perform a computation. In the grid world the obvious contender would be the Globus Toolkit (GTK) (Foster, 2006). Previous forays into GTK by the author showed the system to by buggy and difficult to use. GTK has improved dramatically over the years yet it still remains service oriented (our problem we believe to be data oriented) and has a large security overhead which we do not see as necessary. Indeed though (Demichev et al., 2003) is positive about GTK they introduce a resource broker which seems similar to our whiteboard Section 3.3.2. Unfortunately (Demichev et al., 2003) says little about the data intensive applications the title mentions.

The notion of just using the Java framework without some other layer was reinforced by previous experiences with the Sloan Digital Sky Survey (SDSS). On the SDSS a form of distributed query system was built using Web Services, the SQLServer database and the C# [3] language, this was called CasJobs (O'Mullane et al., 2005). This was done quite rapidly without using any special libraries beyond the facilities available in the programming language. Within the same group at Johns Hopkins a typical Grid application for Cluster finding within a catalogue was taken and quickly rewritten in C#, this ran about ten times faster using a database system. This experiment was reported in (Nieto-Santisteban et al., 2005).

A final justification, perhaps the ultimate and obvious one, for not taking on a library is that of simplicity. Returning to the quote from Einstein at the beginning of this chapter it was believed the distributed computing libraries would not make the system simpler hence none were adopted.

3.3.2 A whiteboard for job distribution

There are at least two main approaches to controlling a grid of distributed processes, the first to have "Agents" register with some central controller which then regulates the entire process, the second to have a less centralised approach with more autonomous processes.

The central controller approach is very appealing and generally the way many agent based systems work. Generally these involve monitoring resources and farming out jobs to particular processors which are not fully loaded. The central registering of agents means the controller knows how many agents of which types exist on the system and further more may reject agents from particular machines or of particular types. Such systems deal well with uneven workloads and ad hoc jobs by many users. Often security layers and user tracking are included.

[3] The # here is the musical sharp hence this is pronounced C sharp

In contrast an AGIS iteration could easily occupy an entire cluster for some days. There are no ad hoc programs only the entire AGIS chain running on all data. There are no users, hence no particular need for a security overhead in terms of certificates etc.

In our data driven approach Section 3.2.2 we may consider the data as the distribution mechanism. Everything hinges on the processing of some block of data be it a time sequence Section 3.2.1.1 or a set of observations Section 3.2.1.2. All we really need to know is if a particular part of the data set has been visited during a particular iteration. If the data segments are chosen properly we may have as many `DataTrains` running as we wish (one per source being the limit as pointed out above). This is very simply and easily achieved through a whiteboard mechanism as depicted in Fig. 3.10.

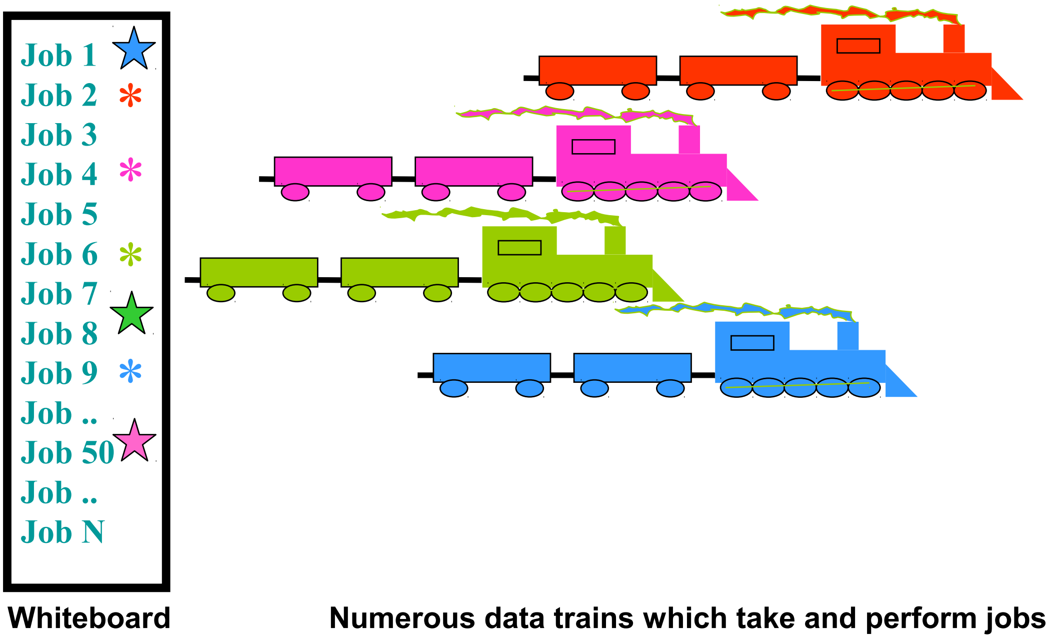
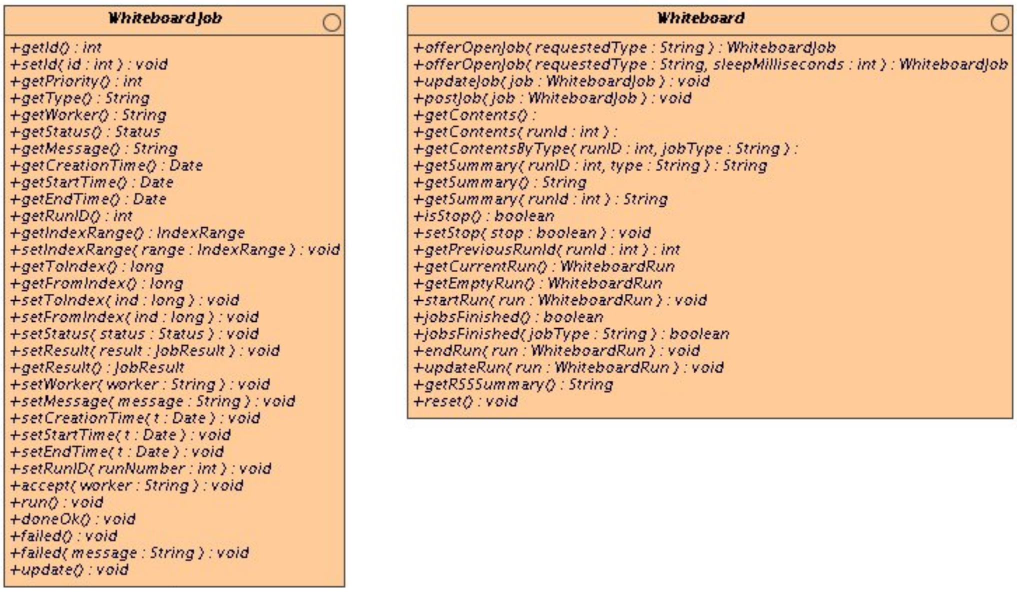


Figure 3.10: A set of Jobs corresponding to sequential sequences of data which cover the entire data range may be posted on a whiteboard. The `DataTrain` may mark a job as in progress when it starts it and complete when it is finished. When all jobs are done all of the data has been seen once. The whiteboard itself has no special knowledge of the jobs or the overall task - it is a simple mechanism to coordinate potentially hundreds of processes.

The whiteboard is quite a simple concept for organising many processes of varying types. Conceptually we may "post" jobs on a whiteboard and workers, in our case "DataTrains", may pick them up. The whiteboard may hold status information e.g. when a job started, when it ended, was it all OK etc. In effect then the whiteboard becomes the central controller however it exercises no control as such. Perhaps the original of the species in this respect is the OPUS pipeline from the Space Telescope Science Institute (Rose et al., 1995). Indeed it is the OPUS blackboard [4] design pattern which is employed here. It is noted that since its early beginning OPUS is itself going toward

[4] Whiteboard was elected as a more modern alternative to Blackboard.

Java (Miller et al., 2003) but maintaining its heterogeneity through CORBA (Common Object Request Broker Architecture). For the purposes of AGIS which is a pure Java implementation a simple white board was coded directly in Java using a Database table to hold the jobs. The latter providing also the ability to ensure no two trains ever get the same job. The JDBC framework in Java makes the whiteboard seamlessly accessible from any node on the network - hence no need for the overhead of CORBA or some other message passing system here. The UML (Unified Modeling Language) interface for the whiteboard is shown in Fig. 3.11.



```
WhiteboardJob
+getId() : int
+setId( id : int ) : void
+getPriority() : int
+getType() : String
+getWorker() : String
+getStatus() : Status
+getMessage() : String
+getCreationTime() : Date
+getStartTime() : Date
+getEndTime() : Date
+getRunID() : int
+getIndexRange() : IndexRange
+setIndexRange( range : IndexRange ) : void
+getToIndex() : long
+getFromIndex() : long
+setToIndex( ind : long ) : void
+setFromIndex( ind : long ) : void
+setStatus( status : Status ) : void
+setResult( result : JobResult ) : void
+getResult() : JobResult
+setWorker( worker : String ) : void
+setMessage( message : String ) : void
+setCreationTime( t : Date ) : void
+setStartTime( t : Date ) : void
+setEndTime( t : Date ) : void
+setRunID( runNumber : int ) : void
+accept( worker : String ) : void
+run() : void
+doneOk() : void
+failed() : void
+failed( message : String ) : void
+update() : void
```

```
Whiteboard
+offerOpenJob( requestedType : String ) : WhiteboardJob
+offerOpenJob( requestedType : String, sleepMilliseconds : int ) : WhiteboardJob
+updateJob( job : WhiteboardJob ) : void
+postJob( job : WhiteboardJob ) : void
+getContents() :
+getContents( runId : int ) :
+getContentsByType( runID : int, jobType : String ) :
+getSummary( runID : int, type : String ) : String
+getSummary() : String
+getSummary( runId : int ) : String
+isStop() : boolean
+setStop( stop : boolean ) : void
+getPreviousRunId( runId : int ) : int
+getCurrentRun() : WhiteboardRun
+getEmptyRun() : WhiteboardRun
+startRun( run : WhiteboardRun ) : void
+jobsFinished() : boolean
+jobsFinished( jobType : String ) : boolean
+endRun( run : WhiteboardRun ) : void
+updateRun( run : WhiteboardRun ) : void
+getRSSSummary() : String
+reset() : void
```

Figure 3.11: The UML interfaces for the `Whiteboard` and the `WhiteboardJob`. Note the "postJob" used to populate the whiteboard and "offerOpenJob" methods which the DataTrains use to get jobs. The job itself has methods for status and messages etc.

Regardless of the jobs being done the whiteboard can give some information on the general state of the system. A series of JSP pages present the whiteboard state on a website. On this site, with little effort, we may show jobs completed/remaining and (assuming uniform jobs) an estimate for the end time. We may also list statistics per processor simply by querying the job table in the database.

3.3.3 Overall control *RunManager* and *ConvergenceMonitor*

The `Whiteboard` alone is not enough to converge an AGIS solution. Some other entity must post the jobs on the board for the *DataTrains* to work on. The *RunManager* has the task of coordinating iterations and the publishing of jobs as depicted in Fig. 3.12. The *RunManager* uses the *JobPublisher* to publish appropriate jobs e.g. one for each block of elementaries to be processed. The *JobPublisher* scans a table of bounds (a list of identifiers of elementaries which are the last in a series belonging to a single source) and creates a number of jobs based on blocks of elementaries. In general the system is configured such that these jobs complete in a few minutes as this gives a better

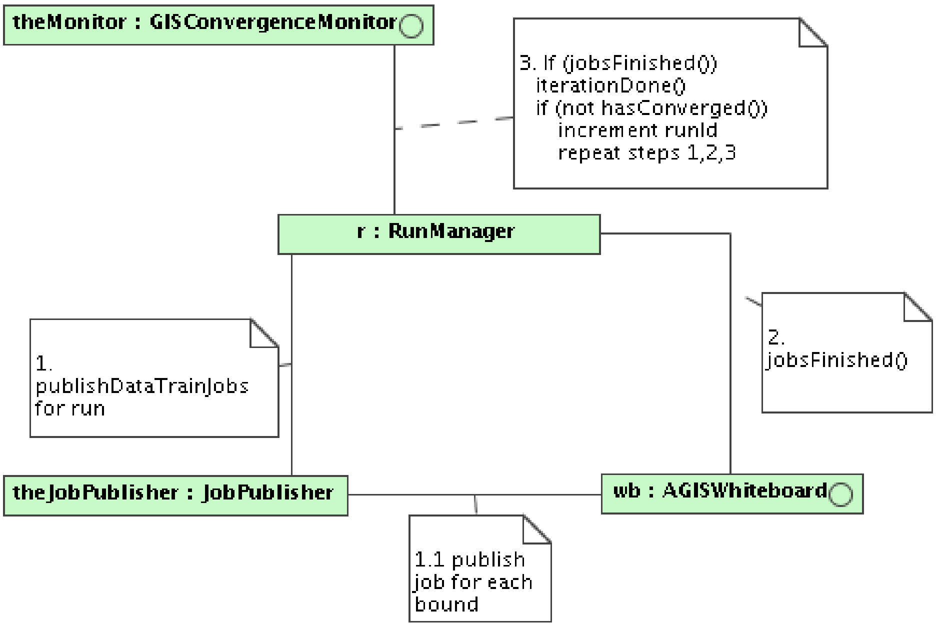


Figure 3.12: Communication diagram for the `RunManager`. This summarises the `RunManagers` role in publishing jobs and checking for convergence.

indication of progress and failed jobs are detected in a timely manner for resubmission. Hence there are typically thousands of jobs in a single run. Once posted the trains pick them up and start working. The order in which the jobs are done does not matter and indeed they may all be executed in parallel as outlined above (see Section 3.3). Jobs are also published for the Calibration, Attitude and Global if these algorithms are attached to the train. These jobs execute for the entire iteration.

The *RunManager* then periodically checks to see if the datatrain jobs have finished. If they are done the main part of the iteration is done and the *GisConvergenceMonitor* is told the iteration is at its end. The *RunManager* then asks the *GisConvergenceMonitor* if the solution has converged and awaits the answer. At this point the attitude, global and calibration servers still must perform their final calculations - when these are complete the *GisConvergenceMonitor* replies as to the state of convergence. The normal convergence criteria being that the median and width of the parallax update histogram is $1\mu as$ or less which is configurable in the property file.

If convergence has not been reached the *RunManager* starts another run through the data by publishing a new set of jobs. If it has converged the *RunManager* declares the cycle ended and converged.

3.4 Algorithms

There are effectively two types of algorithms in the system: those with a centralised part and those which are completely distributable. Let us first look at the “source update” algorithm which is completely distributed and afterwards at the others.

3.4.1 Source Update

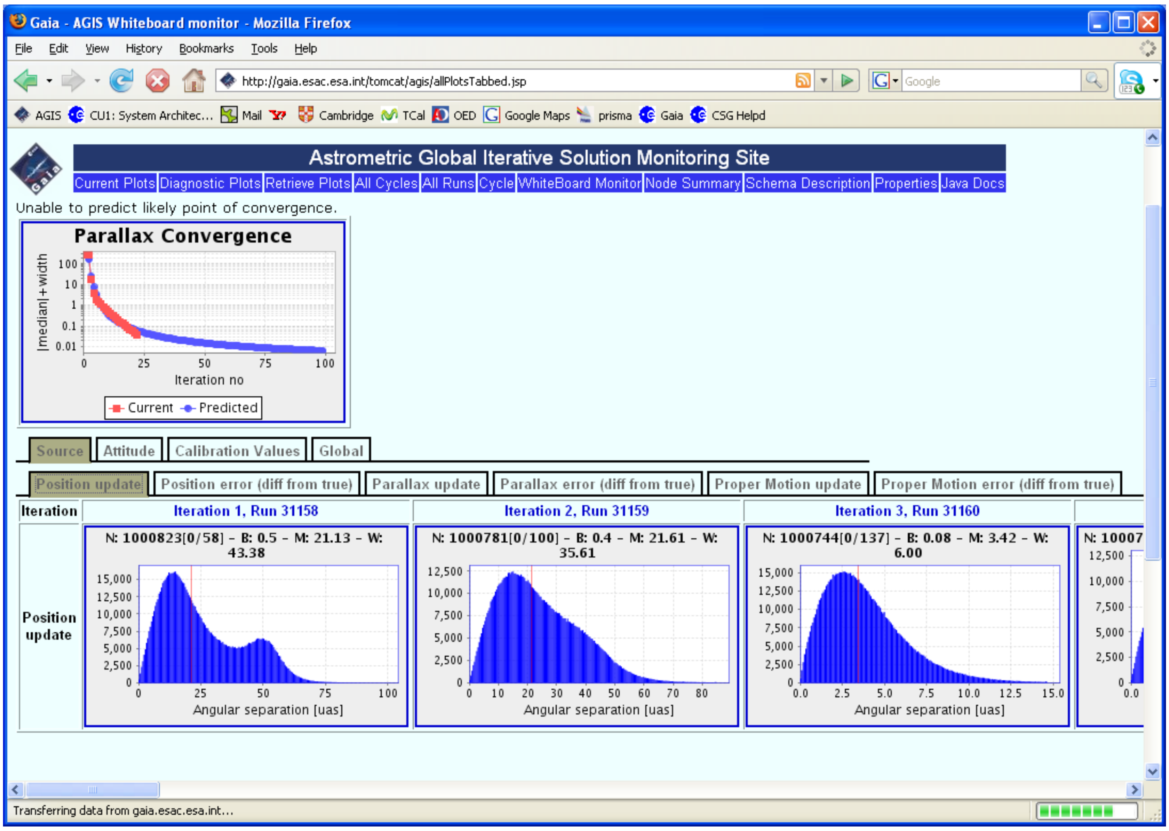
The mathematical formulation of the source update as a least squares fit of the observations was presented in Section 2.4. The source update step is truly distributed. As the DataTrain passes elementaries to the *SourceCollector* (the *Taker* registered with the train for Sources) it accumulates all of the elementaries for a given source. Remember that the elementaries are stored in such a manner that all elementaries for one source are consecutive hence the collector knows when it has all of the elementaries for a given source once the sourceId changes. Once it has a batch of elementaries the source update is called. This is the least squares fit of observations to the source model as described in Section 2.4.1. It is an iterative process itself typically taking three or four iterations. When the new astrometric parameters are available they are passed to the *SourceUpdateManager*, this batches together several sources for efficient storage. Nothing in AGIS is ever actually updated rather a new source row is written to the table with the current runID. In this way a complete history of the updates is preserved. Inserting to the Database is also more efficient than updating.

In fact the *SourceUpdateManager* does not write the sources until the entire job is done - effectively a database transaction is held open until the job finishes. In this manner a job is either completed or not since the transaction may be ’rolled back’ without consequence if there is some problem.

Also when the job is finished the *SourceCollector* sends all of the updated sources to the *GisConvergenceMonitor*. This call is made using RMI. Because the *GisConvergenceMonitor* receives sources throughout the iteration, update histograms can be dynamically generated. These are displayed on the associated AGIS website in real time. The website is shown in Fig. 3.13.

3.4.2 Attitude Update

The continuous attitude model is explained in Section 2.5. In fact the attitude may be divided into segments each of which may be solved simultaneously. There will be natural breaks in Gaia’s attitude in any case but this technique may be used to distribute attitude further. Hence depending on the number of attitude segments there is a limit to the distribution of attitude processing. Each segment may be solved on an individual processor. In actual fact the final fitting of the attitude for five years data as a single spline with knots every fifteen seconds took only 30 minutes on a Xeon processor with



Unable to predict likely point of convergence.

Transferring data from gaia.esac.esa.int...

Gaia - AGIS Whiteboard monitor - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://gaia.esac.esa.int/tomcat/agis/allPlotsTabbed.jsp
Google
AGIS CU1: System Architec... Mail Cambridge TCal OED Google Maps prisma Gaia CSG Helpd
Astrometric Global Iterative Solution Monitoring Site
Current Plots Diagnostic Plots Retrieve Plots All Cycles All Runs Cycle WhiteBoard Monitor Node Summary Schema Description Properties Java Docs
Parallax Convergence
|median|+width
100
10
1
0.1
0.01
0 25 50 75 100
Iteration no
Current Predicted
Source Attitude Calibration Values Global
Position update Position error (diff from true) Parallax update Parallax error (diff from true) Proper Motion update Proper Motion error (diff from true)
Iteration
Iteration 1, Run 31158
Iteration 2, Run 31159
Iteration 3, Run 31160
Position update
N: 1000823[0/58] – B: 0.5 – M: 21.13 – W: 43.38
15,000 12,500 10,000 7,500 5,000 2,500 0
0 25 50 75 100
Angular separation [uas]
N: 1000781[0/100] – B: 0.4 – M: 21.61 – W: 35.61
12,500 10,000 7,500 5,000 2,500 0
0 10 20 30 40 50 60 70 80
Angular separation [uas]
N: 1000744[0/137] – B: 0.08 – M: 3.42 – W: 6.00
15,000 12,500 10,000 7,500 5,000 2,500 0
0.0 2.5 5.0 7.5 10.0 12.5 15.0
Angular separation [uas]
N: 10007
12,500 10,000 7,500 5,000 2,500 0
0.0

Figure 3.13: Update plots such as for the Source position update shown here are generated dynamically and displayed on the AGIS monitoring website while the system is running. Historical plots may also be retrieved. Note also in this screen grab the overall convergence is depicted in a single plot on top of the page. Other plots are grouped in easy to use tabs.

sixteen Gigabytes of memory. The solution is not the bottleneck, it is rather the gathering of the observations. With a single attitude update server every source's observations must be passed to this server from every DataTrain. Once the system surpassed thirty two DataTrains this became a limiting factor.

On each DataTrain an *AttitudeCollector* is registered. This gathers all of the elementaries and passes them to the appropriate *AttitudeUpdateServer*. Appropriate here means the attitude server dealing with the time bin in which the observation falls. In some cases the segments overlap and an observation may be sent to two servers simultaneously. Again RMI is used for this passing and the observations carry the updated source parameters with them.

The *AttitudeUpdateServer*(s) adds to the partial equations for each observation passed. It must wait until the end of the run to ensure all observations have been seen before doing the final computation. The end of the run is signalled, via RMI, by the *RunManager*. At this point the new spline coefficients are calculated and written to the *Store*. The server then sends the updated attitude to the *ConvergenceMonitor* so it may be plotted on the website.

3.4.3 Calibration Update

The geometric calibration model is described in Section 2.6.1. From the perspective of distribution each time scale for calibration could be calculated on separate machines. However one must consider that, unlike attitude, here an observation will end up going to many calibration bins e.g. at least one short term and one long term calibration will need to take account of each observation. Hence distribution is not as trivial as it might seem. On the positive side, the processing for calibration is not a huge overhead, as for attitude, the main bottleneck is the sending of all observations to the calibration server. Unlike the attitude server the calibration server can process the incoming observations more quickly as such it has not been an overall bottleneck in the system and remains to date running on a single server.

The framework is similar to attitude. A *CalibrationCollector* is registered with each *DataTrain* and collects the required observation information and sends it to the *CalibrationUpdateServer* via RMI. The server accumulates the equations during the run and performs the final calculation when signalled by the *RunManger* that the run is complete. It writes the updated calibrations to the *Store* and sends them to the *GisConvergenceMonitor* for plotting on the website.

3.5 Conclusion

The overall AGIS architecture and many of the components have been described in some detail. This is a software system designed and optimised to perform the Gaia Astrometric data reduction. Advanced features of the Java language have been employed to make this system work well and remain very portable. Further work is needed in the coming years but a very good system is already in place and well understood. As described in Section 6 a solid team with a dynamic working method has built the system so far and will continue to refine and test it up to launch and beyond.

Chapter 4

Astrophysical considerations

> Die Astronomie ist vielleicht diejenige Wissenschaft, worin das wenigste durch Zufall entdeckt worden ist, wo der menschliche Verstand in seiner ganzen größe erscheint, und wo der Mensch am besten kennen lernen kann, wie klein er ist. [1]
>
> **Georg Christoph Lichtenberg (1742-1799) physicist, philosopher**

This chapter takes us through some of the astrophysical and physical phenomena which AGIS must account for. Chapter 5 explains some of the experimental results which give confidence that AGIS does indeed correctly deal with these phenomena. What is presented here is of course a subset of the phenomena. Indeed, echoing Lichtenberg above, the range of effects is belittling and it will surely take several volumes in the final catalogue to describe them all. AGIS itself only deals with a subset of all known phenomena which have effects in the μas realm. Many of these topics are well covered in (Lindegren, 2005) which describes the astrometric principles of Gaia. Indeed the Paris symposium (Turon et al., 2005) was a terrific forum for all parts of astrophysics touched by Gaia.

4.1 Number of parameters in the astrometric solution

It is not clear that the inclusion of the sixth astrometric parameter μ_r increases the effectiveness of the solution. In (GAIA-LL-055) Lindegren points out that not all six parameters need necessarily be updated in the solution. In most cases the five parameter solution is sufficient and indeed in some cases even fewer parameters may be used e.g. if there are too few observations it may only be possible to derive α,δ,ϖ. The conclusion is that the Source update algorithm should be parameterised to allow for solving from two (α δ) up to all six astrometric parameters. This is how the current implementation works.

[1] Astronomy is perhaps the science in which the least is discovered by chance, in which the full breadth of human understanding shines through, and in which man can best learn how small he is.

4.2 The basic angle

For an astrometric measurement some standard or scale is needed. Often this is a relative scaling within a field of view e.g. a plate scale for a photographic plate, based on known characteristics of the telescope. Already μas measurements have been made from ground using radio astronomy e.g. for some pulsars (Vlemmings et al., 2005). Fomalont describes the absolute and relative astrometric techniques for radio astrometry in (J.A. Zensus, 1995, Chapter 19). In general these techniques involve such as tropospheric refraction, ionospheric phase delays etc. Still if well modeled and with sufficient observations radio astrometry can yield good positions.

Classical optical astronomical parallax measurements are always relative to some background sources in the same field of view; since in principle all sources in the field of view will have some parallax shift, determination of the precise parallax of any single source is impossible. For these we use the term *relative* parallaxes. Gaia will provide *absolute* parallaxes.

For Gaia astrometry two telescopes observe, or connect, two distinct fields of view simultaneously. The telescopes are separated by 106.5° which is known as the basic angle and usually denoted γ. By essentially measuring parallax against many different and widely separated backgrounds a true parallax may be derived. It has been shown mathematically that this Hipparcos/Gaia approach provides absolute parallaxes (Makarov, 1998) as long as the basic angle is between 30° and 150°. A detailed discussion, from the Hipparcos viewpoint, may be found in (ESA, 1997, Volume 3 Chapter 20) where comparisons are made to other parallax measurements.

As the known positions of the observed sources improve the position of the satellite may also be improved. Observing the sources of one FOV relative to a completely different part of the sky at a later time would allow one to detect a variation in the basic angle. Variations on a short time scale in phase with respect to the direction to the sun, however will be indistinguishable from a global parallax shift (GAIA-LL-057) due to which great pains have been taken to ensure its stability. Long period variations in the basic angle will be solved for in AGIS.

4.3 The scanning law

The Earth Sun Lagrangian point (L2) has been chosen for Gaia. L2 is one of the five libation points in the earth sun system where a small body, such as Gaia, could theoretically remain stationary relative to the two larger bodies. L2 is an excellent choice for Gaia as the sun and earth are both on the same side of the satellite allowing the antennae and solar panels to face them in one direction while the instruments face the unobstructed coolness of space in the opposite direction. Of course Gaia could only theoretically remain stationary at L2; in practice it will need regular thruster firings to maintain its orbit.

As a survey mission Gaia should try to cover as much of the sky as often as possible over its five year life. An even distribution of observation time would be ideal. There are of course several constraints on Gaia's observing pattern. For example:

- minimize fuel usage e.g. not too many thruster firings.
- keep Sun shield pointing toward the Sun and antennae pointing toward Earth.
- maximise the solar aspect angle (ξ); the greater ξ the greater the parallactic displacement of the observations, however thermal stability and power requirements limit ξ to near 45°.
- maximise the uniformity of sky coverage over five years; repeated observations should be well spaced out over the five years (e.g. for periodicity and proper motion having 100 observations of a source in one year is not nearly as good as 20 each year over five years).

It is possible to describe how Gaia observes the sky over the five year period using a formula which is generally referred to as the 'scanning law'. This determines the rotation in three-axis of the satellite at all times. Hipparcos also employed such a revolving scanning law, originally proposed by Erik Høg in 1976, as noted and recapped by Lindegren in (SAG-LL-014).

Gaia rotates around its spin (z) axis (see Fig. 4.1) at a constant speed of 60 arcsec s^{-1} while the axis precesses slowly on the sky at a fixed solar aspect angle (ξ) of 45°. The precession speed is such that during the five years of operation the spin axis will perform twenty nine revolutions around the solar direction, giving a precession period of sixty three days. On average each source on the sky is observed astrometrically seventy times and spectrographically forty times (assuming 20% deadtime, FM-030).

4.4 The number of sources needed for AGIS

The number of sources required for the Astrometric Global Iterative Solution is driven by the calibration needs as discussed in Section 4.8. On the other hand not all sources are suitable for AGIS and indeed having a very uneven distribution of sources is also not very good (see Section 4.8.7)[2].

Let us consider 10^6 sources observed over five years according to the nominal scanning law. If these are evenly distributed then we may consider the fraction of sky seen in any time period as an approximation of the number of sources. Gaia scans a complete swathe of $0.7deg$ in 6 hours or $360 \times 0.7 = 252deg^2$ in a day then Gaia covers $1008deg^2$ (there is overlap but source are still seen even if some of them are the same sources multiple times). So in a day this is about $1008/41253 = 2.44\%$ of the sources or ~ 24000.

[2] The Conjugate Gradient Solver seems to relax this demand

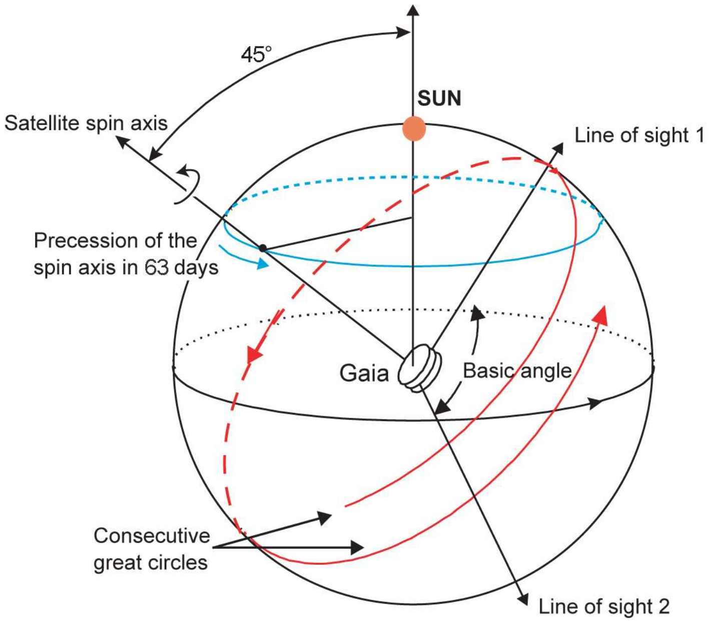


Figure 4.1: Gaia's two fields-of-view scan the sky according to a carefully prescribed, 'revolving scanning law'. The constant spin rate of 60 arcsec s^{-1} corresponds to 6-hour great-circle scans. The angle between the slowly precessing spin axis and the Sun is maintained at 45°. The basic angle is 106.5°. Figure Karen O'Flaherty.

This is fine for large scale calibrations but if we take this down to the pixel column level this must be further divided by 1966 giving only about 12 sources per pixel column. This is insufficient for small scale calibration on the daily scale but perhaps would work on the monthly scale. To make daily small scale calibrations we would probably want about 100 transits per pixel column per day. By reversing the above this would mean at least 10^7 sources in AGIS. To allow some margin AGIS is required to handle 10^8 sources.

Furthermore, as discussed in Section 4.8.1, 10^8 sources seem reasonable to have an appropriate number of sources per magnitude and colour bin across the sky.

One should not forget that it is possible to run AGIS with fewer sources and disable short term small scale calibrations. In this manner one may improve source positions, large scale calibrations and attitude, after which a secondary update may be run. Using the improved sources from the secondary update run one could then run AGIS with considerably more sources to fit the small scale calibrations correctly - this of course would then require another secondary source update.

4.5 Chromaticity

Due to the optical properties of the instrument the centroid of a monochromatic source image is dependant on the wavelength of light from the source. For a polychromatic image the dependency is on the spectral energy distribution (SED) (GAIA-LL-053).

Hence within our astrometric solution we may expect to see small shifts depending on the spectral type and colours of the source.

4.6 Charge Transfer Inefficiency(CTI)

CTI effects are caused by charge traps in the silicon holding electrons and releasing them later (SAG-LL-025), (AS-009). This distorts the PSF centroid, width and height. CTI will become worse as the CCDs are exposed to radiation at L2, resulting in large apparent shifts in position. Most of these effects will be accounted for in IDT and IDU processing. AGIS may provide small deltas to the CTI model parameters but this is not confirmed.

4.7 General Relativity

Gravitational light bending will be a perceivable effect for Gaia observing in the μas domain. Light bending of observations due to the gravitational field of the solar system needs to be accounted for in all Gaia processing not just AGIS. Klioner suggests a model based on so-called Parametrized Post-Newtonian (PPN) formalism for μas astrometry in general and Gaia in particular. Work in this area continues, (Klioner, 2001) being the first postulation of the model which was refined in (Klioner, 2003). The meaning of relativistic astrometry and this model were discussed in the 2004 Paris symposium (Klioner, 2005) as well as its refinement for Gaia (Klioner & Soffel, 2005). This later refinement offered a few calculation savings for Gaia in effects which are sub μas.

In addition to using this model in all calculations AGIS may use the observations to calculate selected PPN parameters, mainly β and γ which could show possible deviations from General relativity where they should both equal unity. Calculating such parameters provides an empirical check of general relativity.

4.7.1 Aberration of light

Already explained in 1727 (Bradley, 1727)[3] due to the motion of the observer and the finite speed of light the apparent position of a source varies from its true position. This effect can be several 10s of arcseconds and as such must be accounted for in Gaia observations. This means that knowledge of Gaia's velocity must be taken into account when calculating source positions. This effect is dealt with within the relativistic model for Gaia observations. It is also described in (Klioner, 2001).

[3] Although the citation was found the article was not found and subsequently not read

4.8 Photon density and relegation

For the DPAC reader this section is a modified version of the relegation note (WOM-014). The current AGIS implementation uses simulations with a fairly uniform sky density of primary sources (Fig. 4.2) - the results discussed in Section 5 use this distribution. The actual star density on the sky is extremely non-uniform. Assuming that an approximately constant fraction of all the stars can in principle be used as primary stars, it follows that the astrometric weight (being roughly proportional to the photon flux per unit area on the sky) is also extremely non-uniform. This is not a problem for the source and calibration updates, but it is for the attitude update – and therefore, indirectly, also for the other updates through the iterative nature of the AGIS. The cause of the problem is the following: when the two fields of view (FOV) look at regions with very different weight (photon) densities, the attitude tends to be almost completely determined by the observations in the FOV with the higher density. This means that the astrometric errors in high-density areas of the sky have little chance of being eliminated by direct linking of the stars to other parts of the sky via the superposed fields. In other words, the connectivity between the two FOVs is effectively lost, and therefore also the ability to build a truly global system of astrometric data. The result may be strongly correlated containing (quasi-systematic) errors within small regions of the sky. The importance of this phenomenon was discovered by F. van Leeuwen during his AGIS-like re-reduction of the Hipparcos data (van Leeuwen & Fantino, 2005).

For the calibrations there are other requirements on the primary sources, in particular that there are enough of them at various magnitudes and colours, while their sky distribution is less important. Securing a sufficient number of primary sources for the calibrations will tend to maximize the total number of primary sources in some areas e.g. the galactic plane, resulting in very non-uniform weight distribution.

Thus we have somewhat conflicting requirements for the attitude (uniform weight) and calibration (good distribution in magnitude and colour). It is proposed to handle this by (1) selecting primary sources suitable for the calibrations, and (2) manipulate the weights of the observations in the attitude updating to balance the fields of view. In practice the second step means downweighting the data in the high-density field.

Both the selection of primary sources and the downweighting scheme based on the relative photon densities in each Field Of View (FOV) are described below. The term ‘relegation’ for demoting a source to secondary was suggested by U. Bastian during AGIS discussions in 2006.

In the pseudo code used below, a selection from `Source` implies a similar select from `Elementary` for all observations of those sources.

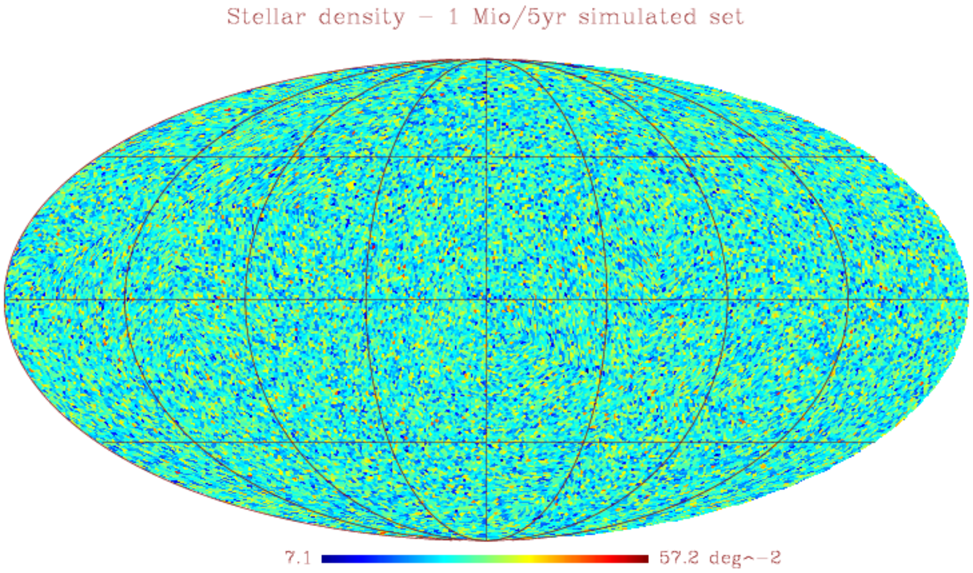


Figure 4.2: Uniform source density as requested from the CU2 simulations.

4.8.1 Source Relegation

Each source carries an attribute representing the ‘relegation factor’, named `relFactor`. This is a floating point number ranging from about 1 to infinity. A `relFactor` near unity implies the source is perfect for use in the astrometric solution. This then provides a continuous value for inclusion in the primary source selection process (see Section 4.8.2).

4.8.1.1 Setting the Relegation Factor

The Unit Weight Error (UWE) calculated during the source update is a good first estimate for the relegation factor. It is (or should be) close to 1 for a source that fits the standard five-parameter astrometric model to within the formal observational errors. A UWE much greater than 1 indicates some model mismatch, e.g., that the source is an astrometric binary. The UWE can later be combined with other factors indicating for example photometric variability, or some other potentially problematic property, so that in general the relegation factor can be determined with some formula using a combination of the source attributes.

For the first run with real data some selection should be made using the initial star catalogue – the resulting source may have the relegation factor set to 1. This will reduce the input to the first run of the astrometric solution. After the first run of the astrometric solution the relegation factor will be updated by the source update step. The secondary source update step runs on all sources not included in AGIS, hence this will set a relegation factor for all sources observed by Gaia.

4.8.2 Selection of primaries

The selection of primary sources is not as trivial as it seems at first glance. One may consider that one simply decides to have some number of sources evenly distributed over the sky. However the solution should also have at least some minimum number of sources across the magnitudes and colours to ensure no bias in these quantities. Furthermore it is suggested that magnitudes should be represented fairly evenly over the sky. We suggest a four phase selection process:

1. A relatively even sky/magnitude distribution of N sources where N is a number close to the total number of sources required in the solution Section 4.8.3.
2. Ensure there are enough sources in each magnitude bin Section 4.8.4.
3. Ensure there are enough sources in each colour bin Section 4.8.5.
4. Ensure there are N sources fairly evenly distributed Section 4.8.6.

4.8.3 Sky/magnitude distribution

The first step in the selection process is to get a relatively even distribution over the sky of some number of sources. Some sources are required in every part of the sky for the attitude reconstruction: the attitude cannot be reconstructed without several sources in each time bin ($\approx 15s$) in each FOV. In addition we need a representative set of magnitudes of sources across the sky. If we consider a histogram of the sources in magnitude bins, it is felt that magnitude bins of order 0.1 mag in G should be sufficient.

At this point it is easiest to tessellate the sky. In principle of course any tessellation scheme could be used for this selection but for statistical operations HEALPix (Górski et al., 2005) has an advantage (O'Mullane et al., 2001). The Hierarchical Equal Area Pixelisation scheme divides the sky first in twelve large pixels which are then divided and subdivided according to the choice of a value known as NSIDE. Fig. 4.3 shows this successive subdivision for NSIDE=0 2 4 and 8. NSIDE is always a power of 2 (representing the number of pixels on one side of the large zero pixel, and the total number of pixels in the sphere is given by $12 * NSIDE^2$.

In the pseudo query (below) we cycle through each HEALPix pixel to retrieve the same number of sources in each pixel per magnitude bin. Density variation then would only be within pixels. Hence the choice of HEALPix `NSIDE` is important as it predicts the level of homogeneity over the sky. For the photon density (see Section 4.8.7) a pixel size of about $1/3$ of the FOV is acceptable, this size should also work here and indeed it should not be larger than this.

The Gaia FOV is about 0.66 deg^2; hence we need a pixel size of approximately 0.22 deg^2 or $42153/0.22 \simeq 187514$ pixels. An `NSIDE` of 128 yields 196608 pixels and as such provides a good granularity. The loop here would then be from 0 to 196607. If we wish

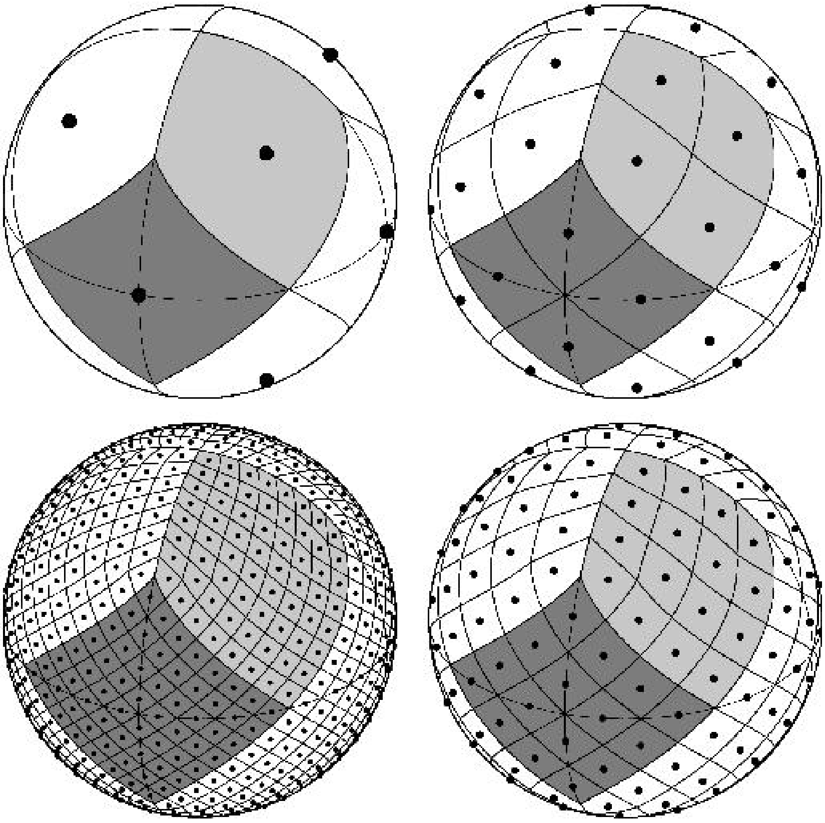


Figure 4.3: The sphere is hierarchically tessellated into curvilinear quadrilaterals. The lowest resolution partition is comprised of 12 base pixels. Resolution of the tessellation increases by division of each pixel into four new ones. The figure below illustrates (clockwise from upper-left to bottom-left) the resolution increase by three steps from the base level (i.e., the sphere is partitioned, respectively, into 12, 48, 192, and 768 pixels). from HEALPix website - the author built and maintains the Java version of this library in collaboration with Gorski and others

to have one hundred million sources we would choose *nPerHeal* to be $10^8/196608 \simeq 508$.

As discussed in Section 4.8.4 we have $(20.1-6)*10 = 141$ magnitude bins. In this case of one hundred million sources then we would want three sources in each bin in each HEALPix. Let this be denoted $nPerHealMag$

The `relFactor` is used to sort the data in ascending order so we take the sources in any given HEALPix pixel with the lowest relegation factor, i.e. those best suited.

Hence the system must allow us to perform a selection from the MainDB such as (pseudo SQL):

```
for (idHeal = 0 ; idHeal < nHealMax; idHeal++) {
  float magStart=6;
    int count =0;
    for (float b = magStart ; b <=20.1; b+=0.1 ){
        rowset = select * from Source into AGIS.Source
        where rowcount < nPerHealMag
            and healpixid = idHeal and magnitude between b and (b+0.1)
        order by relFactor ASC;

    count += rowset.numRows;
    }
}
```

It should be noted that for some tests where only a few million sources are used a larger HEALPix, or larger magnitude bins, will be needed to make the magnitude binning work i.e. thirty million is the smallest possible number of sources with $NSIDE = 128$ and 0.1 magnitude bins. If we choose the Timeline for the photon density the size of HEALPix is not an issue, both numbers need to be input parameters and their compatibility should be checked.

4.8.4 Magnitude distribution

Next we must histogram the resulting set of sources in magnitude bins over the entire sky to make sure we have a minimum number of sources per magnitude bin. It is felt that magnitude bins of order 0.1 mag in G should be sufficient. Each bin must contain some minimum number of sources, let us call this variable `minPerMagBin`. Then for each bin where the number of sources is less than `minPerMagBin` we should select more sources in that bin:

```
int binNum =0;
float magStart=6;
float magEnd=20.1;
for (float b = magStart ; b <=magEnd; b+=0.1 ){
   binNum=(int)((b -magStart)*10);
   if (magBins[binNum] < minPerMagBin) {
       select * into AGIS.Source from Source  where
               magnitude between b and (b+0.1)
```

```
              and (rowcount <  (minPerMagBin - magBins[binNum])
              order by relFactor ASC;
   }
}
```

The variable *magStart* here may seem strange but Gaia only observes to $G \approx 6$. It may also be worth considering $magEnd \approx 19$ as such sources add little weight and much computing.

4.8.5 Colour distribution

Analogous to Section 4.8.4, a histogramming and minimum must be applied for the colour bins to ensure that chromaticity effects may be correctly calculated. We assume that a unique mean 'colour' (it could be the effective wavenumber) can be computed for every source, and that the sources are binned according to this value. Let us consider `minPerColBin` as the minimum number of sources required per colour bin. Then assuming we have a histogram of sources per bin, `binCount`, we may do something like this:

```
for (float bin = 0  ; bin < maxBins; bin++ ){
    if (binCount[bin] < minPerColBin) {
       select * into AGIS.Source from Source s where s.bin = bin
              and (rowcount <  (minPerColBand - binCount[bin])
              order by relFactor ASC;
   }
}
```

In addition it may be necessary to ensure this colour histogram is equalised over certain time periods. Calibration parameters will be calculated in certain time intervals and we need a representative set of colours for these calculations. The scheme above may not yield such a distribution, the Gaia sky is certainly not evenly distributed in colour space, nor does it need to be. But we see enough of the sky over some time interval that an even distribution would be possible. This area requires some further thought.

4.8.6 Check overall coverage

Finally one should check if the overall coverage has been met i.e. that for each HEALPix id the count of sources is at least *nPerHeal*. Assuming the above selections all ended up in a database AGIS and AGIS.sources allowed us to access that table we would do something like this:

```
    int count =0;
  for (idHeal = 0 ; idHeal < nHealMax; idHeal++) {
        select count=count(*) from AGIS.source where healpixid = idHeal;
    if (count < $nPerHeal$) {
          select * from Source into AGIS.source
        where rowcount < (nPerHeal- count)
```

```
                and healpixid = idHeal  order by relFactor ASC
        }
    }
```

In actual fact this histogram (the count selected from AGIS.source) and the colour and magnitude histograms may be built up as the sources are selected - they do not need to be calculated afterwards. But this implementation optimisation will not be discussed here.

4.8.7 Photon Density

Although the primary selection scheme Section 4.8.2 attempts to make a relatively homogeneous selection of sources across the sky some inhomogeneity is inevitable (about 10-20% of the sky will fail the criteria). This could be dealt with by reducing the number of sources in each pixel to equal the pixel with the minimum sources, this however will reduce the overall number of sources used. Not placing a constraint on this distribution will greatly ease the selection of suitable sources for the astrometric solution. Since the source distribution is non-uniform we must take account of this in the attitude solution.

When calculating the weight of observations in the attitude the total number of photons in each FOV should be taken into account. In fact we need some estimator of $1/\sigma^2$ for each FOV. There are at least two approaches to this:

1. Looking at all observations over time and binning them in the FOVs for short time periods Section 4.8.7.1.
2. Constructing a Photon Density Map irrespective of time Section 4.8.7.4.

4.8.7.1 Photon Density Timeline

This is the superior approach in terms of correctness and the preferred approach to start with. In this approach the actual observed flux at any given time is known.

4.8.7.2 Construction of the Timeline

One pass through all of the primary *AstroElementaries* is needed to construct a Timeline for each FOV. A `timeInterval` must be chosen to sum the Photon counts over. The attitude spline which has knots at 15 second intervals (and may be variable later) must be considered here. Initially $timeInterval = 15$ seconds will be used, if necessary later longer intervals may be investigated.

Then for each FOV for each `timeInterval` seconds of the mission the total Photon Density should be calculated by summing the approximate photon count (e.g. $10^{-0.4\max(G,G_{\mathrm{sat}})}$, this may in fact ultimately be some non linear function of magnitude) over all the primary sources in that FOV during that time interval. Here G is the magnitude of the source and G_{sat} the magnitude at which the central pixel of the image saturates.

This would then yield two arrays of numbers with the total observed photon density per time bin. The order in which the data is read is immaterial as its time predicts which bin it should be added to i.e. for the time of the observation t the bin is $Floor((t - startTime)/timeInterval)$

Such Timelines may be constructed as data is loaded into the AGIS database or in one pass through the data after it is loaded.

4.8.7.3 Using the Timeline

For the photon density at any given time one may simply pick the bin which that time falls in as the photon density i.e. for time t the bin is $floor((t - startTime)/timeInterval)$

4.8.7.4 A Photon Density HEALPix Map

This was originally considered the simpler approach but the Timeline approach Section 4.8.7.1 is better and not necessarily more complex to construct.

4.8.7.4.1 Constructing the Photon Density HEALPix Map

A HEALPix map may be constructed which sums the photon counts from the sources in the HEALPix pixel. This may, as a first approximation, be calculated by summing a quantity like $10^{-0.4\max(G,G_{\text{sat}})}$ over all the primary sources in a particular HEALPix pixel. Here G is the magnitude of the source and G_{sat} the magnitude at which the central pixel of the image saturates (or rather, where the first gate is activated). An `NSIDE` of 128 is chosen to give a pixel area of around 0.2 deg^2 which is one third of the FOV. Since we will be interested in neighbouring pixels this is best considered a map in the nested scheme. Such a map may be constructed as the sources are loaded in the astrometric database and is valid for the AGIS cycle as the magnitudes do not change in AGIS. As each source is loaded the HEALPix library may be invoked as follows:

```
int nside = 128;
HealpixMapCreator mc = new HealpixMapCreator(nside);
HealpixMap m = mc.getMap();
m.setScheme(Scheme.NEST);
long pix =0;
double theta;
double phi;
for { Source s: primarySources }  {
    \\ check units and conventions on these very carefully.
    theta = toRadians(s.getAlpha()) ;
    phi = toRadians(s.getDelta()) ;
    pix = Healpix.ang2pix_nest(nside,theta,phi);
```

```
            m.add(pix,GMath.pow(10,(Math.max(s.getG(),GSAT)*-.4));
        }
        \\now store m !
```

A map of this sort does not account for the case when a source is not observed in a FOV at a particular time due to crowding, but this should be a small effect.

4.8.7.5 Using the Map

The HEALPix library allows one to select all pixels around a certain pixel or point on the sphere. For our map above we would want the 10 pixels relating to each FOV and then average these in each FOV. The HEALPix library provides a routine for this extraction. If we consider `m` to be the map above then we may calculate the photon density in a given FOV like this:

```
      long pix = m.ang2pix(fovTheta, fovPhi);
      int width = 3;
      count = width * width;
      //extract 3 pixel box centred on pix.
      int [][] pixNums = m.box(pix, width,width);
      // could use query_disc or polygon for this instead
      double tot = 0;
      int count = width * width;
      for (int i =0; i < pixNums.length; i++){
          for (int j =0; j < pixNums[i].length; i++){
             tot += m.get{pixNums[i][j]};
          }
      }
      double fovDensity= tot/count;
```

4.8.7.6 Using the Photon Density

Which ever approach is taken for calculating the Photon Density it is used to look up the Photon Density in a given FOV which is then used to reduce the weight of observations in one or other FOV.

Let d_{P} and d_{F} be the photon densities in the preceding and following FOV. Then if $d_{\mathrm{P}}/d_{\mathrm{F}} > R$ we apply a weight reduction factor of $d_{\mathrm{F}}/d_{\mathrm{P}}$ to the observations in the P-field (that is, we increase their formal errors by the factor $\sqrt{d_{\mathrm{P}}/d_{\mathrm{F}}}$). Conversely, if $d_{\mathrm{P}}/d_{\mathrm{F}} < R$ we apply a weight reduction factor $d_{\mathrm{P}}/d_{\mathrm{F}}$ to the observations in the F-field. Nominally $R = 1$, the procedure can be modified to allow a certain maximum weight ratio $R > 1$ between the fields (where R may be of order 2–3) rather than strict balancing of the weights. The optimum R remains to be determined, van Leeuwen used $R = 2.7$ in the Hipparcos reprocessing (van Leeuwen & Fantino, 2005).

It seems sensible to pre calculate the weight reduction factor $\sqrt{d_{\mathrm{P}}/d_{\mathrm{F}}}$ since it will be required in every attitude calculation for each iteration of AGIS. We could consider a two dimensional array where each row represents one `timeInterval` and consists of two values: the first the weight reduction for the preceding FOV and the second for the following. One of these numbers would always be one and the other containing the weight reduction as outlined above. The actual look up would be as for the Timeline outlined above Section 4.8.7.3.

4.8.7.7 Photon Density Ratio Server

We propose to present this Photon Density Ratio using a `DataServer` in a manner similar to the Attitude and Calibration Data Servers. Hence the interface would have a method `getFovWeightReductions(GaiaTime t)` which would return the weight reduction factors for the two FOV at the given time. These numbers could then simply always be used in the attitude calculations. A *DefaultPhotonDensityRatioServer* could be created which simply returned a pair of ones to be used in case we do not wish to use the weight reduction e.g. to test its effect.

The server would be instantiated by the `AgisFactory` in the same manner as the other servers and could live without update for the entire AGIS cycle.

4.9 Conclusion

In this chapter the main phenomena and peculiarities of Gaia, which AGIS must deal with, have been outlined. Some phenomena are dependant on magnitudes, colours or spectral types of sources, Section 4.8.7 explains how it is intended to cope with this in pre selecting “primary” sources for AGIS. The following chapter Section 5 outlines some tests with simulated data showing how well AGIS already deals with some of these cases.

Chapter 5

AGIS results

> Results! Why, man, I have gotten a lot of results. I know several thousand things that won't work.
>
> **Thomas Alva Edison (1847-1931) American inventor.**

The AGIS software which has been described in Section 3 has been running on simulated test data since 2005 with an Oracle Store as the back end. Simulated data in fact provides the only possible proof that AGIS does not introduce systematic errors in the solution since the *true* sources and observations are known to compare against. Typically the simulation adds noise which AGIS should remove, how well it removes this noise may be precisely judged by comparing the resulting astrometry to the *true* values. Many effects in Gaia observations are not distinguishable (e.g. an aberration in a mirror or slight displacement of a CCD are calibrated in the same manner) in the real data so we must be confident that AGIS does not introduce systematic errors itself.

In this chapter we present some of the initial tests carried out to verify that AGIS does indeed provide the astrometric solution for Gaia observations. These results come from many years of testing and many failures as well as successes - the Edison quote at the beginning of the chapter echoing perfectly the sentiments on testing AGIS, there are many things now known not to work. Perhaps more optimistically we find AGIS runs both efficiently and correctly.

5.1 Test campaigns

The effort in running AGIS tests should not be underestimated. These are not the sort of test which the author has run on his desktop machine of an evening. Rather it takes a little more hardware (Section 5.2), time and planning. Hence it would scarcely be possible to run these tests without the excellent Gaia team at ESAC.

The production of the high quality simulation data alone is a major task to which the entire processing consortium is indebted to CU2. The AGIS group has been fortunate to have these simulations since the beginning of the development at ESAC in 2005.

The types of tests to be performed on AGIS are now outlined in the software test plan (DH-001). A series of test campaigns are planed up to launch. Each individual test introduces some particular noise in the data which AGIS should then remove. Other tests (usually denoted T4) combine all the noise effects to make sure AGIS can remove noise from attitude, astrometry and calibration. Many of these tests are very straight forward, a few are however of particular interest and are discussed below.

5.2 The Machine

The machine used for the Gaia tests up to 2010 was purchased in instalments by ESA. The initial machine for the first tests in 2005 consisted of twelve nodes each with 6 GB of RAM and two processors (3.6GHz Xeon EM64T). An EMC storage area network (SAN) with 5 TB of disk is attached to the nodes using fibre optic cable and the nodes are also connected via Gigabit Ethernet cards in a local area network. AGIS has also been tested, and shown to run as fast, using a cheap Rack Server Network Attached Storage device of 6 TB. This cluster was upgraded to 18 nodes in 2006 and a further 4 nodes with quad core processors were added in 2007. The quad core processors functioned very well for us providing the performance of 4 processors and in Table 3.1 the number of processors counts each core as a processor.

This is not a very special machine, it consists of standard Dell power edge blades with standard Intel Xeon processors running Red Hat Linux. The blades are housed in a standard DELL rack. There is no special HPC software used, a network of normal Linux machines could work in the same manner. The blades tend to be tidier and more power efficient for cluster computing.

Around 2008 ESAC also started running IDT and FL on a regular basis and the cluster was no longer dedicated to AGIS. In 2010 a new IBM cluster was purchased with newer Xeon multicore chips but very much in the previous vein. The total compute power at ESAC in 2011 was bout 1.5 TFLOPS of which 0.5 TFLOPS was frequently used for AGIS testing.

5.3 Result histograms

In all of the histograms in this section the text line at the top of each histogram has the following structure:

$$\text{N: } X \ [a/b] \text{ - M: } m \text{ - W: } w$$

with:

- X: Total number of values within the monitored and visible abscissa range
- a/b: a/b values less than the lower/larger than the upper bound of the visible abscissa range
- m: Median of the distribution in μas calculated from the values within the histogram's bounds - the position of the median is also shown in the histogram as a vertical blue line
- w: Width of the distribution calculated as the difference between the 90th and 10th percentile

One should also note that the scale on the plots - varies with the distribution of values.

5.4 Initial results

The very first results from AGIS were published within DPAC early in 2006 (UL-015) This was with one million sources simulated over just eighteen months of the mission. This behaved very well making a major adjustment to noisy source positions already in the first iteration. Fig. 5.1 shows the source position updates for this first published AGIS test. This was actually the 373^{rd} run of the AGIS system since work started at the end of 2005. All updates in the cycle were in line with expectations from the simulation. This was a very satisfactory initial result for AGIS.

Further test results in 2006 were mainly to show the system converged in a reasonable time with five years data rather than just 18 months. Simple Gaussian noise was added to the attitude, source and calibration parameters of a one million star five year data set. The data consisted only of well behaved stars which would work in an AGIS solution. This was published in detail in (UL-018). Most of this set of results went very well but the update to the Z component of the attitude showed an unexplained spread as may be seen in Table 5.2. This lead to an extensive investigation of the attitude update with some errors discovered. Attitude update remained one of the AGIS weaknesses until the attitude update code was eventually completely re-written in 2007.

5.5 Viability of the solution

Allowing for an increase in complexity in the AGIS system of a factor 10 from this initial system the target machine assumed to be available at ESAC around launch could perform an AGIS cycle on 100 million sources in about thirty days (UL-018). This initial extrapolation was very pleasing. In 2008 AGIS was more complex with an improved attitude reconstruction and more sophisticated calibrations yet it was three times

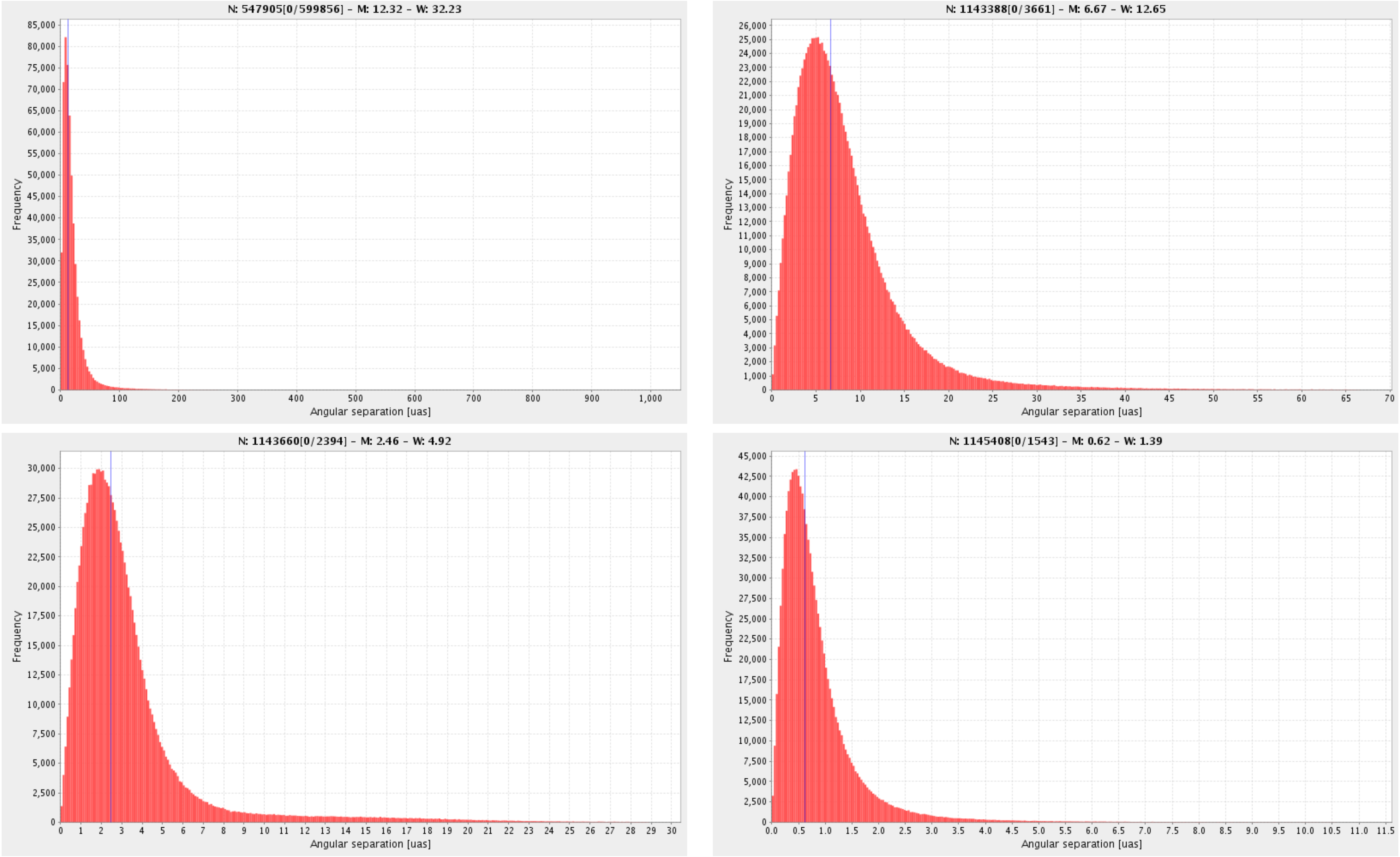


Table 5.1: Source position update histograms for iterations 1 (top-left), 2 (top right), 3 (bottom left) and 4 (bottom right) of cycle 373. One of the first AGIS results in 2006, we see already in four iterations the bulk of the position updates drop to under 0.5 μas (note the scale of the plots change).

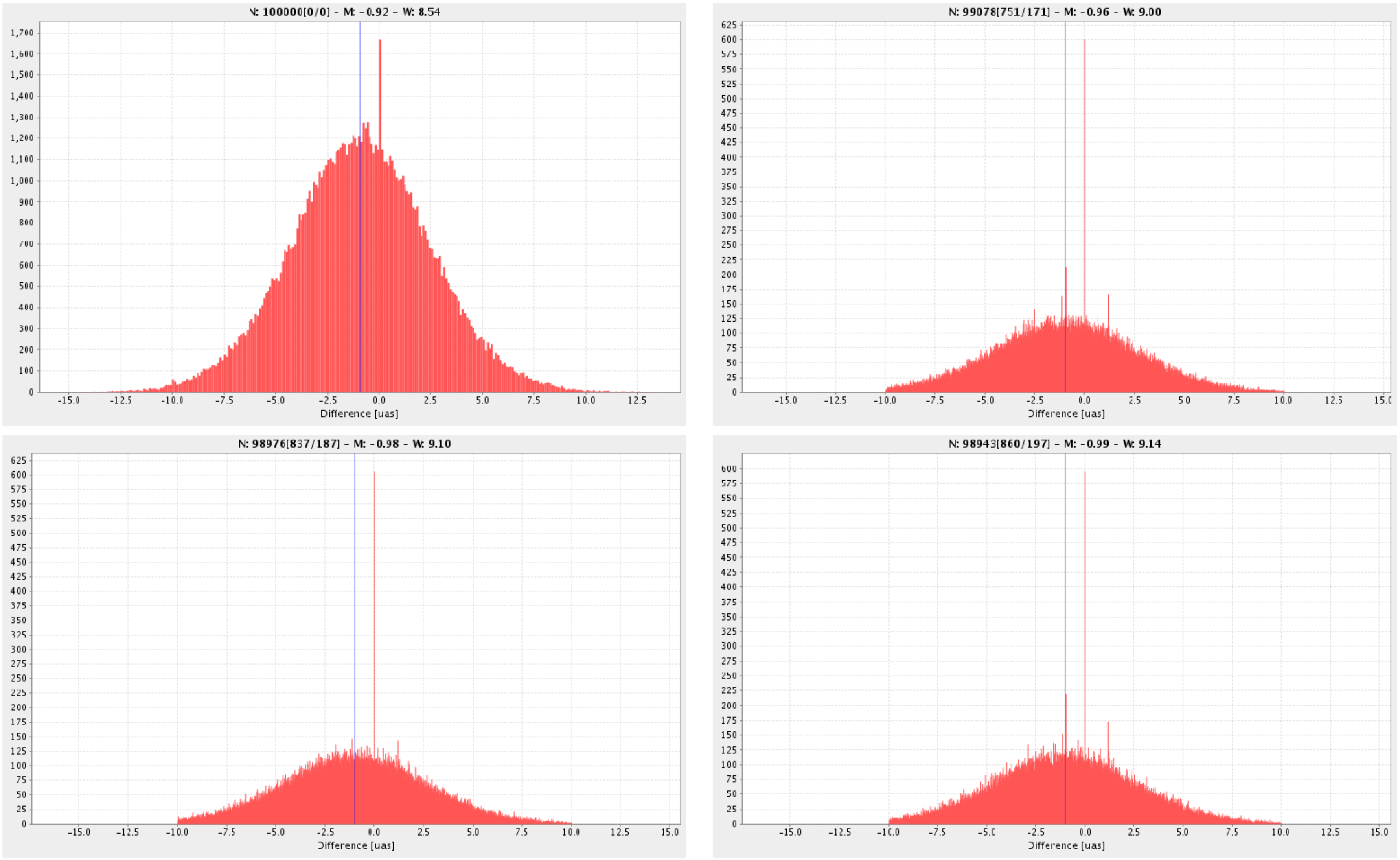


Table 5.2: Updates to the attitude quaternion in the Z axis histograms for iterations 1 (top-left), 2 (top right), 3 (bottom left) and 4 (bottom right) of cycle 41 in the second test campaign.

faster than the original 2005 system (see Table 3.1. AGIS in 2011 is almost complete in terms of functionality (but requires quite some work on robustness of the software for operations) and has demonstrated ability to process 50 million sources in about eight weeks on a subset of the IBM nodes spoken of in Section 5.2. Hence we remain confident that the final AGIS should complete on a reasonably sized machine in between four and eight weeks. Here of course money will be the issue, how long may we wait for the results - is one month acceptable considering the difference in price of the machine needed to make it in two weeks?

5.6 More advanced tests

From testing the initial implementation in 2006 a basic confidence in the stability and suitability of the system was gained. Next some more involved test campaigns were designed to identify and fix other potential problems in the system. Each time a test campaign was undertaken an initial test with noise free inputs was performed to ensure no coding error had introduced some strange behaviour in the system. These tests will not be presented here but should be considered when viewing more complex results. Following the initial sanity check the perturbed simulation data is put through the system - below a few of the tests are discussed.

5.6.1 Reference frame rotation

In all AGIS tests the true source positions are known from the simulator, hence plots of the actual error on the source position are possible while testing. Thus while Table 5.1 shows the update in each iteration Fig. 5.1 shows the actual difference from the true value or the absolute error which of course is only available when one uses simulations.

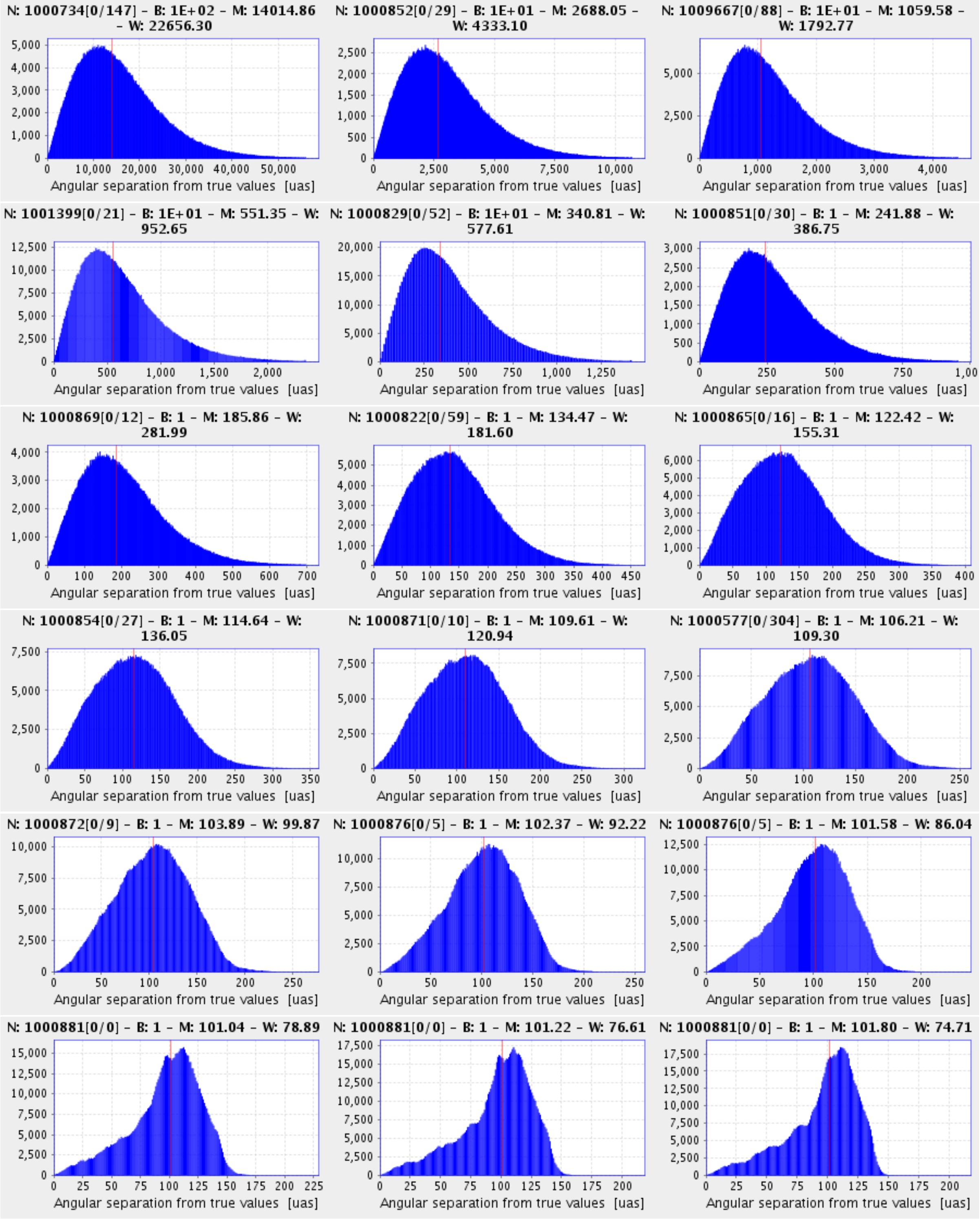


Figure 5.1: Position error (diff from true) histograms for iterations 1 (top-left) to 39 (bottom right) of cycle 40. Notice that a relatively large error remains to the final iteration which may be explained by a rotation of the reference frame one degree of freedom exists in the solution. Note: not all iterations are shown.

As may be seen from Fig. 5.1 even after convergence when the parallax update is less than 1μas there remain errors or some hundreds of μas in very many source positions. This is still a vast improvement over the initial errors, the error being reduced by three orders of magnitude.

This is an expected manifestation of the fact that any AGIS solution is unique only up to a rigid rotation and constant spin of the AGIS-internal reference frame with respect to the true or simulated sky. Algebraically speaking, the problem has a rank defect of 6.

In fact this is immediately visible when we colour code the source position errors globally as depicted in Fig. 5.2, here the very low errors (blue in the figure) are seen around the poles of the rotated reference frame.

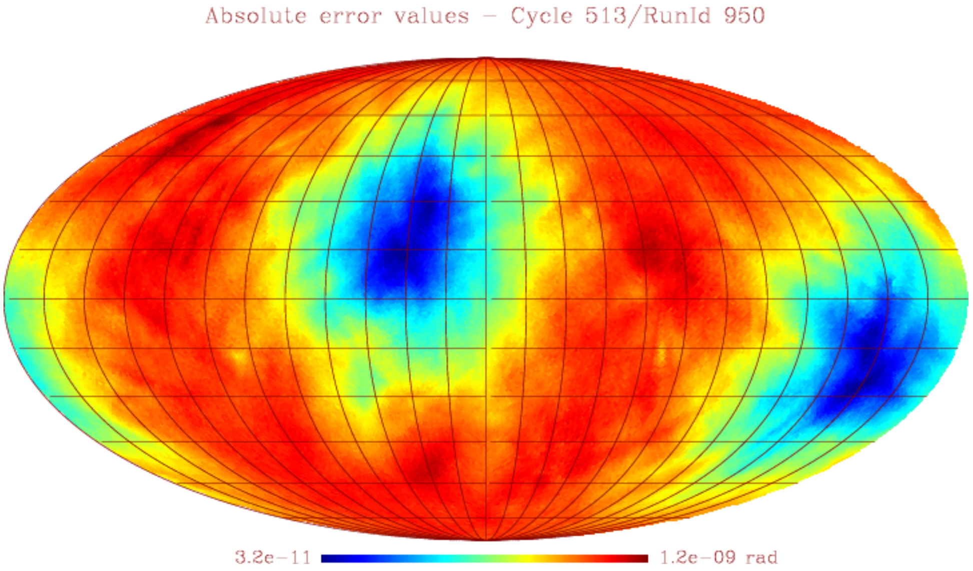


Figure 5.2: Source position errors after AGIS convergence clearly showing a rotation of the reference frame in the solution.

For the Gaia mission, quasars in the International Celestial Reference System (ICRS) will be identified in the catalogue. These sources will be used to work out the precise rotation of the solution, for example by simple least squares minimisation. In the simulations the correct source positions are known and so one hundred or so sources scattered over the sky may be used to work out the rotation vector. This rotation may then be applied to all sources to put them in the ICRS. The resulting rotated error map is shown in Fig. 5.3 where it is clear the errors are at μas level and reflect a trace of the scanning law as expected.

5.6.2 Basic Angle variation

The angle between the two Gaia telescopes is termed the ‘Basic Angle’ (see Section 4.2). The rigidity of this angle is a primary design goal of the Satellite however it is accepted that some variation will occur in the angle. The basic angle is so important as it is the key to the absolute parallax measurements. A variation in the basic angle

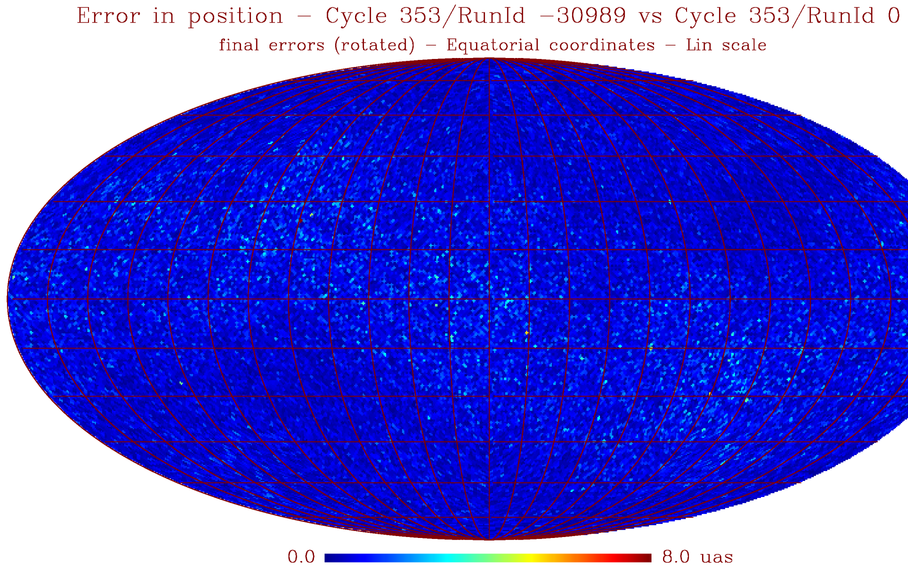

Error in position – Cycle 353/RunId –30989 vs Cycle 353/RunId 0
final errors (rotated) – Equatorial coordinates – Lin scale
0.0
8.0 uas

Figure 5.3: Source position errors after rigid rotation of the coordinate system.

is indistinguishable from a global shift in parallax (GAIA-LL-057) on a short period of time i.e. less than the spin period. Hence the Basic Angle Monitor (BAM) will be on board the satellite to monitor this carefully. Variations over a longer period in principle should be observed in the geometric along scan calibration.

The simulated data set GASS-LSS-1-C contains a variation in the basic angle with a period of longer than six hours. The resulting AL calibration plot is shown in Fig. 5.4. The variation is corrected for almost completely in the first AGIS iteration.

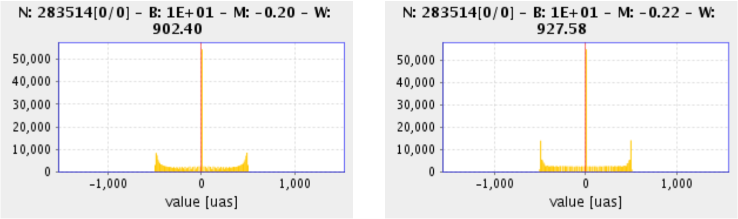


Figure 5.4: Large-scale AL calibration histograms for iterations 1 (left) and 8 (right) of cycle 62. The basic angle variation in the data is quickly seen in the AL calibration updates.

Lammers plotted this basic angle update more aesthetically in Fig. 5.5, here we clearly see the periodicity of the variation in the basic angle and how well AGIS has modeled it. The error is reduced to less than $1\mu as$.

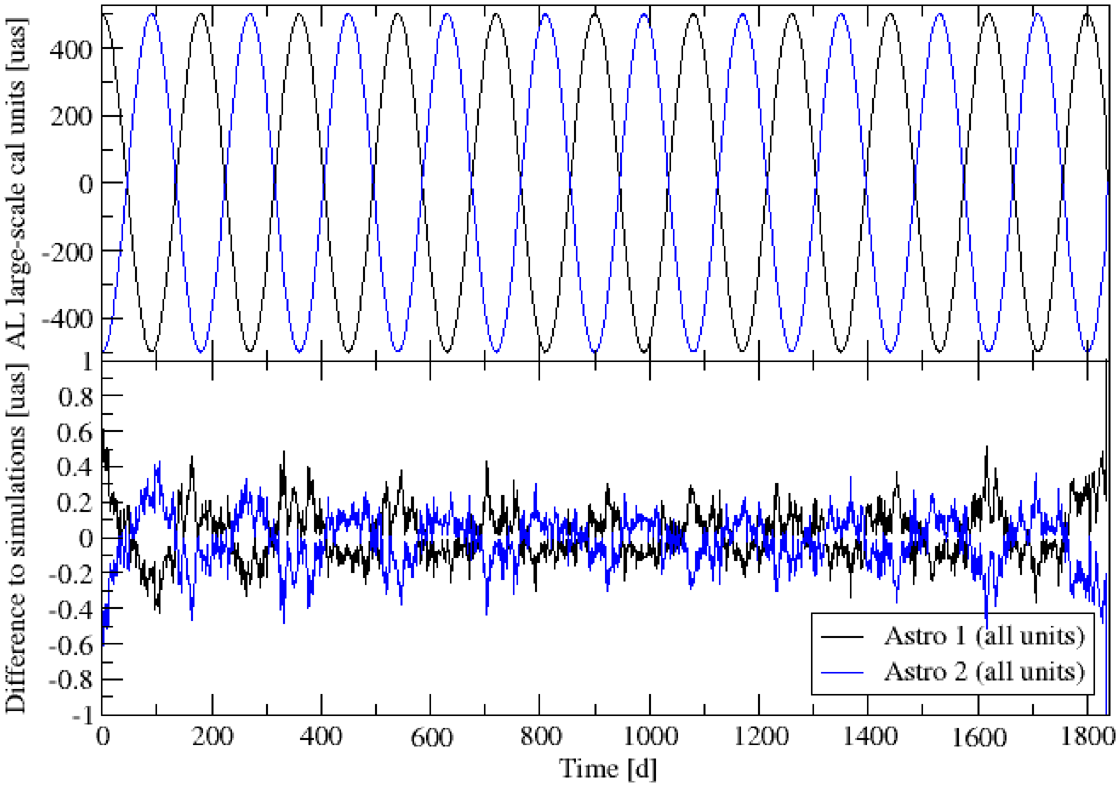


Figure 5.5: Time-evolution of basic angle recovered in T1 (cycle 62). Top pane: Final values of AL large-scale calibration parameters averaged over all AF CCDs as a function of time for the two telescopes Astro-1 (black) $c_0^{A1}(t)$ and Astro-2 (blue) $c_0^{A2}(t)$. Bottom pane: Differences $c_0^{A1}(t) - 1/2\Delta\gamma(t)$ (black) and $c_0^{A2}(t) + 1/2\Delta\gamma(t)$. So, the bottom curves show how well the basic angle variation is recovered in the AL-large scale calibration parameters. plot courtesy U. Lammers

5.6.3 Magnitude dependant centroid shift

The CU2 simulation data set GASS-LSS-1-D (XL-004) contained a magnitude dependent along scan (AL) shift according to (5.1), this is done by changing the AL field angle η:

$$\Delta\eta(t) = \Delta\eta_{ref} + C_0(G - G_0) \tag{5.1}$$

where G is the apparent magnitude of the source, G_0 a constant chosen approximately equal to the mean magnitude of the primary stars, and C_0 a time-independent coefficient chosen, for each CCD, from a uniform random distribution between 0 and +1 mas/mag.

The test report (NB-001) contains the full details of the testing (89 pages). The report concludes that adding a calibration parameter for linear magnitude shifts does not significantly alter the convergence behaviour of the AGIS solution. As may be seen from Fig. 5.6 AGIS recovers the C_0 coefficients to sub μas level.

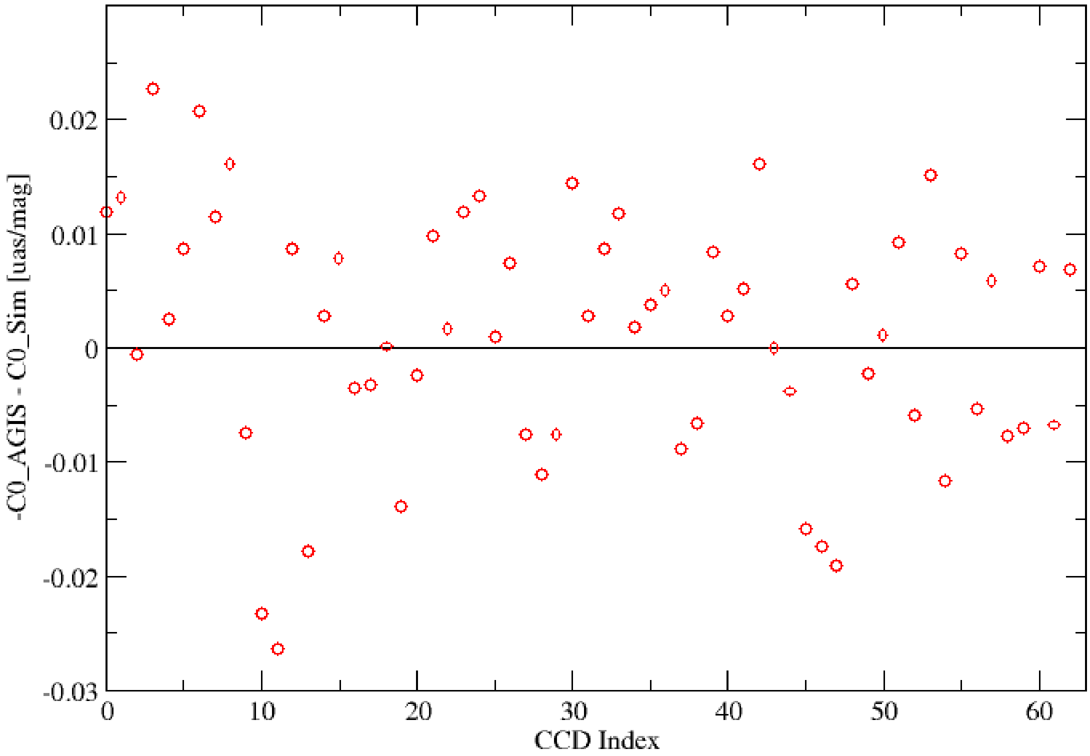


Figure 5.6: Coefficient comparison between values recovered from AGIS last iteration and simulation values in μas/mag as a function of the AF CCD number (from NB-001).

5.6.4 Spectrum dependant centroid shift

The simulation data set GASS-LSS-1-E (XL-004) contained an AL shift dependent on the spectrum of the source as per Eq. 5.2:

$$\Delta\eta(t) = \Delta\eta_{ref} + C_1(W - W_0) \tag{5.2}$$

with W is the effective wavenumber of the source, $W_0 = 1.5\,\mu\text{m}^{-1}$ and C_1 is a coefficient chosen, for each CCD/FOV combination, from a uniform random distribution between -1 and $+1$ mas $\cdot\,\mu$m, to remain constant over the mission.

Here the spectrum is represented by the effective wave number which is computed from the five component ('BBP') fluxes f_k, $k = 0...4$ according to (5.3):

$$W = \frac{\Sigma_k f_k \lambda_{k+1}^{-1}}{\Sigma_k f_k} \tag{5.3}$$

The full report (NB-002) contains many finer details of the test campaign. Fig. 5.7 shows the comparison between the AGIS-recovered values for the linear chromatic shift parameters C_1 and the corresponding true simulation values as a function of CCD number for both FOVs separately, all absolute values lie below 0.6 μas.

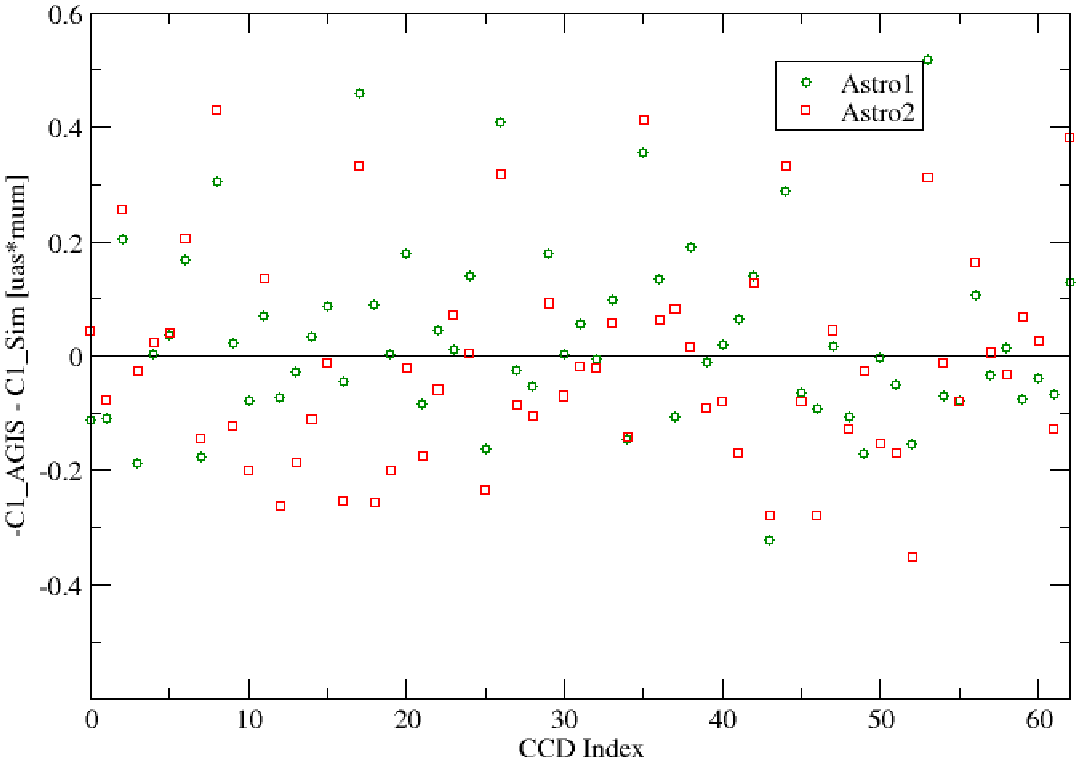


Figure 5.7: Coefficient comparison between values recovered from AGIS and simulation values in μas $\cdot$ μm as a function of the AF CCD number for the two FOVs (from NB-002).

5.6.5 More complex spatial calibration

The initial calibration model included in AGIS only dealt with the CCD position shifts within the plane. Shifted Legendre polynomials L_n^* in a coordinate $\tilde{\mu}$ running from zero to one along the trailing edge of each CCD were added as large scale calibration parameters in 2007 (GAIA-ARI-BAS-011-05). The η equation (2.50) then becomes slightly more complex, rather than a single $\Delta\eta$ term we have (removing the original indexes):

$$\eta_l = \eta_n^0 + \Delta_{\eta 1} L_0^*(\tilde{\mu}) + \Delta_{\eta 1} L_1^*(\tilde{\mu}) + \Delta_{\eta 1} L_2^*(\tilde{\mu}) + \delta\eta \tag{5.4}$$

These polynomials account for rotations and stretching of the CCDs. The first order of this polynomial is equivalent to the original calibration. Hence running AGIS again on a previous dataset with known outcome provided some assurance that the new parameters did not introduce errors.

The GASS-LSS-1-F (UL-021) contains a polynomial derived term in the Across Scan (AC) field coordinate (η). Fig. 5.8 shows the difference between the retrieved coefficients and those actually used per CCD after 33 iterations of AGIS. The rather odd alignment of the SM C_0 values is due to the fact the SM CCDs are not included in the calibration update. The C_1 parameter is recovered to within 20μas which is acceptable but could be better. The C_2 parameters are recovered to less than 1μas. Clearly some work remains to be done in this area but the current results are considered quite acceptable at this stage.

Typically AGIS runs with a complete calibration model regardless of the input data - only T4 tests have all effects included in the simulation.

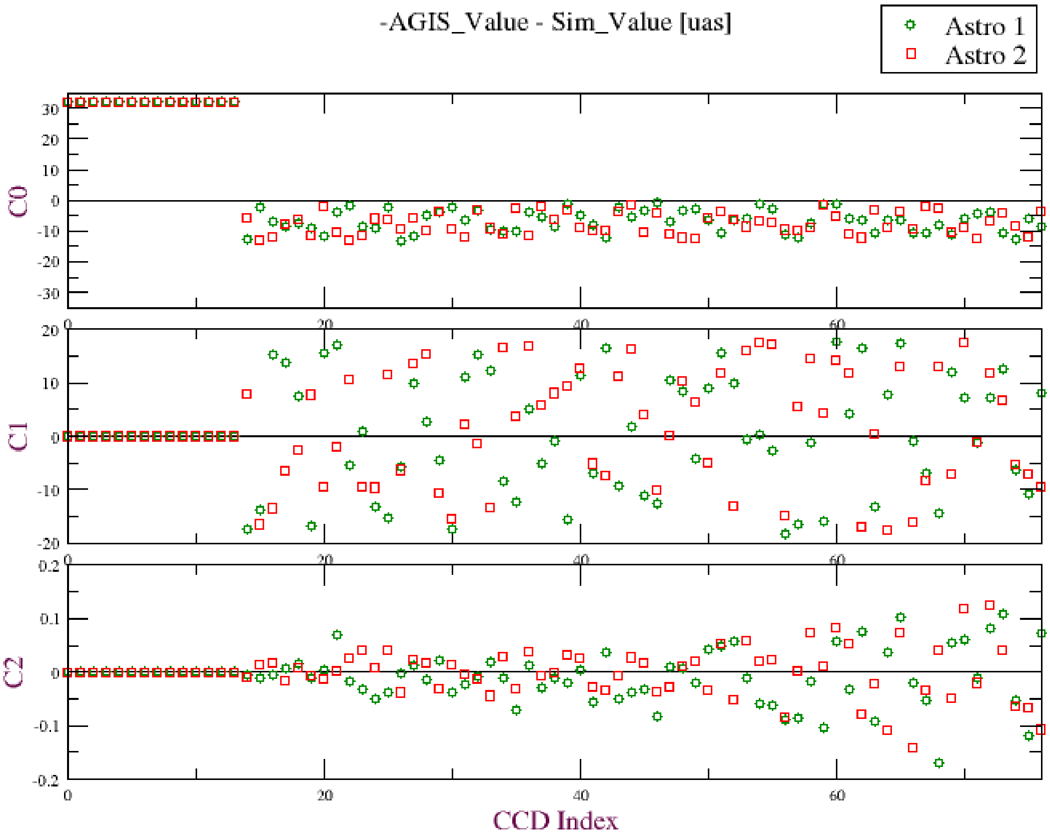


Figure 5.8: Difference in coefficients of the Legendre polynomial per CCD from GASS-LSS-1-F data, each FOV is shown as a separate data set with one panel per coefficient. From (NB-003).

5.7 Looking forward - AGIS during the mission

AGIS will run on one hundred million primary stars during the Gaia mission. Current thinking has this occurring annually throughout the mission and once after all mission data has been collected and reprocessed by the Intermediate Data Update. The results presented here have been for only one million stars or one percent of the final number of primaries. Tests are scheduled for larger numbers of stars, and 50 million have been processed, but let us extrapolate here on the current set up first.

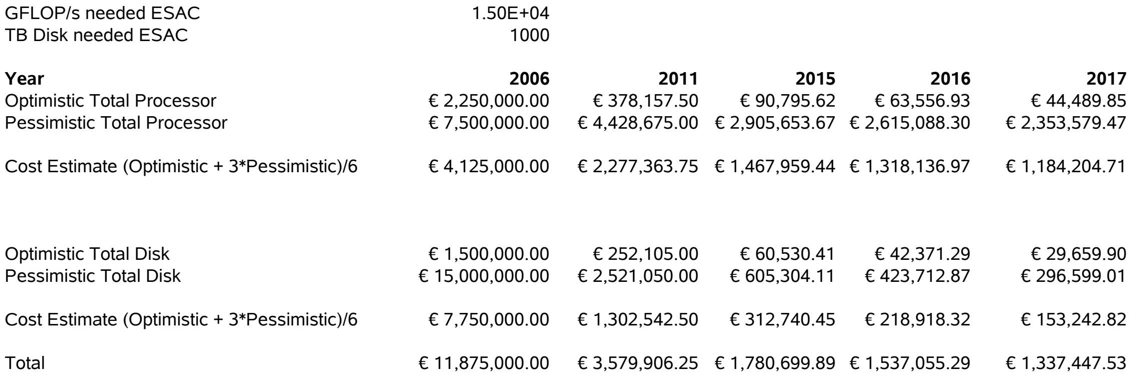
Currently AGIS requires forty iterations to reach convergence. On the present hardware that requires about forty hours of computing time. Since sources processing time varies little this time depends on the number of sources. It is speculated (but not proved) that fewer iterations may be needed with more sources however let us assume we always require forty iterations. If we then say the current sources are one percent of the final number the final number should take one hundred times longer or about four thousand hours or one hundred and sixty seven days. If we consider a processing cycle of one year or even six months this would already be an acceptable run time.

AGIS is of course not finished and it will have additional processing added to it. Although we have seen (in Table 3.1) that more complex processing has not caused an increase in processing time to date, let us assume we are less fortunate in the future and that AGIS is a factor five slower than today. This would mean on the current system spending more than two years to process the one hundred million sources. Two years is of course totally unacceptable.

So then let us look at hardware. The current hardware system would be a mere toy if looked at retrospectively from mid mission say 2015. Even by today's standards it is not particularly powerful rating only about 140 GFLOPS (Giga Floating Point Operations per Second). This machine has been built in parts since 2005 at a cost of around 200K. It is hoped to have a 10 TFLOPS peak computing facility at ESAC during the Gaia mission. Here Moore's law is heavily relied upon to make such a purchase feasible. Gordon Moore (Moore, 1965) told the world (back in 1965) that component density (read processor power) would double every year. His formula has only been slightly revised to eighteen months rather than one year for fitting the last forty years of data. This trend now seems set to last until at least 2017, the production of multi core processors (Intel have hinted at a 64 core processor) leads one to have faith in this. Certainly the huge variation in price will not diminish but the range outlined for the ESAC hardware in the year 2017 would be between 70K and 2.6 Million[1]. Using a PERT costing on a numbers like this we get about 1.3 Million for the machine. These numbers are detailed in Fig. 5.9.

FLOP estimates ignore I/O. We also have a relatively large volume of data which must be operated upon. The volume is too large to consider holding it at all in any form of shared memory so it will need to be repeatedly read from disk. The choice of hardware architecture is therefore quite important in that a super fast machine with bad I/O will probably not do the work. Indeed early work such as (GMV-GDAAS-RP-001) showed

[1] This pessimistic number also assumes Moore's law does not hold true



GFLOP/s needed ESAC	1.50E+04				
TB Disk needed ESAC	1000				
Year	**2006**	**2011**	**2015**	**2016**	**2017**
Optimistic Total Processor	€ 2,250,000.00	€ 378,157.50	€ 90,795.62	€ 63,556.93	€ 44,489.85
Pessimistic Total Processor	€ 7,500,000.00	€ 4,428,675.00	€ 2,905,653.67	€ 2,615,088.30	€ 2,353,579.47
Cost Estimate (Optimistic + 3*Pessimistic)/6	€ 4,125,000.00	€ 2,277,363.75	€ 1,467,959.44	€ 1,318,136.97	€ 1,184,204.71
Optimistic Total Disk	€ 1,500,000.00	€ 252,105.00	€ 60,530.41	€ 42,371.29	€ 29,659.90
Pessimistic Total Disk	€ 15,000,000.00	€ 2,521,050.00	€ 605,304.11	€ 423,712.87	€ 296,599.01
Cost Estimate (Optimistic + 3*Pessimistic)/6	€ 7,750,000.00	€ 1,302,542.50	€ 312,740.45	€ 218,918.32	€ 153,242.82
Total	€ 11,875,000.00	€ 3,579,906.25	€ 1,780,699.89	€ 1,537,055.29	€ 1,337,447.53

Figure 5.9: Cost projection using PERT analysis for SOC hardware. Here disk and FLOP estimates are taken from the current setup where prices are known. A Moore's law scaling is then used to give a best price estimate at several future points. We see that delaying the purchase of the final hardware to 2017 is highly cost-effective.

that I/O is a significant problem for Gaia processing. Furthermore in the AGIS developments it was seen that reorganising the data access has significant impact on the system.

Let us then return to AGIS of the future with one hundred times more data and five times more processing to do. Hence we scale AGIS by five hundred. On the current 140 GFLOPS machine this scaled up AGIS would take about 1000 days. The 10 TFLOPS machine however is 70 times more powerful than the current system so it could do the scaled up AGIS in 15 days, assuming I/O is dealt with as efficiently as in the current system. Even allowing another factor two for I/O slowdown and problems this would lead us to around a 30 day run time for the final AGIS. This is a perfectly acceptable run time. More recently in 2010 a 50 Million source five year run was performed confirming these predictions, it took over a month but not on the final hardware using only about 0.5 TFLOPS of the Cluster. This AGIS is also the most complete yet and quite representative of the final product. This version of AGIS was also used to produce the tests for (Lindegren et al., 2012), in that paper applying some scaling factors to the runtime a final solution is also predicted as not taking more than 60 days on the future hardware.

If we fail to make AGIS fast enough either by having computational problems or by failing to buy sufficient hardware we still have the possibility to run AGIS first on a subset of primaries to work out large scale calibrations and only later include all primaries to calculate the small scale calibrations. This was mentioned earlier in Section 4.4.

5.8 Conclusion

The ESAC team have been busy testing AGIS with various simulation data sets for over five years. Each set has improved AGIS and demonstrated that it can retrieve or smooth out the sort of effects expected in the Gaia astrometric instrument. Perhaps most importantly, the tests have proved a Global Iterative solution is both viable (Section 5.5) and useful for the Gaia Astrometry. With the AGIS system as of 2011 running an iteration over 50×10^6 sources in twenty four hours on a modest multiprocessor system it would be possible to process 10^8 sources in about three months. We may safely deduce this, as the time to completion for AGIS is linear with the number of sources included. Undoubtedly the algorithms will become more complex but the computers will become faster. It is in any case the intention to have a far more powerful machine for the final system. No other approach to the Gaia astrometry has proved viable to the level demonstrated by AGIS and briefly summarised here. The testing is by no means finished and will continue for several more years to ensure the best astrometry may be extracted from the Gaia data.

Chapter 6

The development approach

合抱之木，生于毫末;
九層之臺，起于累土;
千里之行，始于足下。

> The tree which fills the arms grew from the tiniest sprout;
> the tower of nine storeys rose from a (small) heap of earth;
> the journey of a thousand li commenced with a single step.
>
> **Lao Tzu (604-531BC) Chinese Philosopher (Tzu, 1994, Chapter 64).**

The development presented in Section 3 was carried out quite rapidly using *Agile*[1] techniques in particular eXtreme Programming (1999). The approach followed is outlined in this section as it is felt the building and directing of the team to build the software is an important achievement. The approach followed borrows from many sources and personal experience gained in developing operational software over many years. For readers within DPAC this section is a revision of (WOM-006). Initially Section 6.1 posses a question which is argued frequently within ESA - is science development essentially different to satellite development?

6.1 Is development of Science software different?

The author conducted a survey over a period of one year of many large science developments (WOM-003). This proved to be a most interesting study and concluded that science software development is indeed different to traditional software development due to the funding structure and general approach to leadership. This is in any case

[1] Agile is a collective word used today for iterative and incremental development techniques (or *not* waterfall development) `http://en.wikipedia.org/wiki/Agile_development`

quite different to a satellite development project. Despite this study and many interactions within ESA they seem set to impose a much more traditional project management structure on the DPAC. This is lamentable and will cause a loss of effort. The approach to development within DPAC of using cycles and always having some working software leans much more toward the *Agile* techniques discussed here than traditional project management. A paper on this topic was presented at ADASS in 2006 (O'Mullane et al., 2007). Indeed these days there are few if any software companies remaining who follow the ESA waterfall approach to development. The most lamentable thing of all is perhaps that Royce, attributed with inventing the waterfall model, showed the waterfall approach in 1970 (Royce, 1970) Fig. 6.1 as an example of the flawed way to do software development.

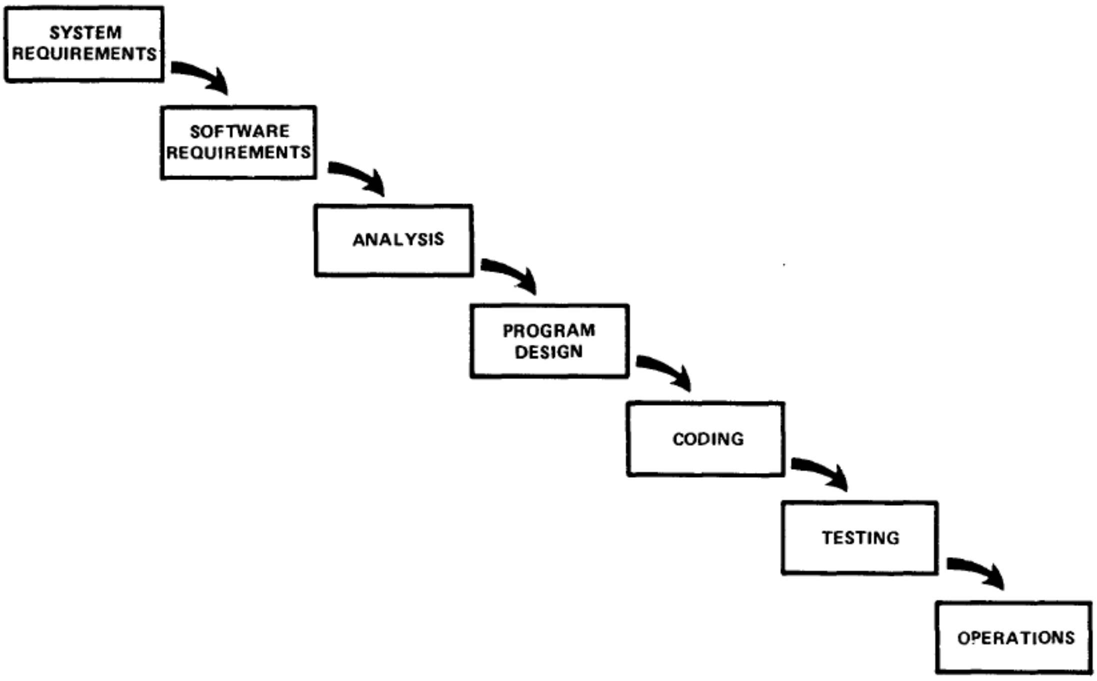


Figure 6.1: Royce's waterfall model reproduced from (Royce, 1970). This is held to be the original waterfall approach but it is a scheme which Royce's paper claims is flawed.

Royce in fact makes many suggestions such as having more documentation than you might expect, involving the customer, lots of testing and doing it twice Fig. 6.2. He points out the flaw is that it takes a long time before implementation starts and real problems are seen. These real problems may fundamentally change requirements. All of these ideas are in line with *Agile* approaches. Our intention is to do it far more than twice - about ten times for DPAC. Even the preliminary AGIS presented here (Section 6.6.7) went through several iterations before it was ready.

6.1.1 Approach to Change and Risk

The perhaps fundamental difference between science software development and satellite development is the risk element. Often for the software we may not know exactly how a particular problem will be solved or indeed if it may be solved at all. It would be a

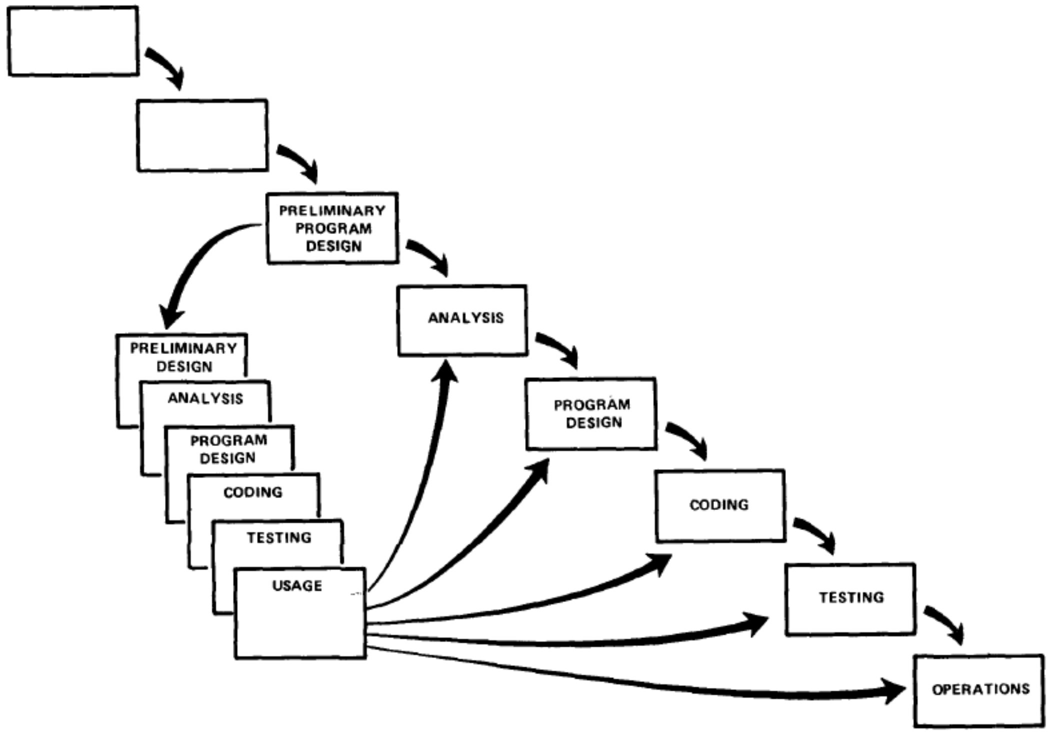


Figure 6.2: Royce's "do it twice" diagram reproduced from (Royce, 1970). Here he suggests that once through the waterfall is not enough - one should, at least, do it twice.

huge mistake to discard certain solutions simply to make a good plan (which is after all what most managers would like to see). Let us take some examples.

It is true certain things on the satellite have two solutions, the possible use of L3CCDs for the spectrometer on Gaia provide a good example. Here is a technology which would meet the science goals of the system but which was considered too immature at an early juncture and so ruled out. This is a typical risk averse style of project management and perhaps it is merited on a system which must be ready on time for launch. Much of the science community would have preferred a more risk embracing approach even if it possibly meant the spectrometer not working in flight. ESA in this respect is no different now to any other funding agency - no one funds true experimentation where there is a possibility of failure, failure simply can not be tolerated. In such a light of course one takes the safe route even at the expense of scientific return. Somehow the failure to meet the scientific requirements is seen as acceptable - the requirements are simply changed and the new easier requirements are met. Success all round it seems.

On the software front similar decisions need to be made. The exact position and velocity of Gaia are needed for the ephemeris reconstruction. Normal ranging from the ground stations is not sufficiently accurate for data processing needs. A proposed solution to this is to make ground based observations of Gaia on a daily basis. These observations can significantly improve the positional accuracy. So we have a plan, in a traditional approach we would look no further. In actual fact the Dresden group are pursuing a mathematical modeling approach to determine Gaia's velocity from the science data. The ground based observations would then be a less accurate check on their result.

On the software side even if we have a solution we must look for a better one. Practically all of the data treatment uses some form of approximation which may be improved upon. Even AGIS has a theoretically better solution which should be, and is, pursued at some level. Such parallel investigations which may fail are essential to get the best results from the Gaia data. On the hardware side parallel developments are avoided for cost reasons - on the software side effort must be spent on essentially risky possibilities.

6.2 Global solution background

Satellite programs are large engineering activities carried out in a very programmatic manner with several phases and checkpoints for feasibility of the project. The Gaia hardware was no different with many special demonstration projects for mirrors, CCDs etc. undertaken as a result of the initial feasibility studies. Perhaps unlike previous space science missions, in 1995 the project scientist felt that the software for the data processing itself was sufficiently complex to merit its own feasibility study.

In this section we look at some of the activities in the global solution field and also set the AGIS achievement in context looking at a little Gaia history. In particular we briefly describe GSR (Section 6.2.1), the Initial Hipparcos re-reduction (Section 6.2.2) and GDAAS(Section 6.2.3).

6.2.1 Global Sphere Reconstruction (GSR)

The notion of a global solution is mentioned in the Hipparcos documentation and some merit was given to pursuing this as a verification of AGIS in DPAC. The Torino group have been working on this since the 90's primarily from the perspective of relativistic effects as in (de Felice et al., 1998). Although this paper speaks of an iterative solution at the founding of DPAC they moved decidedly toward a direct solution i.e. with all unknowns solved in one step unlike the block iterative approach of AGIS. As pointed out by Bombrun (Bombrun et al., 2010) a direct solution such as this can not work for a large number of sources and the aim of Torino was to do this for perhaps one million sources. The current description is in a series of Technotes such as (AVE-002) for astrometry. Between 2006 and 2011 some form of GSR has been run about ten[2] times on the hardware at DPCT. Currently a *source* solution with 100,000 sources is claimed to work however the results have not been well demonstrated to the community. In spite of repeated claims the solution has proved to be elusive - one may note on page 115 of the DPAC response to the AO that GSR was almost finished in 2006 "Thus, a relatively sophisticated GSR model is already available for the verification of the AGIS solution," while such verification has still not taken place in 2011.

Clearly this is a most complex problem to solve and implement even for one million sources.

[2] private communication

6.2.2 Hipparcos re-reduction

As early as 1995 while still finishing Hipparcos the project scientist started to explore the feasibility of the Gaia data reduction. Even in Hipparcos the FAST consortium had the engineering expertise and leadership of CNES for data processing. Gaia would be even more complex. In 1997 the author came to assist in the final CDROM production for Hipparcos and soon after engaged in many conversations on the Gaia processing. It was clear the Global Iterative Solution was novel and required testing. An initial test was agreed with Lindegren and initially coded by the author over about two months in 1998 (O'Mullane & Lindegren, 1999). Some weeks were spent running this distributed system on colleagues' solaris machines while they were out of the office. The idea was to do a form of global iterative solution for chromaticity corrections in the Hipparcos data. The system oscillated around some solution which was not scientifically very interesting but did show that a database and distributed computing could be used for a block iterative solution. This led the way to GDAAS.

6.2.3 Gaia Data Access and Analysis Study (GDAAS)

GDAAS was a study commissioned by ESA in 2000 (GAIA-SOW-001)[3]. The purpose of the study was at least two fold. Primarily it was to establish the feasibility of the Global Iterative Solution and thereby show some feasibility for the entire Gaia data reduction. There was also a feeling that an engineering, rather than scientific, approach to this task might be more appropriate given its scale and complexity. There was no single group in the science community at that time with the discipline and expertise that might be needed.

A funding source within ESA was used to fund a study to turn the prototype Global Iterative Solution (O'Mullane & Lindegren, 1999), based on Hipparcos data, into something based on simulated Gaia data. The Gaia community were strongly behind this study. Lindegren was prepared to provide algorithms, ESA were to provide simulated data, others were ready to analyse results. Ultimately simulations were done by the University of Barcelona group who continue to successfully provide simulations for all of DPAC. (GAIA-SOW-001) asked for many other specific technical points, for example it already suggested Java and Objectivity be used (the prototype being in that combination) and requested some comparisons to C++ and endorsement or not of Java. It also contained lower priority tasks such as looking at SCOS2000[4] and First Look which were subsequently, and correctly, dropped during the study.

The winning bidder (a consortium consisting of GMV, CESCA and UB) proposed a great team with experience in Java and project management from GMV, while scientific expertise was to be provided by UB (Universitat Barcelona). The initial study period was for two years and many progress reports were produced, understandably positive while also mentioning many technical problems. Indeed for the first study the outcome

[3] This is written of with some authority as the statement of work was written by this author.

[4] Spacecraft Operations System - standard in the European Space Operations Centre.

was sufficiently convincing to continue with a second study.

In 2002 a final report (GMV-GDAAS-RP-001) was delivered claiming success e.g. page 57 “system is working, albeit with certain limitations” although no actual converged solution had been achieved for the six month dataset. In the contract GMV had managerial control and were to provide the rigorous engineering and project management support supposed not to be generally available in an academic environment. Somehow, between GMV and ESA, there was a lack of appreciation of the gravity of problems reported by GMV and accumulated by GDAAS. This is not strange in project management, people involved in the project downplay problems, the customer and provider both want the project to succeed after all. Harrison and Associates in management training seminars use the analogy of boiling a frog: if you place the frog comfortably in a pot of water and slowly heat it he may stay until he is actually cooked, if you throw him in hot water he will immediately jump out. The slow accumulation of small problems can be potentially catastrophic.

With the proposed launch date nearer by two years, and the feasibility of the Global Iterative Solution still unproven, this remained a critical task for the project and there was still no group in the community with the capability to demonstrate the feasibility of Gaia data processing. It was decided to give GDAAS a further try with another contract. However things got more complex. For example early pipeline processing (something like today’s IDT) was included in the statement of work as well as many other points of interest. This meant the main goal of showing the astrometric processing feasibility seems to have been slightly overcrowded. During the project several major changes in the Gaia design occurred and were included in the design of the processing system and simulations. After two more years the system performed in a very poor manner with a time close to infinity for one iteration.

In 2004, the Project Scientist was sufficiently concerned about the lack of progress of GDAAS2 and instigated an independent review. This was conducted over a three week period during which O’Mullane[5] and Lammers[6] co-located with the GDAAS team in Barcelona to study the code and the system. Bastian[7] joined for one week to look specifically at astrophysical considerations. UB and GMV provided complete access to their systems and code allowing detailed investigation to be done. The resulting report (Lammers et al., 2004) pointed out many potential problems in the GDAAS2 system. It was too complex using many *nice* but impractical computer science ideas and having serious I/O problems. The wrapping of Fortran in C to be called by JNI was also cumbersome. The report provided several pointers on what to change and how to change it to improve performance. This report was used by the ESAC team, subsequently established by the Project Scientist to take full responsibility for the core data processing, as a starting point for their work.

Later in 2005 the ESAC team commenced work and provided a convergent solution

[5] O’Mullane at this point in time was at JHU working on SDSS but with prior knowledge of the Global solution and a member of the Gaia data processing working group.

[6] Lammers at this time was in the Space Science Department in ESTEC working mainly on XMM but partially on Gaia.

[7] Bastian at this time was in the Gaia astrometry working group and also a leading figure in DIVA, a German Astrometric satellite which was later cancelled. He was also deeply involved in Hipparcos.

(from thence called AGIS) within just four months.

It should be noted that at this point the UB team also gained some success running seven iterations of the system but not achieving convergence (UB-GDAAS2-TN-035) nor the performance of the ESAC system. This later work was funded by UB beyond the scope of the GDAAS2 contract and without GMV management. Another positive outcome of GDAAS was the rather complete Gaia simulations which, in augmented form continue to be used today.

6.2.3.1 Lessons learned

One must not forget the fact that the global solution is a difficult task and probably only achievable if almost all of the project is carried out correctly. UB did a good job understanding the global solution and getting GMV information to implement algorithms. Still there were some obvious weaknesses in the project as a whole.

From an academic perspective, much in the vein of the Edison quote in Chapter 5, GDAAS may be seen as a success in highlighting many problems which could then be avoided in later developments. From an industrial contract perspective it was less successful.

Already at the kick off meeting GMV suggested removing the experienced project manager. Although Java experience was scarce at that time GMV put people on the proposal with Java experience. After some time the experienced Java programmers were essentially moved off leaving one main programmer who at the time had little Java experience.

On the ESA side it was difficult to find a suitable technical officer who truly understood the problem in hand and the status of the project as it progressed.

Given the outcome on GDAAS1 it was probably not a good idea to significantly increase the scope for GDAAS2. Even the scope of the original study was a little wide.

On a more technical note dropping the Fortran and JNI and going for pure Java against some opposition was a good lesson taken from GDAAS and probably aided in the adoption of Java in DPAC later.

It is clear that many technical companies work on the idea of interchangeable resources. However to make something like AGIS work, one must spend time understanding the intricacies of the system. It requires dedication, continuity and fruitful interaction with the community. It is a common misconception on projects that one hundred people do the job faster than one or two. This may be true for basic work but clearly for a complex job you need the correct people to do the job.

By early 2006 it was clear in the Gaia community that AGIS would reside with the ESAC team under the scientific and algorithmic guidance of Lindegren. It was set up as such in the DPAC responses to the Gaia Data Processing Announcement of Opportunity (FM-030). GDAAS work was wound down.

6.2.4 Nano Jasmine

Just as a final note the Jasmine group in Japan will attempt to fly a small Hipparcos like satellite named Nano Jasmine. It has a modest goal of producing a Gaia like catalogue but at Hipparcos accuracy. Since this is a TDI scanning instrument with no input catalogue the Japanese group sought help from DPAC for their astrometric solution. ESAC and Lund are assisting them with applying AGIS to their data. The fact that the Japanese group sought the assistance of the Gaia AGIS team is a further indication of the true complexity involved in implementing the Global Iterative Solution.

6.3 Clients, requirements and teams

In many projects it is conceivable that an individual working alone may implement an entire system. It is equally clear in a project the size of Gaia's DPAC that teams of people will have to work closely together to achieve the final result. Decomposition to sub-systems will of course help and the CU/DU decomposition of DPAC is the first step in this direction. Such a decomposition still leads to relatively large sub-systems and also to an integration problem. At some point a team is needed to achieve the task in hand.

When putting a group of individuals in a team we lose effort. By this we mean that what an individual may do in a week is not necessarily done by a team of five in one day. This is clear to anyone who has worked in a team. What we must try to do is reduce the team overhead and try to get our team output as close as possible to optimum. This requires clear goals and a common understanding of how to achieve those goals. Goals must be clearly agreed with the client, in the initial phase of AGIS development this was the Project Scientist, in advance of any project preferably in the form of a requirements document. Within DPAC now we see System Requirements Specifications (SRSs) for all DPAC subsystems. Without an original goal a team may be in an endless loop of developing to a moving target. This would mean the team never achieves anything in both their own eyes and those of the client. Such a situation must be avoided.

The client issue for DPAC is rather tricky. Officially ESA is the client in that DPAC agrees to produce a Star catalogue for ESA in 2020. From the point of view of the software required to do the processing DPAC is effectively its own client. Hence the use of "System Requirements Specifications" (SRS) from ECSS (ECSS-E-40-1B) rather then "User Requirements Specifications" which would imply more something written by a client. The SRS allows requirements to be mixed with more system design oriented material which is frowned upon in a user requirements document. In one sense this is a huge advantage of ECSS over the space agency's older PSS05 standard which it replaces. The System Requirements and Specification (SSS) document (WOM-018) provides top level requirements and ties all individual SRSs together.

Achieving a goal requires careful planning and tracking of the team performance. Time and intermediate milestones must be carefully planned and tracked in order to know if the desired result will be achieved within the desired time. If it seems impossible to achieve the desired result in time, immediate action must be taken. This may be de-scoping the milestone or increasing manpower. Increasing manpower introduces its own problems, certainly more than doubling a team in over a short period will introduce problems. Doubling over a few months should be possible if carefully planned. Agile techniques such as eXtreme Programming (XP) (Beck, 1999) attempt to give better planning and feed back on progress for projects. The use of these techniques by the AGIS team are described below.

6.4 Risk

To paraphrase Jim Gray [8] there are only two things to fear: the first is failure, the second is success. A successful software project will tend to see the software used more widely and creatively than anticipated in the original design. Furthermore success leads to higher expectations from the client. In many cases risk is avoided to achieve a goal e.g., the risk of a radical change in practise or design. It is difficult to decide when a project is in such a state that a major risk should be taken. Often people ask if the change will make the system work - perhaps the correct question is rather, is it more likely to fail than staying as we are? We need to embrace change and learn to deal with it in a good way. Indeed "embrace change" is one of th tenets of eXtreme programming (Beck, 1999).

6.5 The initial conditions

In September 2005 the initial ESAC team of O'Mullane, Lammers and Hoar (part time) was formed. The ultimate goal of the ESAC team is to manage, develop where needed, and operate all Science Operations Centre software within DPAC. The initial goal, however, was to get AGIS working by the end of the year. Perryman set a clear goal here - the system was to complete one outer iteration in under three weeks on the machine installed at ESAC. There was no indication that such a goal was achievable (see Section 6.2.3) but all agreed that if we could not make progress with the Global Solution serious questions would be asked about the overall feasibility of the Gaia processing. Without the core astrometric solution Gaia would not achieve its science goals and as such could have been dropped by ESA. Indeed there were many high level meetings in

[8] Jim was a Microsoft fellow and a founding designer of SQL during his time at IBM. Sadly he was lost at sea on January 28^{th}2007 the resulting search showed how well liked and influential he was `http://www.theregister.co.uk/2007/04/30/jim_gray_tribute/`

ESA concerning the feasibility of the astrometric processing in 2006, fortunately AGIS results were available to calm fears in this area.

6.6 General approach to the development

The first two weeks were spent in setting up machines and installing development tools such as Eclipse (`http://www.eclipse.org/`). The first major decision was to stick with Java, at least initially. During this period the GDAAS code was revisited and a new Architecture (see Section 3) based on the Data Train crystallised. The second major decision was to use this new architecture and not to use any GDAAS infrastructure code. This meant taking and modifying only algorithm code from GDAAS. Many years had been put into the GDAAS infrastructure and we had only a few months to produce a better system, however the team agreed the risk was merited. We also felt it would not be possible to improve the existing system sufficiently in the time allocated i.e. the radical option was felt to have a slightly better chance of success. An overall Use Case for the final system was drafted as a goal document. This document defined the bounds of the system e.g. to only deal with AGIS not IDT and data loading.

6.6.1 Agile Techniques

Many IT companies are now employing so called ‘Agile’ techniques for team planning and programming. The notion behind these ideas is to make the team more responsive to change and to problems. A rapid development is presumed as are poorly defined requirements. Rather than well defined requirements the client is supposed to be involved in the definition of tests often extensions of use cases. Such approaches have existed in IT in many guises since the beginning (Larman & Basili, 2003). Approaches such as Rational Unified Process (RUP) (Kruchten, 2003) follow some of the ideas presented here. RUP is the commercial realisation by Rational of the unified design methodologies of some of the great gurus of software design who finally gave up arguing about which way was correct and settled on the *Unified Approach* (Jacobson et al., 1999). The Unified Modelling Language (UML) (Booch et al., 2005) also springs from this collaboration and is now the defacto standard for diagramming in object oriented systems.

In the second week of September 2005 the notion of using eXtreme programming (Beck, 1999) to organise the effort was discussed. The team were fairly open to this idea. The facets employed are described in this section. Employment of a radical technique was seen as a risk however no better alternative was on offer - a traditional waterfall approach to AGIS would not work in the four month timescale allocated, a complete free for all would also be too risky.

6.6.2 Collaborative development

The Concurrent Versioning System (CVS) was set up to hold the source code. The repository was later replaced by the more sophisticated Subversion (SVN) system. In principle any team member is allowed to modify any code which he feels is incorrect. Also if a system wide change is planned nominally one person does this for all code or a small group do it in unison while in the same office. This is quite different to traditional developments where one team or individual may decide to change an interface and then tell others they must fix their code to work with the new interface. Code should be owned by the team not by individuals. Less up front design work is done in the eXtreme approach but "pair programming" is encouraged. Hence for difficult or novel tasks two team members will sit together to write the initial code - this again leads to better team ownership of the code. Regular reviews of code should be organised about monthly. The ESAC team now also organise large active code reviews where, for example, all of the GaiaTools code is split between a team sitting in a meeting room. Each class is examined, comments and coding standards compliance are checked as well as readability. In these reviews, however, rather than just telling someone this is wrong small fixes are done on the spot by the reviewers. Quick discussions may be held for any changes a reviewer is not sure about. This sort of event leads to homogeneous code as well as more homogeneous developers.

6.6.3 Testing and continuous integration

Cornerstones of eXtreme Programming are unit tests and continuous integration. It was agreed to install Cruise Control and to employ JUnit for testing. The notion here is that extensive tests should exist for each class, if something is changed and the tests pass the team has confidence that the system should still work. Hence fear of change is reduced. Ideally developers run all tests before checking their changes in. If there are merge conflicts the developer must also fix these. This is already a good start but it is possible for a developer to run all tests successfully and check in code and still cause a system breakage. Cruise Control periodically checks out all code, builds it and runs all of the tests. A second set of longer running "Integration Tests" are also run by Cruise Control, the developer typically does not run these tests. In 2008 Cruise Control was replaced in ESAC by Hudson.

Even if the system is not running it should always build - this is the responsibility of every team member: *Do not break the build.* If a developer inadvertently breaks the build they are supposed to fix the problem as quickly as possible.

6.6.4 Iterations and Stories

The notion of an iteration is to have a working system in a short period of time - the final target is reached through a series of iterations. The system at the end of each iteration does not have to be a final system but it should compile, pass all tests and show some more features than the previous iteration. Iterations are composed of stories. A story is normally a short description of something the system should do - it is a form of Use Case and should ideally be testable. In the ESAC case we include other stories in our iterations such as writing documents and setting up systems.

This iterative approach has been expanded and incorporated in to the DPAC development approach where ten *cycles* of six months have been defined for the entire DPAC development. Here each *cycle* is similar in idea to an XP iteration. This should not be confused with an AGIS cycle which is an attempt to run the AGIS software to convergence. This approached was specified in (WOM-001) which is a DPACE approved document. In the fun loving nature of XP, these cycles were given the names of the ten highest mountains in the world.

6.6.5 Iteration planning and costing - points

At ESAC each iteration starts with a planning meeting. In this meeting the start date (usually the next day) and end date of the iteration are agreed. At least one "big story" is agreed e.g. *Get get the Data Train infrastructure running*. Next the whole team breaks this down to smaller stories which should be less that a few days work. These are written on post-it notes and stuck on a whiteboard [9]. At this stage stories are grouped mainly according to priority - what we must do (top of the board)- what may be done later (bottom of the board). The initial AGIS iterations are briefly described below in Section 6.6.7.

With a bunch of stories and a few developers how do we plan the work? Obviously we need to cost stories. The approach we have taken here is exactly the XP approach. The idea of a point is introduced, a point is a unit of work. It should not be considered to exactly coincide with any time period but at ESAC a point may be considered to be about half a day's effort. The entire team then costs each of the stories in terms of points. If a story is considered to be ten points it is broken down to stories of five points or less. Ten points indicates too great an uncertainty on a short planning period. The important thing here is that a manager is not saying how long it will take to do something but the team collectively decides. Now we have stories with points, or costs, on them.

Next, points are allocated to developers. When points are allocated for an iteration several factors need to be considered. First the actual number of available days in the iteration are considered.

Next for each developer any leave or meetings must be discounted from the remaining days. So finally we have a number of work days in the iteration. Using our half-day-to-a-point rule we multiply this by two. This is the maximum this developer could do in the iteration. As iterations continue we may calibrate this number against what a developer actually does i.e. some developers may do more than the theoretical maximum. So this number needs to be tailored over time - new members may get less points for

[9] Here we depart from XP which uses index cards as they are more durable and they should be pinned to a board. These days we do it on a projector with excel

example. Furthermore there is supposed to be some collaborative programming - in XP this number should be halved immediately as pair programming is to be the norm. In the ESAC case we discount this by about thirty percent.

With the points allocated to each developer we know how much work we may do in the iteration. At this point a quick sanity check must be done to see if the points on priority stories exceed the available manpower and priorities need to be adjusted accordingly. Now each developer is allowed to "buy" stories with the allocated points until all points are used up. Leaving the room at this point is fatal as one ends up with all the worst stories to do. With a short iteration period we may quickly asses our planning skills and adjust the meaning of a point accordingly. It is for this reason we chose only two weeks for our first iteration. We wished to try out the technique and asses our ability to get a working system in place in the agreed time frame. Four weeks seems about the longest an iteration should be. Bear in mind iteration end means 'working system'. The principle here is always working to working - we should not break the system.

Another advantage of short iterations is that a ten or even twenty percent overrun equates to only a few days in real terms. In our planning we do not have iterations back to back rather we have a few days unplanned between iterations. This allows a breathing space to catch up on other work, run tests and investigate issues on an individual basis. This unplanned time has been very good for the ESAC team.

6.6.6 XP Tracker

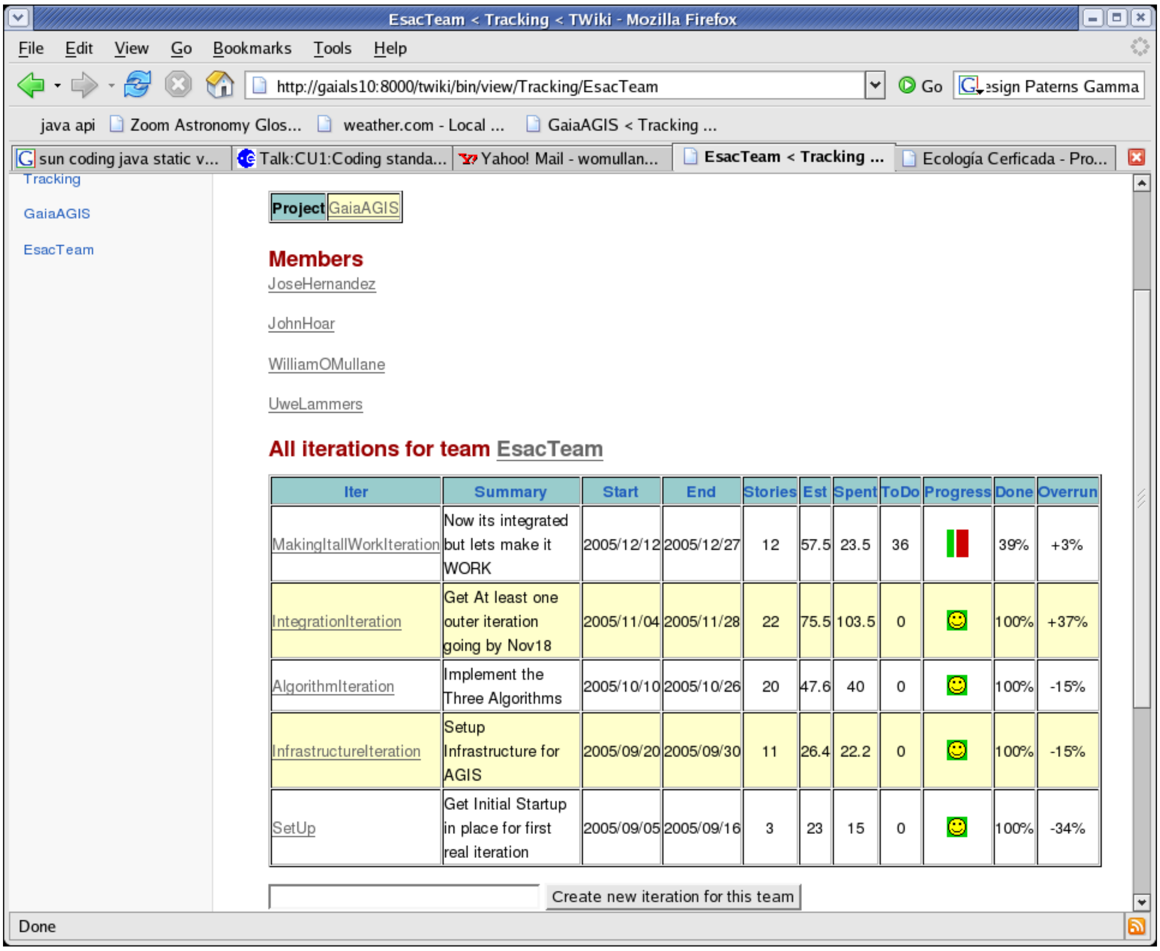
At ESAC we have used the XP tracker plug-in for Twiki. This is a web based tool allowing iterations, stories and tasks to be entered easily. Each developer enters the stories which they bought (hence the post-it is no longer important) and creates one or more tasks in the story with the effort estimate (in points) associated with it. The system then allows effort to be entered against each task as it progresses and generates summary pages showing the percentage done at the present time. Here again rather than a manager making a 'Plan' and trying to track it, the team maintains individual tasks online and the status may be viewed at anytime. The team found this a little strange in the beginning but quickly got used to the ability to see how we were doing in terms of the iteration. Ideally team members update their tasks on at least a daily basis. Traditional Work Package numbers may also be entered in the FEA field of the story - a script is then used to produce effort per work package reports.

There are several quirks of XP tracker which make it inaccurate but since it is all only an indication based on passed experience this is OK. However the overrun numbers in Fig. 6.3 are not accurate (they are too low)[10].

[10] In 2011 ESAC switched to a Google Spreadsheet created by Gonzalo Gracia, this option was previously explored by Hassan Siddiqui in 2009 but not pursued at that time.

6.6.7 The AGIS iterations

The XP tracker gives a nice overview of the iterations which is reproduced here in Fig. 6.3.


EsacTeam < Tracking < TWiki - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://gaials10:8000/twiki/bin/view/Tracking/EsacTeam
Go
java api
Zoom Astronomy Glos...
weather.com - Local ...
GaiaAGIS < Tracking ...
sun coding java static v...
Talk:CU1:Coding standa...
Yahoo! Mail - womullan...
EsacTeam < Tracking ...
Ecología Cerficada - Pro...
Tracking
GaiaAGIS
EsacTeam
Done

Project GaiaAGIS

Members

JoseHernandez

JohnHoar

WilliamOMullane

UweLammers

All iterations for team EsacTeam

Iter	Summary	Start	End	Stories	Est	Spent	ToDo	Progress	Done	Overrun
MakingItallWorkIteration	Now its integrated but lets make it WORK	2005/12/12	2005/12/27	12	57.5	23.5	36	🟩🟥	39%	+3%
IntegrationIteration	Get At least one outer iteration going by Nov18	2005/11/04	2005/11/28	22	75.5	103.5	0	☺	100%	+37%
AlgorithmIteration	Implement the Three Algorithms	2005/10/10	2005/10/26	20	47.6	40	0	☺	100%	-15%
InfrastructureIteration	Setup Infrastructure for AGIS	2005/09/20	2005/09/30	11	26.4	22.2	0	☺	100%	-15%
SetUp	Get Initial Startup in place for first real iteration	2005/09/05	2005/09/16	3	23	15	0	☺	100%	-34%

Create new iteration for this team

Figure 6.3: The iterations for the ESAC team are shown with the most recent on top. Start and end dates are provided with a short textual summary. This is from February 2006 there are somewhat more iterations now.

If we total up the points from the AGIS iterations we get 204 points. Using our rough guide to points this is 102 days of effort. Points are only a rough planning tool however. If we take the available days in the period of time from the developers involved this is maximum 225 days. If we take out meetings and other work 180 days is probably a more realistic cost for the system. This is not a realistic cost for a polished system - several corners were cut (see section 6.6.8) and no standards were adhered to.

The following sections briefly describe each of the iterations.

6.6.7.1 Set Up Iteration

The first two weeks of loosely planed work was retrospectively put in the XP tracker as a demonstration that the system worked. During this period Eclipse was selected for development, XPtracker, Cruise Control, ant etc. were installed. The existing GDAAS code was looked over by the team to decide how best to utilise it. The first proper iteration was the Infrastructure Iteration.

6.6.7.2 Infrastructure Iteration

For the first ESAC iteration, called Infrastructure Iteration, a two week iteration was chosen. The end date was chosen to coincide with the first visit of the Project Scientist (our client).

The principle of the Infrastructure Iteration was to show all elementaries of the one million sources could be read in a reasonable time. There was no intention to do anything with them in principle, just see the data delivered by the DataTrain. See (WOM-004) for complete information or Section 3 for an overview.

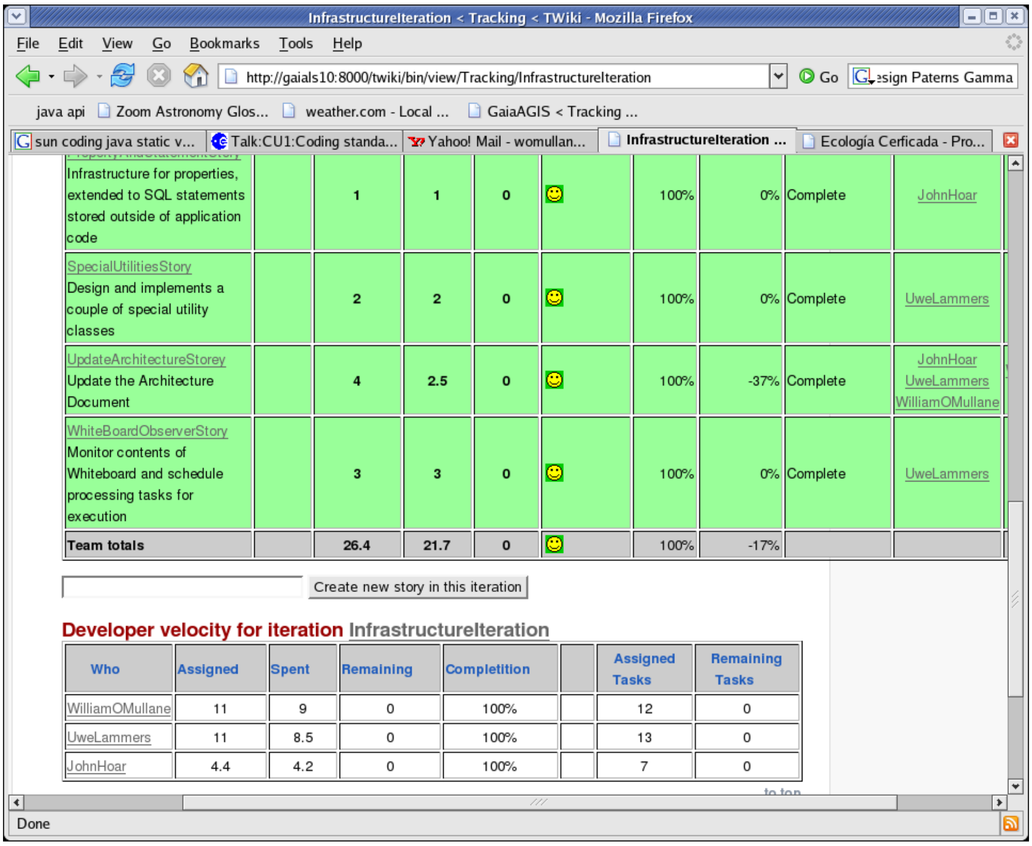
There were three developers in this iteration and only one (O'Mullane) had experience of the XP approach. The XP tracker allows a summary of the iteration to be displayed, it is too large for one page but the end of the summary is shown in figure Fig. 6.4.

The iteration goal of a running DataTrain was achieved on schedule. Figure 6.4 shows a 15% overestimate on this iteration which is not really true. Generally more was achieved in a day than two points although it was still recorded as two points. Partially more than two points were achieved by working longer hours - this is not the XP way. The notion of making GDAAS work on our system was dropped which saved us some points. Figure 6.4 also shows that none of the developers achieved the full allocation of work - this was mainly due to other issues needing to be dealt with.

At the end of this iteration the DataTrain was able to read all Astro Elementaries in 27 minutes. The use of small stories and a short iteration gave the team a good feeling of achievement in a short period of time. Furthermore the principle of having a demonstrably working system at the end of the iteration enabled the team to show the client a good deal of progress at the first meeting. Most importantly a plan was followed and the objective achieved on schedule. It was very important for the team to learn their limitations, since in general people tend to underestimate tasks.

6.6.7.3 Algorithm Iteration

The two week iteration had gone well and for this iteration a small time frame was also chosen, this time about three weeks so the end would coincide with a visit by Lindegren.


InfrastructureIteration < Tracking < TWiki - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://gaials10:8000/twiki/bin/view/Tracking/InfrastructureIteration
Go
esign Paterns Gamma
java api
Zoom Astronomy Glos...
weather.com - Local ...
GaiaAGIS < Tracking ...
sun coding java static v...
Talk:CU1:Coding standa...
Yahoo! Mail - womullan...
InfrastructureIteration ...
Ecología Cerficada - Pro...
Done

Infrastructure for properties, extended to SQL statements stored outside of application code		**1**	**1**	**0**	☺	100%	0%	Complete	JohnHoar
SpecialUtilitiesStory Design and implements a couple of special utility classes		**2**	**2**	**0**	☺	100%	0%	Complete	UweLammers
UpdateArchitectureStorey Update the Architecture Document		**4**	**2.5**	**0**	☺	100%	-37%	Complete	JohnHoar UweLammers WilliamOMullane
WhiteBoardObserverStory Monitor contents of Whiteboard and schedule processing tasks for execution		**3**	**3**	**0**	☺	100%	0%	Complete	UweLammers
Team totals		**26.4**	**21.7**	**0**	☺	100%	-17%		

Create new story in this iteration

Developer velocity for iteration InfrastructureIteration

Who	Assigned	Spent	Remaining	Completition		Assigned Tasks	Remaining Tasks
WilliamOMullane	11	9	0	100%		12	0
UweLammers	11	8.5	0	100%		13	0
JohnHoar	4.4	4.2	0	100%		7	0

to top

Figure 6.4: The Infrastructure Iteration

We had a Data Train running, so this iteration was concerned with attaching algorithms to the train. The decision was to concentrate on the Source Update. This was considered to be the most difficult and time consuming of the algorithms. Achieving this Story required considerable work from the entire team. At this point we also accreted a new team member (Hernandez).

The new team member was introduces to the XP approach by including him in the planning meeting at the beginning of the iteration. Hernandez came at 50% and had his points further discounted because he was new to the team. He was already well acquainted with the principles of AGIS. In true XP style the new team member was writing part of the system on the first day of the iteration and achieved far more than the allocated points over the iteration.

Some tasks were slightly underestimated while others were over estimated bringing the team in only slightly ahead of the overall estimate. The iteration was finished on time for the visitor. Source updating could be performed on half a million sources in about five hours. It was clear from the previous iteration data access was about 30 minutes, furthermore processors were seen to be running to over 80%.

Some collaborative programming was indulged in with our visitor also. Lindegren spent an afternoon with Lammers looking at numbers flowing though the system. Several bugs in the Source Update were discovered in this session which further reduced the time for an iteration and saw sources converge. Because of the agile techniques it was possible to include Lindegren's changes in the system and run an iteration quickly enough to send him away with the first few thousand updated sources. This is precisely how XP should work.

At this point the team adopted the mantra "AGIS in a day".

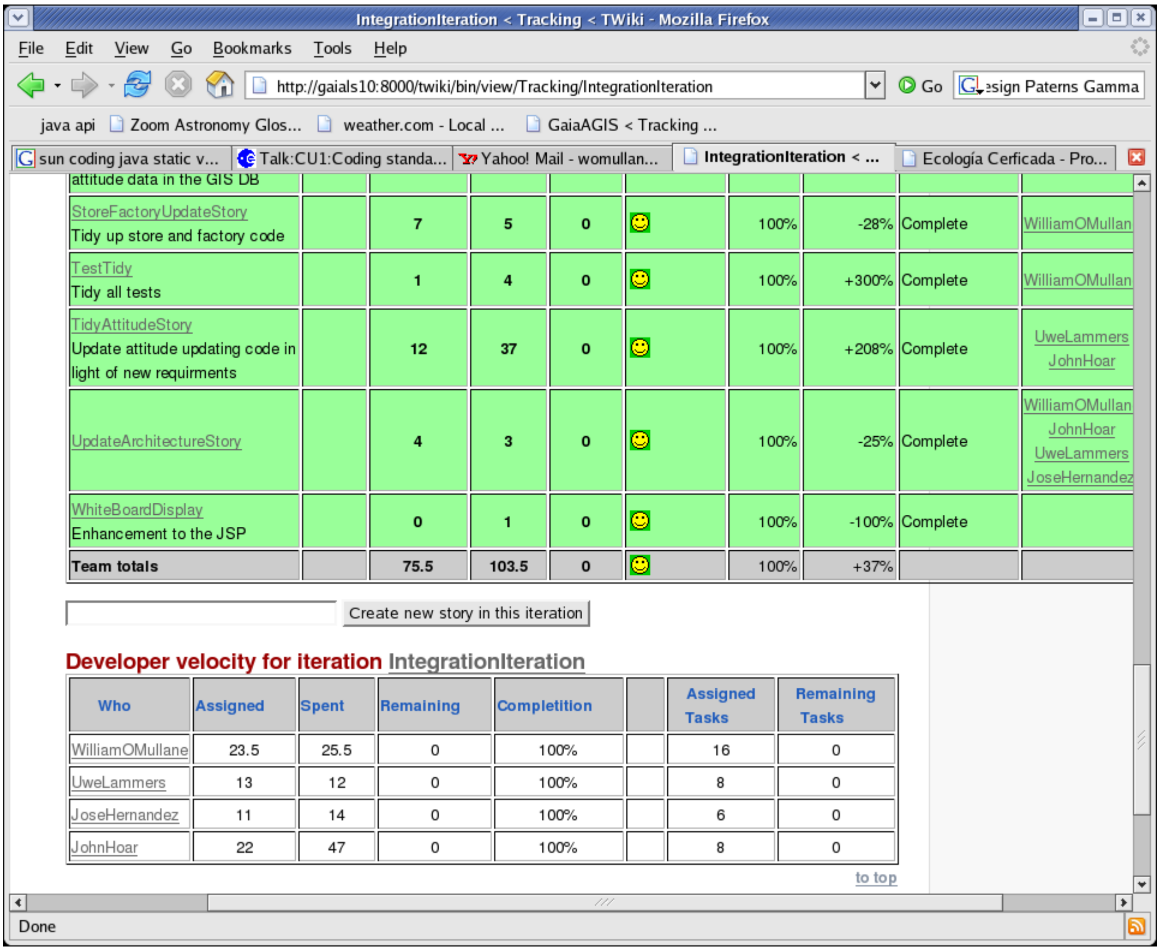
6.6.7.4 Integration Iteration

With Source Update running in record time the team was upbeat and the client was happy. The next iteration was to integrate the Calibration and Attitude Updates. Global had been de-scoped from the outset and was finally done in 2007.

Here there were two tasks which were similar in infrastructure requirements namely: Calibration and Attitude updating. Although a common system could have been designed for this it was rather decided to let the two developers do each in his own way. Later the results were merged. Again with short iterations and stories it is ok to have parallel effort like this - it doubles the chance of getting something working. In any case the Calibration was started after Attitude and did indeed use some of the Attitude code.

Attitude turned out to be the most demanding of the algorithms in the end. As with source updating, code was taken from GDAAS and some bugs were fixed. Lindegren (remotely this time) looked over some of the attitude code and found several more bugs. However it remained incredibly slow.

At the end of this iteration the working system included Source, Calibration and Attitude updating. Again there was some overrun particularly in attitude as there were problems. Fig. 6.5 shows a 37% underestimate of the iteration in XPtracker. This is not accurate, i.e. the underestimate was actually larger. The short iteration helps with this as in the next iteration more time could be allocated to the problem area.


IntegrationIteration < Tracking < TWiki - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://gaials10:8000/twiki/bin/view/Tracking/IntegrationIteration
Go
esign Paterns Gamma
java api
Zoom Astronomy Glos...
weather.com - Local ...
GaiaAGIS < Tracking ...
sun coding java static v...
Talk:CU1:Coding standa...
Yahoo! Mail - womullan...
IntegrationIteration < ...
Ecología Cerficada - Pro...
Create new story in this iteration
to top
Done

attitude data in the GIS DB									
StoreFactoryUpdateStory Tidy up store and factory code		7	5	0	☺	100%	-28%	Complete	WilliamOMullan
TestTidy Tidy all tests		1	4	0	☺	100%	+300%	Complete	WilliamOMullan
TidyAttitudeStory Update attitude updating code in light of new requirments		12	37	0	☺	100%	+208%	Complete	UweLammers JohnHoar
UpdateArchitectureStory		4	3	0	☺	100%	-25%	Complete	WilliamOMullan JohnHoar UweLammers JoseHernandez
WhiteBoardDisplay Enhancement to the JSP		0	1	0	☺	100%	-100%	Complete	
Team totals		75.5	103.5	0	☺	100%	+37%		

Developer velocity for iteration IntegrationIteration

Who	Assigned	Spent	Remaining	Completition		Assigned Tasks	Remaining Tasks
WilliamOMullane	23.5	25.5	0	100%		16	0
UweLammers	13	12	0	100%		8	0
JoseHernandez	11	14	0	100%		6	0
JohnHoar	22	47	0	100%		8	0

Figure 6.5: The Integration Iteration

6.6.7.5 Making it all work Iteration

The team was hoping to have the final system at the end of the Integration iteration however the overrun ruled that out. In XP fashion, the iteration was not extended in this case, rather another iteration with a new goal was started. It is important to refocus when a goal is not met rather than blindly fight forward toward the missed goal. Everything was running but there was no convergence monitor and it was not clear the system did what it was supposed to do. This iteration was to produce some convincing output from the system to show it really worked - iron out the kinks and write some documentation. A process of investigative programming found a bug in the Sun JDK which caused the attitude server to be ten times slower than it should be. A patch was available. More interaction with Lindegren found further bugs in the constants - the result was indeed a convergent system running in a remarkable three hours per iteration (UL-015).

6.6.8 What should be different

The original team was too small and it was impossible to get new team members on board as quickly as planned. This led to some corner cutting in the initial months which have been ironed out. It remains difficult to add to a carefully balanced team and the ESAC team remained understaffed compared to plans until 2008.

The code had not been reviewed as often as it might have been. The commenting had not been checked as rigorously as it should - the Java Docs had noticeable holes in them. These things have been remedied in the time since the “making it all work” iteration. Code reviews are now more frequent and rigorous. Mantis is used to raise issues about insufficient tests and documentation.

The team would have liked a little more time for code consolidation - this was done in some cases e.g. merger of Calibration and Attitude Server code, and not in others e.g. database writing. Again this has been subsequently handled with part of the RMI server even going into GaiaTools, database writing was put into the store in 2007 and all included in GaiaTools 4.0.

There was no detailed design documentation for the system. This is not simply to pay service to ECSS but genuinely for maintenance and long term understanding of the system it is necessary. Subsequently the Software Design Description (WOM-004) was issued and reissued, it is a valuable document for anyone looking at the AGIS code.

The hours worked by the team had been excessive in the first six months which is alright for a short challenge. For a long project, Gaia runs to 2020 so we are speaking of 15 years, and in accordance with XP, hours should not be excessive for the most part. Of course it is a difficult balance to achieve sufficient goals while not over working - XP relies a lot on self motivated team members.

6.7 Conclusion

The original proposition of the AGIS challenge to the newly formed ESAC team was well met. The employment of novel coding and planning techniques helped the team achieve their objective. In addition to achieving this important milestone the team as a whole gained a good understanding of how the AGIS system should work.

The short planning cycles of the XP approach seem to be ideal for the scientific programming we shall undertake in Gaia. It is clear however, to employ such techniques the team leader needs to be focused and believe the technique works. Each team would need to find their own way of working in this manner. However it is done, the principle of an evolving system which is always going from ‘working’ to ‘working better’, is a good approach for DPAC.

The risk of failure of the Gaia DPAC in the coming years is far too great to allow long planning cycles. In essence the development of science software does not always lend itself to traditional waterfall approaches to software design. The iterative approach of XP (Larman & Basili, 2003) with a lot of input from the scientists is clearly a better approach.

Six monthly cycles were proposed to DPAC. Within that cycle at ESAC monthly, or shorter than monthly, internal XP style planning is performed. Other groups will do this in their own way. For a large DPAC planning cycles of less than six months would seem too short while anything longer would allow almost a year to pass without a checkpoint to return to if something goes wrong. These ideas have been expressed in (WOM-001) and the six month cycles have been adopted by DPAC.

In the following year the AGIS added far more complexity and better monitoring while maintaining a working system at all times. The ESAC team, and we believe DPAC, is now convinced that AGIS will indeed work for the mission. Unfortunately Perryman left the project in 2006 which was a blow to the team in terms of loosing leadership.

Chapter 7

Conclusion

The Astrometric Global Iterative Solution (AGIS) is a key part of the Gaia data processing. In its essence it may be represented by a few equations and yet it has taken many years to see an operational software system which has been able to solve these equations in a reasonable time. Making a functional AGIS has been one of the problems in astronomy which requires a good understanding of the science as well as the computing to achieve the science. Concentrating purely on the computing aspect and trying to fit AGIS into some existing framework did not prove useful. Rather a simple but powerful framework tailored to the AGIS problem has been built.

The initial chapters of this document attempt to show the depth of understanding of the AGIS problem required to make an efficient AGIS implementation. The system as it stands was presented in Section 3. Building such a system is not the work of one person but rather a team. The aspects of team building required to put such a group together have not been presented in detail here but perhaps merit a volume of their own. Needless to say constructing the team and directing them to construct the system is also a non-trivial task. The *Agile* approach used for AGIS is described in Section 6. This approach has proved very successful and has been extended toward the DPAC in general. The author then has not written every line of code in the implementation but has certainly influenced every line of code written for AGIS and many lines of code throughout the consortium.

For the author the journey to a working AGIS commenced in 1998 with first contact with the notion of a block iterative solution for astronomical data and first contact with the Gaia Science Team and in particular Dr. Lindegren. Reciprocally modern programming constructs such as the object oriented domain and the Java language were introduced to the Science Team. It was not until 2005 that a concerted effort by the author was put into what was then, and remains now, named AGIS. This Astrometric Global Iterative Solution, as crystallised in 2005 and in development ever since has been presented here.

Another system may replace AGIS in the long run up to 2021 and the final catalogue production. The work on the AGIS framework has now influenced many of the other Gaia processing systems. Currently Variability, Initial Data Treatment and First Look use a modified version of the AGIS framework for their tasks. Should AGIS be replaced, which is quite unlikely, any system coming after it would already benefit from the framework put in place for AGIS, for in any case all observations must be treated in a global manner to achieve the absolute astrometry required by Gaia.

It is not the end of the journey for AGIS but it is certainly a short moment of respite. It has been demonstrated that AGIS is capable of reducing Gaia like observations to provide the reference frame for the Gaia data. A novel approach has been employed in the solution of a unique problem, the authors contributions to which have been presented here. It has been equally difficult and rewarding, the Gaia flight data is eagerly awaited!

Appendix A

Quaternion Primer

> And how the one of Time, of Space the Three,
> Might in the Chain of Symbols girdled be.
>
> **William Hamilton Rowan to John Herschel regarding Quaternions (Hankins, 1980).**

When Hamilton presented quaternions to the Royal Irish Academy in 1843 it was with no little shock to the community. It was, as we would say today "a bolt from the blue" for Hamilton as he himself wrote:

> I then and there felt the galvanic circuit of thought close; and the sparks which fell from it were the *fundamental equations between i,j,k; exactly such* as I have used them ever since. I pulled out on the spot a pocket-book, which still exists, and made an entry, on which, *at that very moment*. I felt that it might be worth my while to expend the labour of at least ten (or it might be fifteen) years to come. But then it is fair to say that this is because I felt a *problem* to have been at that moment *solved* - an intellectual want relieved -which had haunted me for at least *fifteen years before*.
>
> **Hamilton to P.G Tait 1854 (Hankins, 1980).**

Mathematicians had not thought that whole new algebra could be constructed with their own rules. Today it is not so shocking, but you, dear reader, may still appreciate a small overview of quaternion properties. Hamilton of course has a wealth of papers on this topic in particular "On Quaternions, or on a new system of imaginaries in algebra" (Hamilton, 1844) is an excellent compendium of the articles.

Quaternions form a four dimensional non commutative normed divisional algebra over real numbers. Hamilton describes his quaternion as having one real and three imaginary parts as follows:

$$H = a + bi + cj + dk \tag{A.1}$$

often the i,j,k are replaced with x,y,z to represent 3 dimensional coordinates and this is perhaps the sense which best suits Gaia.

Hamilton's rules should also be noted, this is of course where he leads us away from traditional algebra somewhat. The following are taken directly from (Hamilton, 1844):

$$i^2 = j^2 = k^2 = ijk = -1 \tag{A.2}$$

$$i^2 = j^2 = k^2 = -1; \tag{A.3}$$

$$ij = k, \quad jk = i, \quad ki = j; \tag{A.4}$$

$$ji = -k, \quad kj = -i, \quad ik = -j; \tag{A.5}$$

In fact the quaternion may be considered to be the sum of a scalar and vector, the scalar here being a.

For simplicity we may consider our quaternion to be of four parts as follows:

$$a = a_1 + a_2 i + a_3 j + a_4 k \tag{A.6}$$

We may represent the quaternion as a 4 vector. Using the ^{T} meaning transpose notation we may denote this column vector as:

$$\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix}^T \tag{A.7}$$

It should be noted that in representing the quaternion as a vector there is a choice as to whether the first or last value (e.g. a_1 or a_4) represents the scalar in the quaternion. In Gaia the scalar is taken to be a_4 the last rather than the first value. This is known as the scalar last convention. In his papers Hamilton places the scalar first (A.6), since he always expresses his work algebraically there is little confusion as he uses sums of terms such as w,x,y,z. However when using vector or matrix rules this convention is of consequence.

The similarity of vectors and quaternions is no great surprise since Gibbs, of Gibbs and Heaviside those attributed with the invention of vector analysis, states it was a simplification of Quaternion methods.

Quaternions may be added and subtracted just like vectors as follows:

$$\mathbf{a} + \mathbf{b} = [(a_1 + b_1)(a_2 + b_2)(a_3 + b_3)(a_4 + b_4)]^T \tag{A.8}$$

representing $(a_1 + b_1) + (a_2 + b_2)i + (a_3 + b_3)j + (a_4 + b_4)k$.

All scalar operations: addition, subtraction, multiplication and division are as for vectors. For example:

$$s \cdot \mathbf{a} = \begin{bmatrix} s \cdot a_1 & s \cdot a_2 & s \cdot a_3 & s \cdot a_4 \end{bmatrix}^T \tag{A.9}$$

Quaternion products however are slightly more involved although if looked at from the vector perspective quite simple. It is the vector product of the two vector parts added to the two scalar products. Let $\mathbf{v_a}$ denote the vector part of the quaternion $\mathbf{a}$ which in our case is the vector $[a_1 a_2 a_3]$, then the product is as follows:

$$\mathbf{ab} = [\mathbf{v_a v_b} + a_4\mathbf{v_b} + \mathbf{v_a}b_4, a_4b_4 - \mathbf{v_a}\cdot\mathbf{v_b}]^T \tag{A.10}$$

first the vector part:

$$=> \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix} + \begin{bmatrix} a_4b_1 \\ a_4b_2 \\ a_4b_3 \end{bmatrix} + \begin{bmatrix} a_1b_4 \\ a_2b_4 \\ a_3b_4 \end{bmatrix} \tag{A.11}$$

then the scalar part:

$$a_4b_4 - \mathbf{v_a}\cdot\mathbf{v_b} = a_4b_4 - (a_1b_1 + a_2b2 + a_3b_3) \tag{A.12}$$

Which gives the following quaternion:

$$\begin{bmatrix} a_2b_3 - a_3b_2 + a_4b_1 + a_1b_4 \\ a_3b_1 - a_1b_3 + a_4b_2 + a_2b_4 \\ a_1b_2 - a_2b_1 + a_4b_3a_3b_4 \\ a_4b_4 - a_1b_1 - a_2b2 - a_3b_3 \end{bmatrix} \tag{A.13}$$

Still using our scalar last convention we may express the product using the following matrix which results in exactly the quaternion derived in (A.13):

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = \begin{bmatrix} b_4 & b_3 & -b_2 & b_1 \\ -b_3 & b_4 & b_1 & b_2 \\ b_2 & -b_1 & b_4 & b_3 \\ -b_1 & -b_2 & -b_3 & b_4 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \tag{A.14}$$

This product is not commutative $\mathbf{ab} \neq \mathbf{ba}$ it is worth noting however that unlike vector cross products quaternion products are associative. Algebraically the vector product has a fourth part which is not zero, vector analysis neatly throws this fourth part away. The quaternion product above although using vectors and matrices is identical to Hamilton's algebraic formulation - the rules in (A.2) ensure that all other algebraic terms disappear. The lack of commutativity of quaternions is perhaps what kept Hamilton from them for nearly 15 years as he tried to obey all existing numerical laws.

The conjugate of a quaternion is defined as:

$$\bar{\mathbf{a}} = \begin{bmatrix} -a_1 & -a_2 & -a_3 & a_4// \end{bmatrix} \tag{A.15}$$

The inverse of a quaternion is given by:

$$\mathbf{a}^{-1} = \frac{\bar{\mathbf{a}}}{\mathbf{a}\bar{\mathbf{a}}} \tag{A.16}$$

A.1 Quaternion rotations

More of interest in Gaia terms is the property of the quaternion which allows a rotation around a unit vector to be expressed as, again considering a vector and scalar part:

$$[\mathbf{v}, s] = [\mathbf{v}sin(1/2\phi), cos(1/2\phi)]^T \tag{A.17}$$

These components are sometimes refereed to as Euler parameters. Hamilton implies we should see the quaternion more as an operator and not an object. It is in this sense that such a quaternion may be used to translate points in three dimensional space, this is what makes them popular for computer games and problems such as attitude determination and modeling. We may express any point for translation as $[\mathbf{p}, 0]^T$ thus setting the scalar part to zero. The translation of the point is given by:

$$\mathbf{p}' = \mathbf{q}\mathbf{p}\mathbf{q}^{-\mathbf{1}} = \mathbf{q}\mathbf{p}\bar{\mathbf{q}} \tag{A.18}$$

Of course for any ϕ in (A.17) we could equally well have $2\pi = \phi$ as an equivalent angle. For any quaternion $\mathbf{q}$ an equivalent quaternion also exists which is its negative i.e. $\mathbf{q} = -1\mathbf{q}$.

Another very useful feature of the quaternion is that a series of quaternions applied to point may be expressed as the product of the quaternions, which gives a single quaternion to apply. In this case the original Hamilton convention is used in Gaia where rotations are applied from right to left. This is perhaps counterintuitive to someone familiar with matrix multiplication. A comprehensive overview of attitude representations may be found in (Shuster, 1993).

Appendix B

Acronyms used in this document

As for any large endeavour there are copious acronyms batted about in the Gaia realm. Interestingly Gaia itself is no longer an Acronym! At one time it did stand for Global Astrometric Interferometer for Astrometry - but not since around 1998 when the design switched from interfometry.

The following table has been generated from the on-line Gaia acronym list:

Acronym	**Description**
AC	ACross scan (direction)
ADASS	Astronomical Data Analysis Software and Systems
AEE	Agência Espacial
AF	Astrometric Field (in Astro)
AGIS	Astrometric Global Iterative Solution
AL	ALong scan (direction)
AO	Announcement of Opportunity
ASC	ASCending (in SQL command)
ASCII	American Standard Code for Information Interchange
BAM	Basic-Angle Monitoring (Device)
BC	Before Christ
BP	Blue Photometer
CCD	Charge-Coupled Device
CDROM	Compact Disc Read-Only Memory (also known as CD-ROM)
CESCA	CEntre de Supercomputacío de CAtalunya
CG	Conjugate Gradient
CNES	Centre National d'Etudes Spatiales (France)
CORBA	Common Object Request Broker Architecture
CTI	Charge Transfer Inefficiency
CU	Coordination Unit (in DPAC)
CVS	Concurrent Versions System (obsolete)
CoMRS	Centre of Mass Reference System

DB	DataBase
DBMS	DataBase Management System
DIVA	Deutsches Interferometer für Vielkanalphotometrie und Astrometrie (cancelled)
DM	Data Model
DPAC	Data Processing and Analysis Consortium
DPACE	Data Processing and Analysis Consortium Executive
DPC	Data Processing Centre
DPCT	Data Processing Centre Torino (ALTEC)
ECSS	European Cooperation for Space Standardisation
EMC	Electro-Magnetic Compatibility
ESA	European Space Agency
ESAC	European Space Astronomy Centre (VilSpa)
ESTEC	European Space research and TEchnology Centre (ESA)
FAST	Fundamental Astronomy by Space Techniques (Hipparcos)
FEA	Field in XP tracker for work package tracking
FITS	Flexible Image Transport System
FL	First Look
FLOP	FLoating-point OPeration
FOV	Field of View (also denoted FOV)
FPA	Focal Plane Assembly (Focal Plane Array)
GAP	Gaia Archive Preparations (DPAC WG)
GB	GigaByte
GDAAS	Gaia Data Access and Analysis Study (obsolete)
GIS	(Astrometric) Global Iterative Solution
GMV	Spanish 'business solutions' company
GSR	Gaia Sphere Reconstruction
GST	Gaia Science Team
GTK	Globus ToolKit
GUI	Graphical User Interface
HEALPix	Hierarchical Equal-Area iso-Latitude Pixelisation
HPC	High Performance Computing
IBM	International Business Machines
ICD	Interface Control Document
ICRS	International Celestial Reference System
IDT	Initial Data Treatment
IDU	Intermediate Data Update
IPC	Industrial Policy Committee (ESA)
IT	Information Technology
JDBC	Java DataBase Connectivity
JDK	Java Development Kit
JHU	Johns Hopkins University
JJPF	Java/Jini Parallel Framework
JNI	Java Native Interface
JSP	Java Server Page
L3CCD	Low-Light-Level CCD (obsolete)

LHS	Left Hand Side
MDB	Main DataBase
MPI	Message Passing Interface
MainDB	MAIN DataBase
NDAC	Northern Data Analysis Consortium (Hipparcos)
NM	Normal Mode (AOCS)
OO	Object Oriented
OPUS	Pipeline System from STScI - not an acronym
PERT	Program Evaluation Review Technique
PPN	Parametrised Post-Newtonian (formalism in gravitational physics)
PSF	Point Spread Function
PhD	Doctorate in Philosophy
QSO	Quasi-Stellar Object
RAC	Real Application Cluster
RAM	Random Access Memory
RHS	Right Hand Side
RMI	Remote Method Invocation
RP	Red Photometer
RUP	Rational Unified Process
RVS	Radial Velocity Spectrometer
SAG	Science Advisory Group (obsolete; superseded by GST)
SAN	Storage Area Network
SDSS	Sloan Digital Sky Survey
SED	Spectral Energy Distribution
SGC	South Galactic Cap
SM	Sky Mapper
SOC	Science Operations Centre
SQL	Structured Query Language
SRS	System Requirement Specification
SSS	System and Software Specification
SVD	Singular Value Decomposition
SVN	SubVersioN (Source code control system).
TB	Tera Byte
TDI	Time-Delayed Integration (CCD)
UB	University of Barcelona (Spain)
UML	Unified Modeling Language
UWE	Unit-Weight Error
VARI	VARIability processing
VPU	Video Processing Unit
WFS	WaveFront Sensor
WP	Work Package
XMM	X-ray Multi-mirror Mission (ESA; officially known as XMM-Newton)
XP	eXtreme Programming

Appendix C

Published Papers

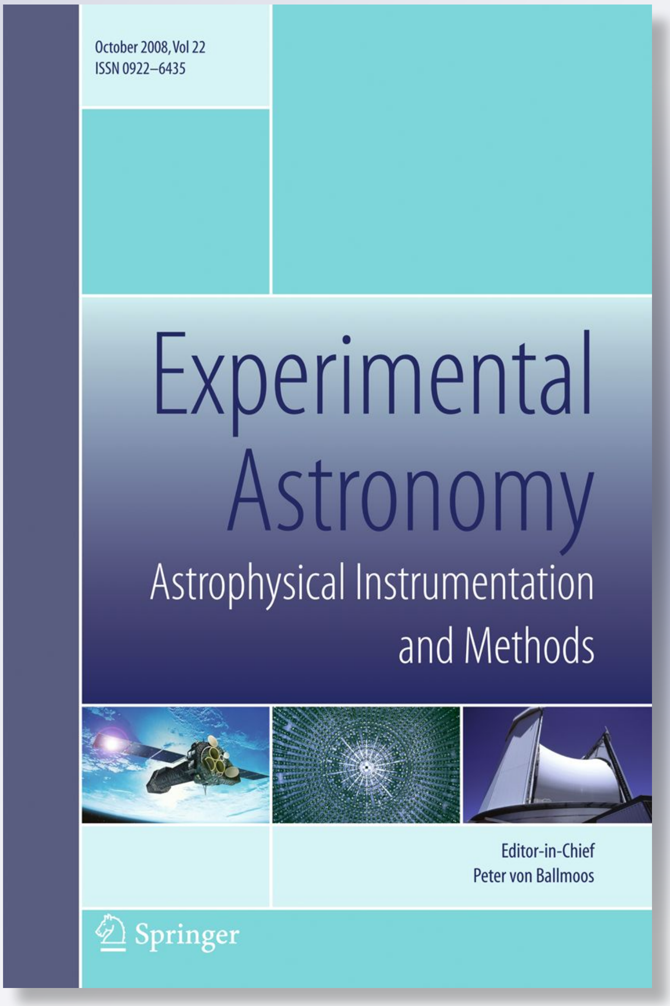
Implementing the Gaia Astrometric Global Iterative Solution (AGIS) in Java

William O'Mullane, Uwe Lammers, Lennart Lindegren, Jose Hernandez & David Hobbs

Experimental Astronomy
Astrophysical Instrumentation and Methods

ISSN 0922-6435

Exp Astron
DOI 10.1007/s10686-011-9248-z



Exp Astron
DOI 10.1007/s10686-011-9248-z

ORIGINAL ARTICLE

Implementing the Gaia Astrometric Global Iterative Solution (AGIS) in Java

**William O'Mullane · Uwe Lammers ·
Lennart Lindegren · Jose Hernandez ·
David Hobbs**

Received: 19 December 2010 / Accepted: 27 July 2011

Abstract This paper provides a description of the Java software framework which has been constructed to run the Astrometric Global Iterative Solution for the Gaia mission. This is the mathematical framework to provide the rigid reference frame for Gaia observations from the Gaia data itself. This process makes Gaia a self calibrated, and input catalogue independent, mission. The framework is highly distributed typically running on a cluster of machines with a database back end. All code is written in the Java language. We describe the overall architecture and some of the details of the implementation.

Keywords Astrometry · Satellite · Algorithms · Implementation · Data management

W. O'Mullane (✉) · U. Lammers · J. Hernandez
European Space Astronomy Centre (ESAC), P.O. Box—Apdo. de correos 78,
28691 Villanueva de la Cañada, Madrid, Spain
e-mail: womullan@sciops.esa.int

U. Lammers
e-mail: Uwe.Lammers@sciops.esa.int

J. Hernandez
e-mail: Jose.Hernandez@sciops.esa.int

L. Lindegren · D. Hobbs
Lund Observatory, Lund University, Box 43, 22100 Lund, Sweden

L. Lindegren
e-mail: Lennart.Lindegren@astro.lu.se

D. Hobbs
e-mail: David.Hobbs@astro.lu.se

Published online: 17 August 2011

1 Introduction

Astrometry is one of the oldest pursuits in science. The measurement of positions and later motions of celestial bodies has been an occupation for millennia. The most famous, but now lost, star catalogue of the Antiquity was compiled around 129 BC by Hipparchus [24], whose name is echoed in the Hipparcos mission [8] which brought the first space-based astrometry. Gaia continues in this ancient tradition using the most modern of techniques.

ESA is due to launch the ∼2000 kg Gaia satellite in 2013 on a Soyuz-Fregat rocket to the L2 point some 1.5 million km from earth. It consists of an astrometric instrument with two viewing directions, complemented by photometric and radial-velocity instruments providing astrophysical information and allowing it to build a phase-space map of our galaxy.

Over its five-year mission Gaia will obtain astrometric and photometric data for about a thousand million sources (stars, quasars, and other point-like objects); a subset of about 250 million of the brighter sources will also be observed spectrographically. Gaia will use a mosaic of CCD detectors operated in a drift-scanning mode throughout the five years, producing an average of approximately 700 individual CCD observations of each source and covering the entire sky three-fold every six months. For more detailed overviews of the Gaia project and its science goals we refer to, e.g., [13, 15, 16, 25] and [6].

A central part of the data processing for Gaia is the so-called Astrometric Global Iterative Solution (AGIS), which transforms the $\sim 10^{12}$ individual observations into an astrometric catalogue of unprecedented accuracy. The full mathematical details of AGIS are given elsewhere [17] and are only briefly referred to below. In the present paper we discuss the overall architecture of the processing framework that is being set up to carry out this huge task, as well as some details of the implementation.

When reading this paper it should be borne in mind that the word *Object* will be used in the sense that is normal in computer science or object-oriented programming. It should not be confused with an astronomical object, for which, in general, we use the term *Source*. For improved clarity, names of classes and methods are generally set in italics when they appear in regular text.

2 The Gaia Astrometric Global Iterative Solution (AGIS)

2.1 Astrometry as a minimization problem

In [8] the general principle of a global astrometric mission is succinctly formulated as the minimization problem:

$$\min_{\mathbf{s},\mathbf{n}} \left\| \mathbf{g}^{\mathrm{obs}} - \mathbf{g}^{\mathrm{calc}}(\mathbf{s}, \mathbf{n}) \right\|_M \tag{1}$$

where $\mathbf{g}^{\text{obs}}$ is the vector of all the observations (measurements), $\mathbf{g}^{\text{calc}}$ the corresponding calculated values, and the norm is calculated in some metric M that takes into account the different weights of the observations.

The vector $\mathbf{s}$ represents the (unknown) astrometric parameters of the sources. As described in detail in [17], each source i is modelled in terms of six astrometric parameters, namely:

α_i right ascension at a given reference time, i.e., the longitude-like coordinate along the celestial equator

δ_i right ascension at a given reference time, i.e., the angular distance from the celestial equator (positive towards north)

ϖ_i annual parallax, inversely proportional to distance from the sun

$\mu_{\alpha*i}$ $(= \mu_\alpha \cos\delta)$ proper motion in right ascension, i.e., the annual change in α times $\cos\delta$

$\mu_{\delta i}$ proper motion in declination, i.e., the annual change in δ

v_{ri} radial velocity, i.e., the rate of change of the distance to the source.

The radial velocity v_{ri} is best determined spectroscopically, using the Doppler shift of spectral lines, and is not included among the unknowns to be determined by the astrometric solution. The vector $\mathbf{s}$ therefore contains five unknowns for each source. The astrometric solution will operate on a subset of about 10% of the sources known as the primary sources (see Section 7.5), so the total number of astrometric unknowns is some 5×10^8.

The vector $\mathbf{n}$ contains the nuisance parameters, i.e., all other parameters that need to be determined simultaneously with $\mathbf{s}$, using the same observations, because they cannot be measured accurately enough by other means. These include the satellite attitude, the geometric calibration of the instrument, and a few global parameters. Their total number is of the order of 10^7.

2.2 Iterative solution

Equation (1) means that the model, encapsulated by the function $\mathbf{g}^{\text{calc}}$, is fitted to the observations by adjustment of the parameters $\mathbf{s}$ and $\mathbf{n}$. To directly fit all parameters is infeasible, considering their number in excess of $n = 5 \times 10^8$. A brute-force direct solution would require about $n^3/6 \sim 2 \times 10^{25}$ FLOPs and the normal equations matrix would occupy about $n^2/2 \sim 10^{17}$ doubles or 1 exabyte (1 million TB) of storage. Rather than a direct solution we take a block iterative approach.

We model the effects of the source, attitude, calibration and global parameters independently, treating the dependencies as given. Hence to solve for the astrometric parameters of a source we assume some attitude, calibration and global parameters; then for calibration we assume the global, attitude and astrometric parameters, and so on. The order in which this is done should in principle not matter although solving the astrometry for the individual sources first is logical and has some advantages (Section 7.5). Hence the solution would involve four relatively independent blocks of equations, where each takes the form of the general minimization problem of (1), although only for a subset of the parameters. The four blocks are referred to as the Source Update, Attitude Update, Calibration Update, and Global Update.

The convergence properties of this kind of (simple) iterative solution were essentially unknown when the Gaia data processing system was first planned. Although it was *felt* that it should converge, there was no proof of even that. The early work outlined in [20] was a first indication that convergence in a few tens of iterations should be possible. Subsequent experiments have shown that the iterations do indeed converge, although slowly, and that the convergence speed can be improved considerably by modifying the updates to take into account previous updates. The current solution method, based on the conjugate gradients algorithm, converges and effectively removes all systematic errors in the initial catalogue data in some 40–100 iterations, when applied to simulated data [3]. In practice one must iterate until the updates become very small, and further work continues to define an exact convergence criterion.

The efficient software implementation of the block iterative solution is challenging. A first attempt for such a solution during the Hipparcos data processing was abandoned. A basic proof of concept, actually more a pseudo implementation, using again Hipparcos data and a database management system, was presented in [20]. A good deal of effort went into scaling this up to Gaia dimensions until finally a degree of success was gained by the ESAC group [22] in 2005. It is this ESAC framework which is presented here and which shall continue to be developed up to and even after the launch of Gaia.

3 Overview of the AGIS data processing system

AGIS is just one of many parts of the Gaia processing, a central or core part certainly but still a part. In the overall design of the Gaia processing system the Main Database is the central repository of all information. Figure 1 depicts AGIS is this broader context with the Main Database.

A simplified overall AGIS picture is presented in Fig. 2. Each of the components in the picture may run on practically any regular machine apart from the Attitude Update Server, which requires a little more memory (of the

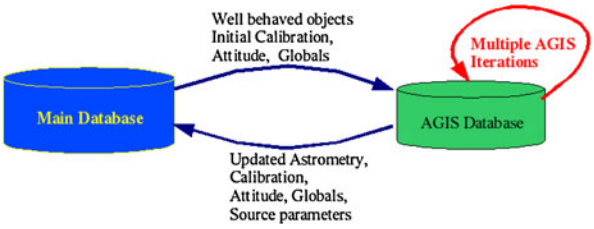


Fig. 1 AGIS, like other Gaia processing systems, extracts data from the Main Database. Updated results are fed back to the Main Database and merged with results coming from other processing systems

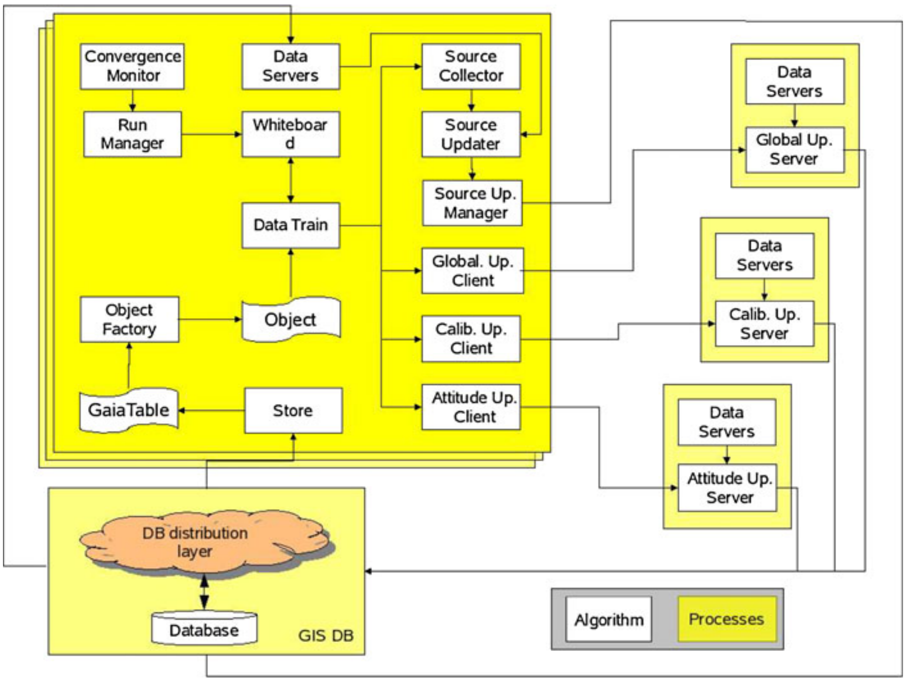


Fig. 2 Logical overview of AGIS. The many processes of AGIS run on many different machines (not shown here). The *large box* on the left represents the *DataTrain*, of which there may be a great number running. On the *right* are the update servers, of which there may be only one of each kind running in the entire system. A database management system underpins all of these processes

order of 16 GB). The *DataTrain*, as mediator, is seen in the middle of the left box and is explained in some detail in Section 4.3. The database systems—currently InterSystems Caché, Oracle Real Application Clusters, or (for small data sets) Apache Derby—may also run on several machines (or nodes) to improve data access performance. The data access and storage is abstracted through the *Store* interface which is described in Section 4.4. The algorithms and collectors are described in Section 7.

The AGIS system is deployed on a local multi-processor machine dedicated to Gaia. All the classes are available on each node but objects will be run on specific nodes according to the configuration specified in the *agis.properties* file. Objects on different hosts communicate through Remote Method Invocation (RMI), although we actually use JBoss remote-method calls for efficiency. This would be an ideal candidate for Enterprise Java Bean (EJB) implementation but we found EJBs very inefficient. In general a class with the name *SomeServer* will only have one instance on the cluster, while the *DataTrain* may have numerous instances, e.g., one on each node in the cluster. Internally the *DataTrain* makes use of multiple processors and cores available in a node.

4 Data access

The key to an efficient implementation of AGIS is in the data access. Even with today's machines, accessing a large volume (tens of terabytes) in both spatial and temporal order is demanding.

4.1 Data access patterns

Looking at the four main blocks of AGIS we see that each has a seemingly unique data access pattern, viz.:

Source	All observations of a given source—spatial
Attitude	All observations within a given time period—temporal
Calibration	All observations within a given time period falling on a given CCD—temporal/spatial
Global	All observations—any order

(The 'observations' here refer to the *AstroElementary* objects described in Section 5.) The naive approach would be to go through the data once for each block, updating the parameters in turn and then repeating this for each iteration. This is indeed the basic mathematical formulation of the block-iterative solution method and the corresponding data access scheme is depicted in Fig. 3(left).

Running through the approximately ten terabytes of data four times per iteration is rather daunting, considering that many tens of iterations will be needed. Immediately, though, we see that the calibration and attitude updates are similar enough that the can perhaps be combined. The global update is order-independent and as such could be combined with the data access of any of the other blocks, for example source. Indeed this was already remarked

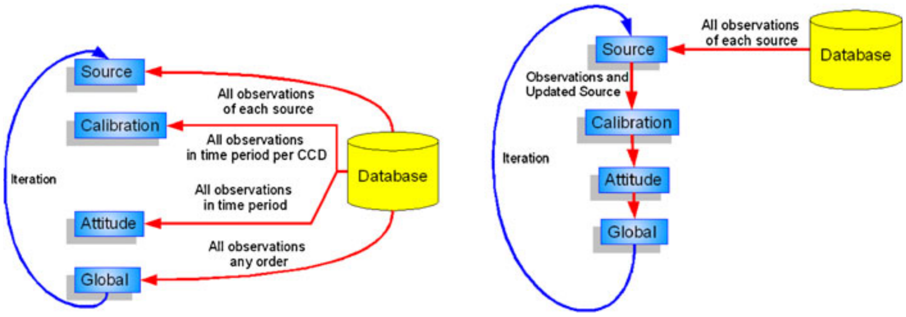

Source
Calibration
Iteration
Attitude
Global
All observations of each source
All observations in time period per CCD
Database
All observations in time period
All observations any order
All observations of each source
Database
Source
Observations and Updated Source
Calibration
Iteration
Attitude
Global

Fig. 3 *Left* Each block of the AGIS solution has a slightly different data access requirement. This could cause four passes through the data for each AGIS iteration. However, it is immediately clear that some of these could be combined, e.g., the calibration and attitude updates could run together, and similarly the source and global updates. *Right* With a little in-memory accumulation in the calibration, attitude and global update blocks, a complete iteration can be made in one pass though the data. Hence the optimal ordering is spatial. Furthermore the updated source parameters may already be used in the other blocks

in [20], where the prototype made just two passes through the data for each iteration rather than four. The question then is: could this be reduced to one pass through the data per iteration?

4.2 A question of order

Let us assume that all four blocks could be executed in one pass; what then would be the impact of the ordering of the data? There are two primary orderings we may choose: spatial or temporal.

Temporal ordering If we assume an ordering based on the time of observation, then for the attitude we may read the data once, break it in time chunks suitable for the attitude update, process each chunk in turn and finish with it. With a small buffer we may also accumulate the observations required for the calibration and similarly finish with calibrations in a timely manner during the same pass through the data. For the global update the order is immaterial, so it can be done in parallel with the attitude and calibration updates.

The problem here comes with the source update. Since any given source is observed many times over the entire mission, if we process in time order we must accumulate the data for each source until we have all observations of it. This will not happen until we have seen all of the data—only then can we be certain that no more observations of a given source will show up. This would effectively mean that all observation data would end up in memory. For a hundred million sources (with almost 10^{11} observations) and some clever organizing this would be of the order of 5 TB of data, which is infeasible to have in shared memory on our budget. The final solution may require five times as many observations. The alternative is another pass through the data in spatial order. Since we must wait until the end of the first pass for the updated calibration, attitude and global parameters, these updated values could already be used for the source update.

Spatial ordering If we assume a spatial ordering, i.e., that all observations of a source are clustered together, then the story is quite different. Now we may process each source to find its new astrometric solution, which can immediately be written out to disk. Since we are finished with that source, the updated parameters may be used to find its contributions to the global parameters. The situation for the attitude and calibration updates is however that all contributions from all observations must be accumulated until the end of the pass through the data—only then may the calibration and attitude updates be calculated. It is important to note that it is not the observations which must be held but their contribution to the matrices of attitude and calibration, which is much smaller than the accumulation of the source matrices in the temporal ordering. The entire attitude accumulation for the five year mission data can be done in 8 GB of memory. The size of the calibration matrix depends on the number of calibration artifacts—currently it requires about 4 GB but is estimated to need as much as 32 GB when additional calibration parameters are added in the coming years. Hence with spatial ordering one pass may be made though the data for each AGIS iteration, as depicted in Fig. 3(right), and a minimum amount of data needs to be held in memory.

This clearly represents a better approach to the ordering from a technical point of view. Additionally, it is more natural to keep astronomical data of the same part of the sky together and easily accessible. Hence the AGIS database has observations of the same source sequentially grouped together on disk.

4.3 Getting data to the algorithms: the *DataTrain* and *Taker*

Throughout the Gaia processing there are choices to be made concerning data access patterns such as those outlined in Section 4.2. The ideal approach, for efficiency, is a data driven approach whereby data is accessed in the sequential order in which it is stored. Hence rather than algorithms requesting data they should be presented with data by a *mediator*. The mediator pattern [11] is a very powerful tool for decoupling software modules. The implementation of the mediator for the astrometric solution is called the *ElementaryDataTrain*.

The generic notion of a *DataTrain* (Fig. 4) is to access data in the fastest possible manner, usually meaning sequentially, and call a given set of algorithms passing them the data. The concept and code are quite simple. To enable the calling of the algorithms in a generic manner they must implement the *Taker* interface, which has a method to 'take' some data. By implementing this interface, the algorithm will have its input when it is called by the *DataTrain*.

More specifically, for AGIS the *ElementaryDataTrain* accesses *AstroElementary* objects, which are effectively the observations of a given source. The train decides which data to access by taking a *Job* (see Section 6.2). It uses the *Store* to access a set of *AstroElementary* objects, each of which

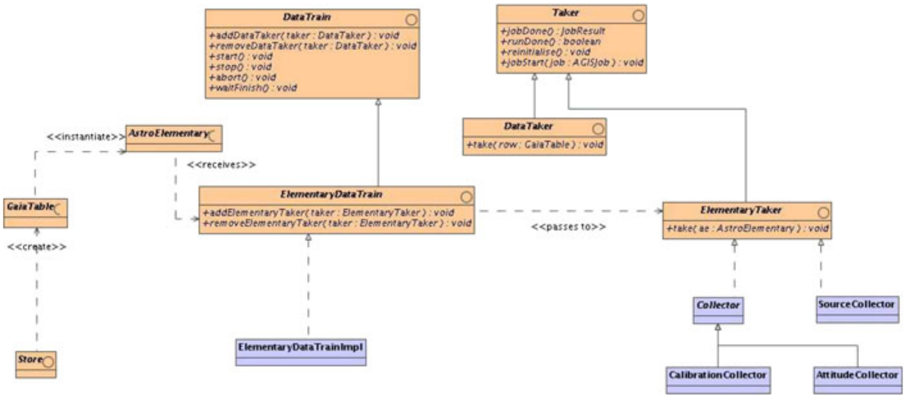


Fig. 4 The *DataTrain* acts as a mediator between algorithms and data access (the *Store*) thus leading to a less coupled system. The *ElementaryDataTrain* accesses *AstroElementary*s in the fastest possible manner for the AGIS algorithms. The participating algorithms must implement the *Taker* interface

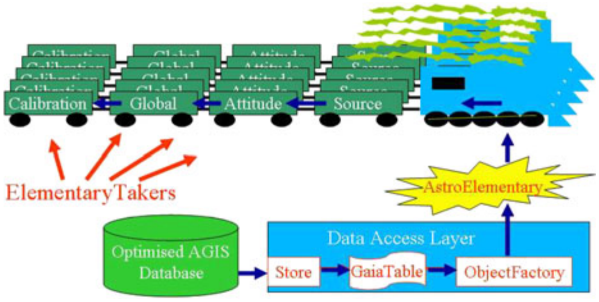


Fig. 5 Here the blue arrow shows the flow of data from the database through the *Store* and *ObjectFactory* to the algorithms attached to the *ElementaryDataTrain*. We may think of the *ElementaryDataTrain* as driving through the database, passing observations to the algorithms. We may have as many trains in parallel as we wish

is then passed to each registered *ElementaryTaker*, i.e., the source, attitude, calibration and global update algorithms. Each algorithm (see Section 7) must implement the *ElementaryTaker* interface to allow the *DataTrain* to interact with it. The *ElementaryDataTrain* has a method for registering the algorithms (*addElementaryTaker* in Fig. 4). The algorithms must then accumulate observations until they can process a particular source or time interval. This forces the algorithms to accept data in the order it is stored allowing the infrastructure to be built without fixing the data storage order. Choosing spatial ordering (Section 4.2) means that all of the elementaries for a given source are sequential. Any given train accesses complete sets of elementaries with respect to sources. The cartoon in Fig. 5 depicts this in a another manner showing how the *AstroElementary* is constructed by the *ObjectFactory* from a *GaiaTable* resulting from a query to the database through the *Store* interface. The *AstroElementary* is then passed to the algorithms attached to the *DataTrain*.

4.4 Abstraction of data storage: the *Store*

To give a degree of independence from the physical storage mechanism, it is normal to use some abstraction. Java interfaces provide an excellent approach to provide such insulation. Creating an interface is a small coding overhead, while in usage one gets a real implementation, i.e., without overhead. It is very important to realize that a Java interface is a contract binding the using class and the providing class but does no translation of any kind. This should not be confused with rooted persistence systems requiring all classes to inherit from some root class. Here we simply have to implement a few methods implied by the interface. They are more for our convenience than a design principle—we also like to keep clear in our code which objects we will be storing and which we will not. It is also useful in the *ObjectFactory* to have a base interface to cast to, other than *Object*. We are not far from Java Persistence Architecture (JPA) in both principle and implementation—this however was not mature when we started in 2005. More recently we have considered simply switching to something like Hibernate but found the offerings far slower than our own system. We could remove the restriction of having *GaiaRoot* but it has not been a pressing issue to date. Having our own *Store* also made it easy to build a *CacheStore* which took advantage of the Cache high speed interface, they do not support JPA.

No algorithm code in the system interacts directly with the database management system; rather a query interface to the data is provided through the *Store* interface (see Fig. 6). The implementation of the *Store* is hidden behind

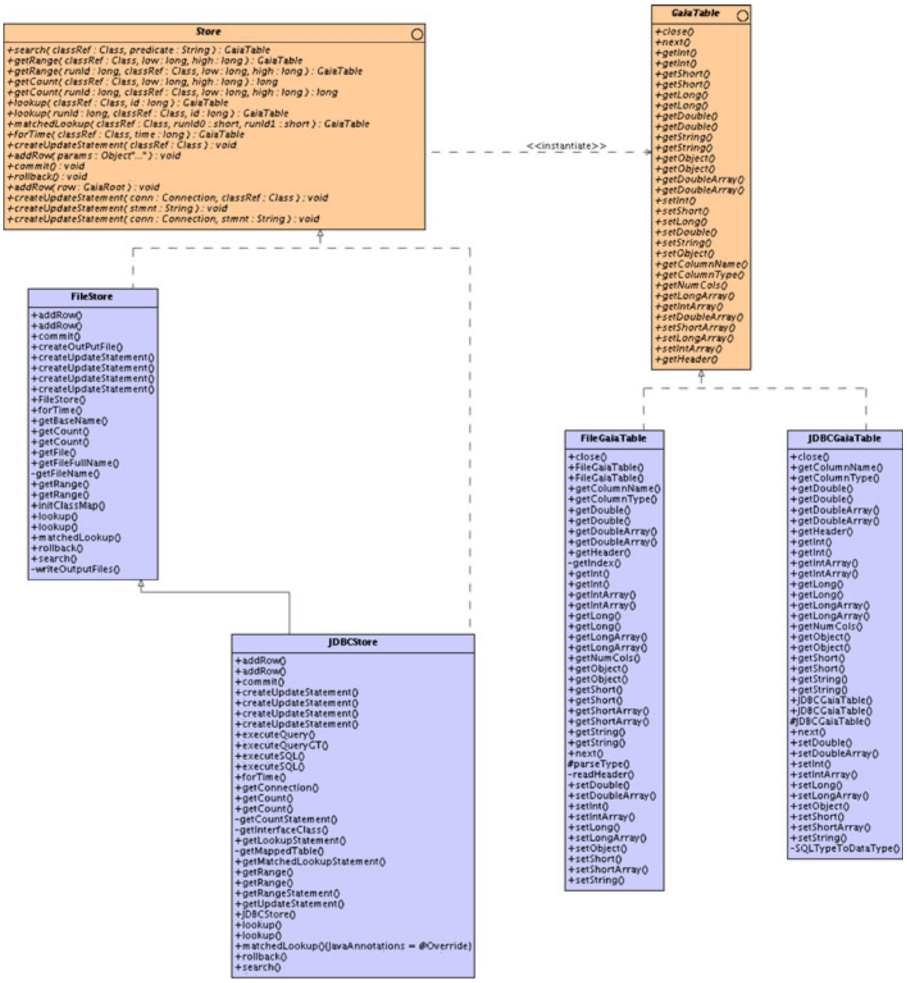


Fig. 6 The *Store* provides an interface for data access, whereby many *Store* implementations may exist. In the Figure we see a *FileStore* and a *JDBCStore*, both of which implement *Store*. With these implementations of AGIS code we may switch between FITS files and a JDBC Database for storage in a seamless manner

the interface; thus the data store may be implemented in files or any database management system.

An implementation of the *Store* is requested from the *AGISFactory*, the actual implementation of the store is configured using the *gaia.tools.dal.Store* property in the *agis.properties* file and thus can be changed at run-time (rather than at compile-time). The *Store* interface includes an explicit range query which returns all objects within a certain id range, which is required to support the *DataTrain*.

As depicted in Fig. 6 there are multiple implementations of the *Store*. The *FileStore* does not support the same level of querying as the *JDBCStore* but is sufficient for running the testbed on a laptop. Most recently we have also implemented a *CacheStore* over the InterSystems Caché database.

GaiaTable in Fig. 6 represents an interface to tabular data. The assumption of dealing only with tabular data is a major simplification for AGIS. This is a fair assumption dealing with astrometry data. Both files (be they FITS or whatever) and relational database tables may be represented as a *GaiaTable*. The interface defines methods for retrieving the next row and for getting columns by name or index. The whole row may be passed to the algorithm or *ObjectFactory* and it may extract the required columns. The *DataTrain* loads the entire row.

The *GaiaRoot* UML (Unified Modeling Language) diagram is given in Fig. 7. Color interfaces are shown in brown colour (and are also marked with a ∘), while implementations are in blue. Any objects in the Gaia data model which use the *Store* (see also Fig. 6) and *ObjectFactory* must implement this interface. A basic implementation is provided which most classes may inherit from, but in some cases, due to single inheritance in Java, this may not be

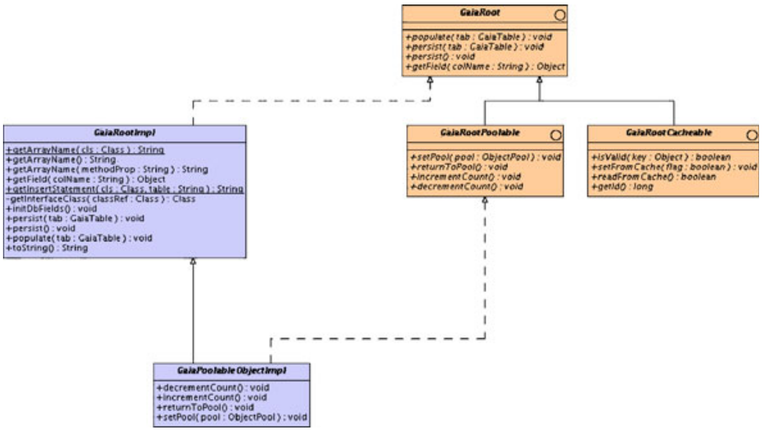

GaiaRoot
+populate(tab : GaiaTable) : void
+persist(tab : GaiaTable) : void
+persist() : void
+getField(colName : String) : Object
GaiaRootImpl
+getArrayName(cls : Class) : String
+getArrayName() : String
+getArrayName(methodProp : String) : String
+getField(colName : String) : Object
+getInsertStatement(cls : Class, table : String) : String
-getInterfaceClass(classRef : Class) : Class
+initDbFields() : void
+persist(tab : GaiaTable) : void
+persist() : void
+populate(tab : GaiaTable) : void
+toString() : String
GaiaRootPoolable
+setPool(pool : ObjectPool) : void
+returnToPool() : void
+incrementCount() : void
+decrementCount() : void
GaiaRootCacheable
+isValid(key : Object) : boolean
+setFromCache(flag : boolean) : void
+readFromCache() : boolean
+getId() : long
GaiaPoolableObjectImpl
+decrementCount() : void
+incrementCount() : void
+returnToPool() : void
+setPool(pool : ObjectPool) : void

Fig. 7 All data objects implement *GaiaRoot*, which makes certain methods available to the *Store*. All data objects are interfaces, not real classes—this allows them to be easily replaced by different implementations

possible. In fact practically all of the required functionality is in the *Store* or *ObjectFactory*.

Interfaces were chosen for the data classes originally, since the first implementations in 1998 used Objectivity/DB (from Objectivity, Inc.) which was a rooted system, thus requiring the objects to actually inherit from the Objectivity/DB base class. Even then the *Store* was working both with Oracle Real Application Clusters and Objectivity/DB, which meant having two implementations of the data objects. These days we usually only have one implementation; however, there are instances where the interfaces are still useful. For example higher-level classes such as *AstrometricSource* can have multiple subclasses. These may not follow the same inheritance hierarchy but can still be *AstrometricSource*s since it is an interface; if it were only a class there could be inheritance problems.

4.5 Access to objects: the *ObjectFactory*

The *Store* deals essentially with tables but some code will require objects. The *ObjectFactory* sits on top of the *Store* and returns objects implementing the data model interfaces. The object-from-table method of the interface is also exposed, allowing code to do this conversion exactly when required. We need to take care that not too many pieces of code perform such a transformation—preferably it would be done once by the *DataTrain*. Splitting this out allows for very direct measurement of the performance.

This is implemented as a *Generic* class. The *Factory* is instantiated for a specific data model interface and then provides a method returning that class of object only. Java Generics are very nice for this and, although similar to C++ templates, should not be confused as being the same. Generics provide type checking and safety but they do not generate extra code with new types.

The *Factory* relies on the *populate* method of the *GaiaRoot* to populate the fields of the object from a *GaiaTable*. A generic implementation of this using a mapping from the configuration file is provided in the *GaiaRootImpl* class. This provides a convenient mechanism to read the data from the *Store* into a Java object that can be used elsewhere in the system.

The *ObjectFactory* also has caching capabilities. Whenever an object is read from the *Store* it may be cached in memory in order to avoid new reads when it is requested again. Any object which is created by the *ObjectFactory* can be made cacheable just by implementing the *gaia.tools.dm.GaiaRootCacheable* interface. The caching can also be disabled by adding an entry to the property file. The interface contains a method to determine the 'validity' of the object.

The *Factory* also has the possibility to implement object pooling. The notion here is to reuse objects by filling them with new data rather than reconstructing new ones. This technique was very popular in early Java implementations to reduce garbage collection time. Tests with the new JDK (1.5 and 1.6) show that this is no longer beneficial. Still, by having all data object creation done through one class the possibility to change the way it works later remains available.

5 The data model

The algorithms work in terms of Java objects such as *Source* and *AstroElementary*. These objects form what is generally termed a Data Model for the system.

The data used for AGIS will comprise between 10% to 50% of the sources (and their corresponding observations), corresponding to the so-called primary sources briefly discussed in Section 7.5. The selection of the primary sources is described in [17] and is implemented as several database queries. The selected data will be put in the special AGIS database (see for example Fig. 5).

The data model is made in terms of interfaces to allow easy substitution of multiple implementations. The *ObjectFactory* (Section 4.5) and *Store* (Section refsect:store) are used to construct real implementations of these interfaces but all code refers only to the interface. Hence all client code may be compiled without any implementation if necessary. This is a technique used throughout AGIS and indeed also for *GaiaTools*, the common software toolbox for all Gaia processing tasks. The most important interfaces are:

- *AstroElementary*: An object of this kind represents the transits of a celestial source over the first dedicated 10 CCD strips of the focal plane, namely, SM1 or SM2 and AF1–9 (see [17], for an outline of the CCDs in the focal plane). Each *AstroElementary* in AGIS is uniquely associated with a *Source*.
- *Source*: An object of this kind represents celestial sources that follow the standard astrometric model (thus modelled by the six astrometric parameters described in Section 2.1) and are eligible for AGIS source processing.
- *Attitude*: An object of this kind represents an interval of continuous attitude data. Attitude is parametrized using B-spline coefficients of a given order representing the four components of the attitude quaternion (Section 7.2).
- *CalibrationEffect*: The geometrical calibration of the instrument is made up of multiple *CalibrationEffect*s (Section 7.3) all of which may be configured in an XML file.

6 Distributed processing

Regardless of the ordering chosen (Section 4.2) the access of the data does not need to be done serially. Indeed we require the data to be sequential on disk but multiple parts of that sequence may be read simultaneously. In the case of sources we may process simultaneously each source, in terms of distributed computing this is 'embarrassingly parallel' [26].[1] We may theoretically gain up

[1] http://en.wikipedia.org/wiki/Embarrassingly_parallel

Table 1 Evolution of AGIS performance during the development of the processing framework

Date	Observations	Processors	Time (hr)	Normalized rate
2005	1.6×10^7	12	3	0.9×10^6
2006	8.0×10^7	36	5	0.5×10^6
2007	8.0×10^7	24	3	1.3×10^6
2008	8.0×10^7	31	1	3.2×10^6
2009	2.6×10^8	50	1.8	2.8×10^6
2010	4.0×10^9	68	9.5	6.2×10^6

Data volumes are indicated by the number of observations (*AstroElementaries*), depending on the number of sources and the length of the observation period. The *time* is the processing time per AGIS iteration for the given number of processors. The *last column* shows the throughput, in observations per processor per hour, as an indication of the real performance

to a factor N in speed by using N processors, if N is the number of sources to be processed. We say theoretically with reason, as the data must still be read from disk and we are unlikely to actually put in place 10^8 processors. Still, tests have shown that the processing time indeed decreases in proportion to the number of processors used for AGIS. Some numbers are given in Table 1.

6.1 Distributed processing frameworks

Many different approaches exist for distributed processing, and they are usually embodied in some library. However since we have an ‘embarrassingly parallel’ problem we have little need for such a complex and heavy library. In fact all that we require is already available within the standard Java library, namely:

- Communication between processing nodes: the Remote Method Invocation (RMI) framework in Java provides this.
- Access to a database or databases: the Java Data Base Connectivity (JDBC) framework provides this.
- Some form of graphics library for GUIs: Java Swing library provides this.

Additionally, in this age of the web, Java provides easy support for dynamic web site generation using Java Server Pages (JSP).

Hence an early feeling was to use the tools of Java directly, rather than try to fit the problem into one of the many distributed programming libraries, each with their own assumptions and problems. The modern programming languages of the day, such as Java, are very sophisticated in the feature set and tools they provide. For example the Java/Jini Parallel Framework (JJPF; [5]) provides some reliability on top of these tools while also taking a much more process-oriented view—each worker has a *getData* call to pass back results. JJPF is also more coordinator oriented with a single server eliciting support from available nodes to perform a computation. In the grid world the obvious contender would be the Globus Toolkit (GTK; [10]). Previous forays into GTK showed the system to be buggy and difficult to use. GTK has improved dramatically over the years, yet it still remains service oriented (we believe our problem to be data oriented) and has a large security overhead which we do not see as necessary. Indeed though [7] is positive about GTK they introduce a resource broker which seems similar to our whiteboard (Section 6.2). Unfortunately say little about the data intensive applications mentioned in the title of their paper.

The notion of just using the Java framework without some other layer was reinforced by previous experiences with the Sloan Digital Sky Survey (SDSS). On the SDSS a form of distributed query system known as CasJobs [21] was built using Web Services, the SQLServer database and the C# language.[2] This was done quite rapidly without using any special libraries beyond the facilities available in the programming language. Within the same group at Johns Hopkins a typical Grid application for finding galaxy clusters in a large catalogue was taken and quickly rewritten in C#. As reported by [19], this ran about ten times faster using a database system than the traditional file-based Grid system.

A final justification, perhaps the ultimate and obvious one, for not taking on a library is that of simplicity. It was believed the distributed computing libraries would not make the system simpler hence none were adopted.

6.2 Job distribution: the *Whiteboard*

There are at least two main approaches to controlling a grid of distributed processes. The first is to have 'agents' register with some central controller which then regulates the entire process; the second is to have a less centralized approach with more autonomous processes.

The central controller approach is very appealing and generally the way many agent-based systems work. Generally these involve monitoring resources and farming out jobs to particular processors which are not fully loaded. The central registering of agents means the controller knows how many agents of which types exist on the system, and furthermore may reject agents from particular machines or of particular types. Such systems deal well with uneven workloads and ad hoc jobs by many users. Often security layers and user tracking are included.

By contrast, an AGIS iteration could easily occupy an entire cluster for some days. There are no ad hoc programs, only the entire AGIS chain running on all data. There are no users, hence no particular need for a security overhead in terms of certificates, etc.

In our data driven approach (Section 4.3) we may consider the data as the distribution mechanism. Everything hinges on the processing of some block of data, be it a time sequence or a set of spatially ordered observations (Section 4.2). All we really need to know is if a particular part of the data set has been visited during a particular iteration. If the data segments are chosen properly we may have as many *DataTrains* running as we wish. This is very

[2] The # here is the musical sharp; hence this is pronounced 'See sharp'.

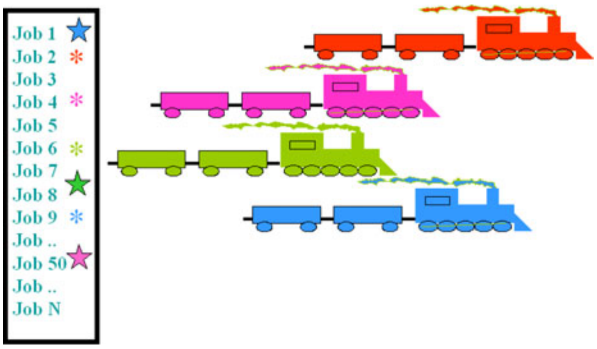


Fig. 8 A set of jobs corresponding to sequential batches of data which cover the entire data range may be posted on a whiteboard (*left*). The *DataTrain* marks a job as in progress when it starts it and as completed when it is finished. There may be many *DataTrain*s (*right*). When all jobs are done all of the data have been seen once. The whiteboard itself has no special knowledge of the jobs or the overall task—it is a simple mechanism to coordinate potentially hundreds of processes

simple and easily achieved through a whiteboard mechanism as depicted in Fig. 8.

The whiteboard is quite a simple concept for organizing many processes of varying types. Conceptually we may 'post' jobs on a whiteboard, and then workers, in our case *DataTrains*, may pick them up. The whiteboard may hold status information, e.g., about when a job started, when it ended, if all was OK, etc. In effect then the whiteboard becomes the central controller, although it exercises no control as such. Perhaps the original of the species in this respect is the OPUS pipeline from the Space Telescope Science Institute [23]. Indeed, it is the OPUS blackboard [3] design pattern which is employed here. It is noted that since its early beginning, OPUS is itself moving toward Java [18] but maintaining its heterogeneity through CORBA (Common Object Request Broker Architecture). For the purposes of AGIS, which is a pure Java implementation, a simple *Whiteboard* was coded directly in Java using a database table to hold the jobs. The latter also provides the ability to ensure that no two trains ever get the same job. The JDBC framework in Java makes the whiteboard seamlessly accessible from any node on the network—hence no need for the overhead of CORBA or some other message passing system here. The UML interface for the *Whiteboard* is shown in Fig. 9.

Regardless of the jobs being done, the whiteboard can give some information on the general state of the system. A Series of JSP pages present the whiteboard state on a website. On this site with little effort we may show jobs completed/remaining and (assuming uniform jobs) an estimate for the end time. We may also list statistics per processor simply by querying the job table in the database.

[3] Whiteboard was elected as a more modern alternative to Blackboard.

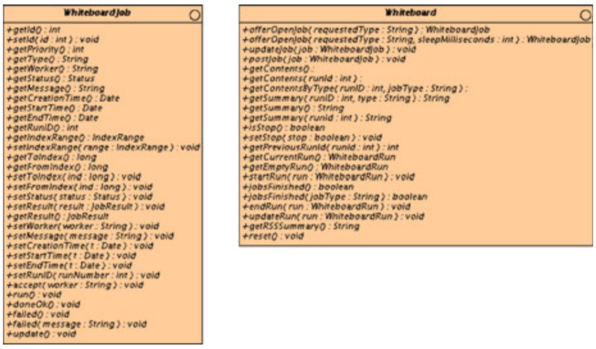

WhiteboardJob
+getId() : int
+setId(id : int) : void
+getPriority() : int
+getType() : String
+getWorker() : String
+getStatus() : Status
+getMessage() : String
+getCreationTime() : Date
+getStartTime() : Date
+getEndTime() : Date
+getRunID() : int
+getIndexRange() : IndexRange
+setIndexRange(range : IndexRange) : void
+getToIndex() : long
+getFromIndex() : long
+setToIndex(ind : long) : void
+setFromIndex(ind : long) : void
+setStatus(status : Status) : void
+setResult(result : JobResult) : void
+getResult() : JobResult
+setWorker(worker : String) : void
+setMessage(message : String) : void
+setCreationTime(t : Date) : void
+setStartTime(t : Date) : void
+setEndTime(t : Date) : void
+setRunID(runNumber : int) : void
+accept(worker : String) : void
+run() : void
+doneOk() : void
+failed() : void
+failed(message : String) : void
+update() : void
Whiteboard
+offerOpenJob(requestedType : String) : WhiteboardJob
+offerOpenJob(requestedType : String, sleepMilliseconds : int) : WhiteboardJob
+updateJob(job : WhiteboardJob) : void
+postJob(job : WhiteboardJob) : void
+getContents() :
+getContents(runId : int) :
+getContentsByType(runID : int, jobType : String) :
+getSummary(runID : int, type : String) : String
+getSummary() : String
+getSummary(runId : int) : String
+isStop() : boolean
+setStop(stop : boolean) : void
+getPreviousRunId(runId : int) : int
+getCurrentRun() : WhiteboardRun
+getEmptyRun() : WhiteboardRun
+startRun(run : WhiteboardRun) : void
+jobsFinished() : boolean
+jobsFinished(jobType : String) : boolean
+endRun(run : WhiteboardRun) : void
+updateRun(run : WhiteboardRun) : void
+getRSSSummary() : String
+reset() : void

Fig. 9 The UML interfaces for the *Whiteboard* and the *WhiteboardJob*. Note the *postJob* method used to populate the whiteboard and the *offerOpenJob* methods which the *DataTrains* use to get jobs. The job itself has methods for status and messages, etc

6.3 Overall control: the *RunManager* and *ConvergenceMonitor*

The *Whiteboard* alone is not enough to run an AGIS solution. Some other entity must post the jobs on the board for the *DataTrains* to work on. The *RunManager* has the task of coordinating iterations and the publishing of jobs as depicted in Fig. 10. The *RunManager* uses the *JobPublisher* to publish appropriate jobs, e.g., one for each block of sources to be processed. The *JobPublisher* scans a table of bounds (a list of identifiers of elementaries which are the last in a series belonging to a single source) and creates a number of jobs based on blocks of elementaries. In general the system is configured such that these jobs complete in a few minutes, as this gives a better indication of progress and the need to redo a job, in case of a problem, is detected in a timely manner. Hence there are typically thousands of jobs in a single run. Once posted, the trains pick them up and start working. The order in which the jobs are done does not matter. Jobs are also published for the calibration, attitude and global updates if these algorithms are attached to the train. These jobs execute for the entire iteration.

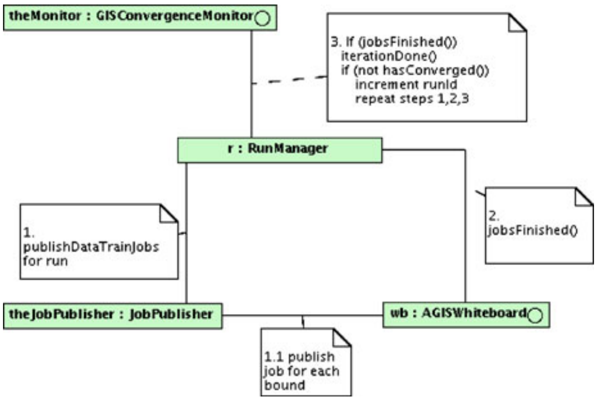


Fig. 10 Communication diagram for the *RunManager*. This summarises the *RunManagers* role in publishing jobs and checking for convergence

The *RunManager* then periodically checks to see if the *DataTrain* jobs have finished. If they are done the main part of the iteration is done, and the *GisConvergenceMonitor* is told the iteration is at an end. The *RunManager* then asks the *GisConvergenceMonitor* if the solution has converged and awaits the answer. At this point the attitude, global and calibration servers still must perform their final calculations—when these are complete the *GisConvergenceMonitor* reports the state of convergence. The convergence criterion is currently based on the typical size of the source updates in the current iteration.

If convergence has not been reached the *RunManager* starts another run through the data by publishing a new set of jobs. If it has converged the *RunManager* declares the run ended and converged.

7 Algorithms

There are effectively two types of algorithm in the system: those with a centralized part and those which are completely distributable. Let us first look at the source update algorithm which is completely distributed and subsequently at the others. The mathematical formulation of the algorithms (or blocks) is given in [17]. As explained in Section 2.2 the blocks are iterated until the solution is considered converged.

7.1 Source update

The mathematical details of the source update are provided in [17]. Very briefly, the update for source i is obtained by solving the overdetermined system of equations

$$\mathbf{A}_i\mathbf{d}_i \simeq \mathbf{h}_i\,, \tag{2}$$

where $\mathbf{d}_i$ is the n-vector of updates to the astrometric parameters $\mathbf{s}_i$ of the source (usually with $n = 5$, as described in Section 2.1), $\mathbf{h}_i$ the m-vector of residuals, where $m \gg n$ is the number of observations of the source, and $\mathbf{A}_i$ the design matrix. The problem is complemented by an m-vector of measurement uncertainties, σ_i. The residual vector $\mathbf{h}_i$ contains the observed minus the calculated values for the source, such that the jth element is $h_j = g_j^{\rm obs} - g_j^{\rm calc}(\mathbf{s}_i, \mathbf{n})$, $j = 1 \ldots m$, where $g_j^{\rm obs}$ is the observed position of the source on the CCD and $g_j^{\rm calc}$ the calculated position based on the current best estimate of the source parameters $\mathbf{s}_i$ as well as the attitude, calibration and global parameters in $\mathbf{n}$, cf. (1). The elements of $\mathbf{A}_i$ are $A_{jk} = \partial g_j^{\rm calc}/\partial s_k$ for $j = 1 \ldots m$ and $k = 1 \ldots n$. Each *AstroElementary*, consisting of up to 10 CCD transits, generates several rows of design equations.

The least-squares solution of (2) is by itself an iterative process, in order to have a self-adapting system of observation weighting (essentially by adjusting σ_i) that is robust against outliers. Typically three or four such internal iterations are needed to compute the update $\mathbf{d}_i$, after which the improved source parameters are obtained as $\mathbf{s}_i + \mathbf{d}_i$. The solution of (2) is done in a very standard fashion by forming normal equations [2], which is computationally very efficient.

The source update step is truly distributed. As the *DataTrain* passes elementaries to the *SourceCollector* (the *Taker* registered with the train for sources) it accumulates all of the elementaries for a given source. Remember that the data are stored in such a manner that all elementaries for one source are consecutive; hence, when the *sourceId* changes, the collector knows that it has all the data for a given source. Once it has a batch of elementaries, the source update is called to compute the required update of the astrometric parameters.

Figure 11 provides a UML overview of some of the classes involved in the source update. When the updated astrometric parameters are available they are passed to the *SourceUpdateManager*, which batches together several sources for efficient storage. Nothing in AGIS is ever actually updated, rather a new source row is written to the table with the current *runId*. In this way a complete history of the updates are preserved. Inserting to the database is also more efficient than updating.

In fact the *SourceUpdateManager* does not write the sources finally until the entire job is done. When all sources are updated a database transaction is opened to write all the results—only when this is done is the job considered

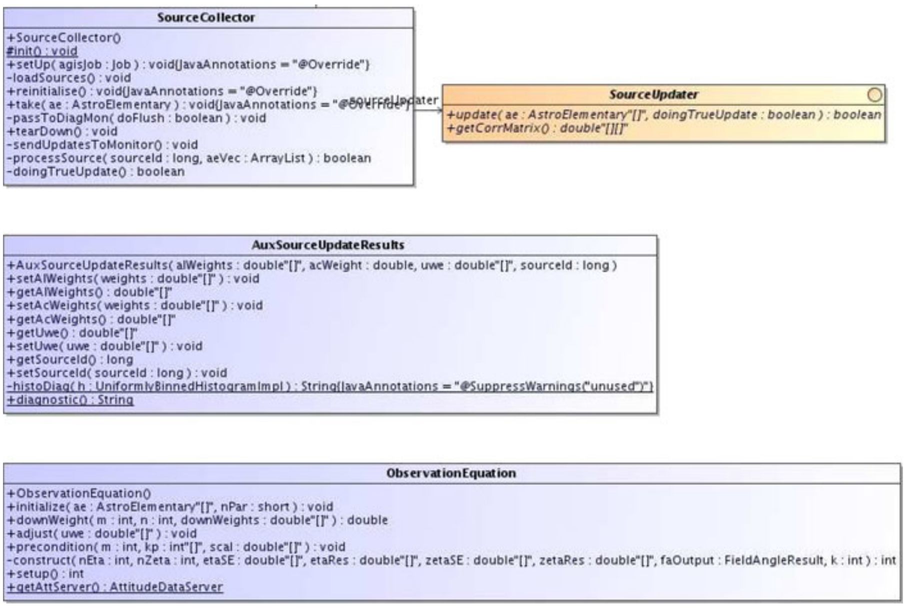

SourceCollector
+SourceCollector()
#init() : void
+setUp(agisJob : Job) : void{JavaAnnotations = "@Override"}
-loadSources() : void
+reinitialise() : void{JavaAnnotations = "@Override"}
+take(ae : AstroElementary) : void{JavaAnnotations = "@Override"}
-passToDiagMon(doFlush : boolean) : void
+tearDown() : void
-sendUpdatesToMonitor() : void
-processSource(sourceId : long, aeVec : ArrayList) : boolean
-doingTrueUpdate() : boolean
-sourceUpdater
SourceUpdater
+update(ae : AstroElementary"[]", doingTrueUpdate : boolean) : boolean
+getCorrMatrix() : double"[][]"
AuxSourceUpdateResults
+AuxSourceUpdateResults(alWeights : double"[]", acWeight : double, uwe : double"[]", sourceId : long)
+setAlWeights(weights : double"[]") : void
+getAlWeights() : double"[]"
+setAcWeights(weights : double"[]") : void
+getAcWeights() : double"[]"
+getUwe() : double"[]"
+setUwe(uwe : double"[]") : void
+getSourceId() : long
+setSourceId(sourceId : long) : void
-histoDiag(h : UniformlyBinnedHistogramImpl) : String{JavaAnnotations = "@SuppressWarnings("unused")"}
+diagnostic() : String
ObservationEquation
+ObservationEquation()
+initialize(ae : AstroElementary"[]", nPar : short) : void
+downWeight(m : int, n : int, downWeights : double"[]") : double
+adjust(uwe : double"[]") : void
+precondition(m : int, kp : int"[]", scal : double"[]") : void
-construct(nEta : int, nZeta : int, etaSE : double"[]", etaRes : double"[]", zetaSE : double"[]", zetaRes : double"[]", faOutput : FieldAngleResult, k : int) : int
+setup() : int
+getAttServer() : AttitudeDataServer

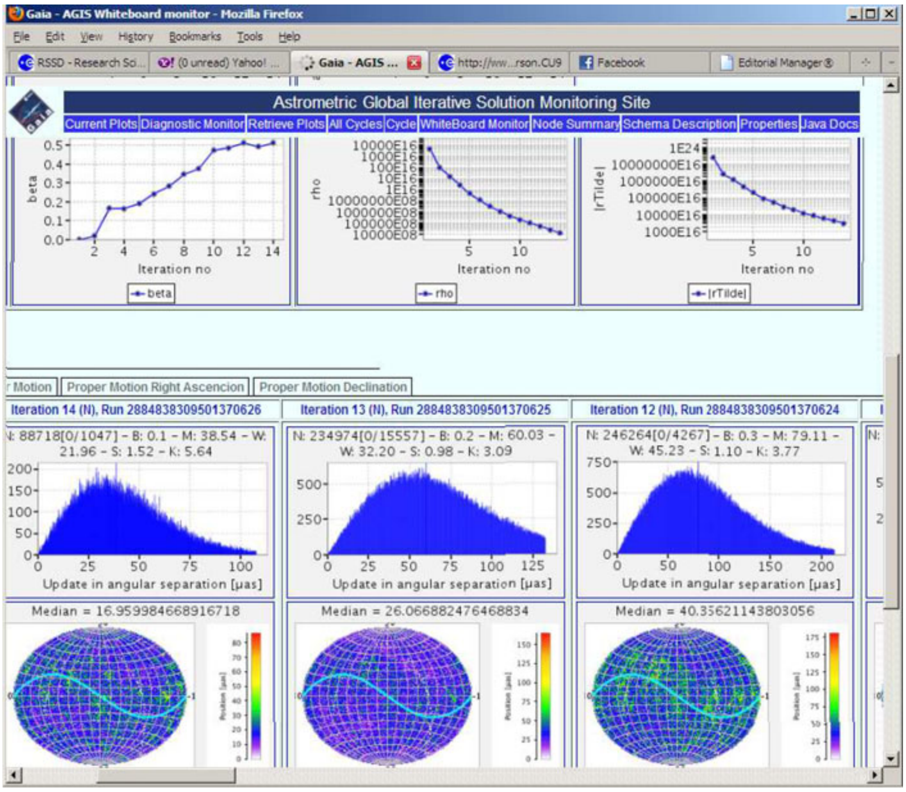
Fig. 11 The *SourceCollector* is attached to each *DataTrain* and has a *SourceUpdater* associated with it to update all sources on disk when a job is finished

finished. In this manner a job is either completed or not, since the transaction may be 'rolled back' without consequence if there is some problem.

When the job is finished the *SourceCollector* sends all of the updated sources to the *GisConvergenceMonitor*. This call is made using RMI. Because the *GisConvergenceMonitor* receives sources throughout the iteration, histograms of the updates can be dynamically generated. These are displayed on the associated AGIS website in real time. An example of the website is shown in Fig. 12.

7.2 Attitude update

The attitude specifies the instantaneous orientation of Gaia in the same celestial reference frame as used for the astrometric parameters—for Gaia this is known as the Center-of-Mass Reference System (CoMRS). Being the local rest frame of Gaia, the axes of the CoMRS are aligned with the International



Iteration 14 (N), Run 2884838309501370626

N: 88718[0/1047] - B: 0.1 - M: 38.54 - W: 21.96 - S: 1.52 - K: 5.64

Median = 16.959984668916718

Iteration 13 (N), Run 2884838309501370625

N: 234974[0/15557] - B: 0.2 - M: 60.03 - W: 32.20 - S: 0.98 - K: 3.09

Median = 26.066882476468834

Iteration 12 (N), Run 2884838309501370624

N: 246264[0/4267] - B: 0.3 - M: 79.11 - W: 45.23 - S: 1.10 - K: 3.77

Median = 40.35621143803056

Fig. 12 Update plots such as for the source position update shown here are generated dynamically and displayed on the AGIS monitoring website while the system is running. The Conjugate Gradients parameters such as ρ and $|\tilde{\mathbf{r}}|$ (see [17], for details) are also tracked as shown. Historical plots may be retrieved from the system

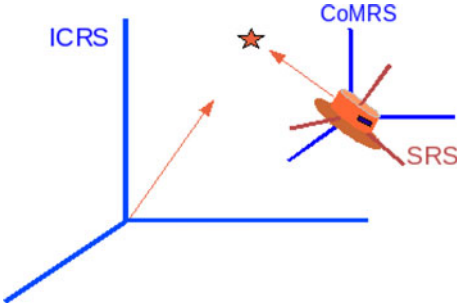


Fig. 13 The position of a source is initially given in the ICRS, centred on the solar-system barycentre, and is then transformed to the Gaia-centred CoMRS by taking into account parallax and relativistic effects. Finally the position may be transformed to the SRS (Scanning Reference System), which is fixed to the spinning instrument, by means of a rotation given by the attitude quaternion **q**

Celestial Reference System (ICRS; [9]), but with Gaia as the origin of the coordinate system instead of the solar system barycentre (Fig. 13). While the CoMRS is an inertial frame, the Scanning Reference System (SRS) rotates with the satellite and the optical axes of the astrometric telescope are fixed in the SRS. To a first approximation, the CCDs therefore measure the positions of the sources in the SRS.

The CoMRS and SRS frames are related by a purely spatial rotation, which defines the instantaneous attitude of Gaia. We use quaternions [12] to represent the attitude, as is common practice for spacecraft (e.g., [14]). The quaternion **q** is a 4-vector representing a direction in space (expressed in either the CoMRS or the SRS) and an angle of rotation around that direction. The four elements of the quaternion are continuous functions of time, here denoted $q^k(t)$, $k = 1 \dots 4$, which allow a singular-free attitude representation for arbitrary rotations. These functions are modelled as cubic splines, using short-range B-splines $B_n(t)$ as basis functions [4]; thus

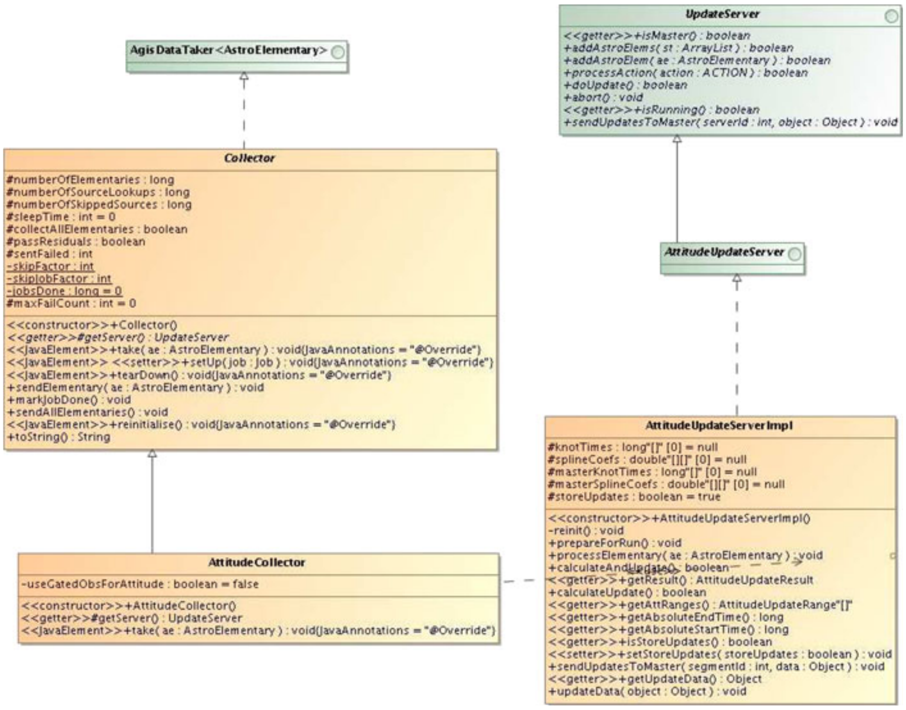
$$q^k(t) = \sum_n a_n^k B_n(t)\,, \tag{3}$$

where a_n^k are the attitude parameters, of which there are a few million in the system (see [17], for details). The attitude update solves a linearised least-squares problem similar to (2) but with the unknowns **d** now being the updates to the attitude parameters a_n^k and the partial derivatives in **A** being taken with respect to these attitude parameters. The dimension m is however very much greater in this case, since the attitude update in principle has to consider all the observations throughout the mission. However, thanks to the short range of the B-splines, the attitude normal matrix is band-diagonal, and the resulting system can be stored and solved very efficiently.

In fact the attitude may be divided into segments each of which can be solved simultaneously but separately. There will be natural breaks in Gaia's attitude that can be used to segment the data, but this technique may be used to distribute the attitude processing further. Hence, depending on the number of attitude segments, there is a limit to the distribution of attitude processing. Each segment may be solved on an individual processor. In actual fact the final fitting of the attitude for five years data as a single spline with knots every fifteen seconds took only 30 min on a Xeon processor with 16 GB of memory. The solution itself is not the bottleneck, but rather the gathering of the observations. With a single attitude update server all source observations must be passed to this server from every data train. Once the system surpassed 32 *DataTrain*s this became a limiting factor.

On each data train an *AttitudeCollector* is registered. This gathers all of the elementaries and passes them to the appropriate *AttitudeUpdateServer*. Appropriate here means the attitude server dealing with the time bin in which the observation falls. In some cases the segments overlap and an observation must be sent to two servers simultaneously. Again RMI is used for this passing and the observations carry the updated source parameters with them (Fig. 14).

The *AttitudeUpdateServer*(s) adds to the partial equations for each observation passed. It must wait until the end of the run to ensure all observations have



UpdateServer
<<getter>>+isMaster() : boolean
+addAstroElems(st : ArrayList) : boolean
+addAstroElem(ae : AstroElementary) : boolean
+processAction(action : ACTION) : boolean
+doUpdate() : boolean
+abort() : void
<<getter>>+isRunning() : boolean
+sendUpdatesToMaster(serverId : int, object : Object) : void
Agis DataTaker <AstroElementary>
Collector
#numberOfElementaries : long
#numberOfSourceLookups : long
#numberOfSkippedSources : long
#sleepTime : int = 0
#collectAllElementaries : boolean
#passResiduals : boolean
#sentFailed : int
-skipFactor : int
-skipJobFactor : int
-jobsDone : long = 0
#maxFailCount : int = 0
<<constructor>>+Collector()
<<getter>>#getServer() : UpdateServer
<<JavaElement>>+take(ae : AstroElementary) : void{JavaAnnotations = "@Override"}
<<JavaElement>> <<setter>>+setUp(job : Job) : void{JavaAnnotations = "@Override"}
<<JavaElement>>+tearDown() : void{JavaAnnotations = "@Override"}
+sendElementary(ae : AstroElementary) : void
+markJobDone() : void
+sendAllElementaries() : void
<<JavaElement>>+reinitialise() : void{JavaAnnotations = "@Override"}
+toString() : String
AttitudeUpdateServer
AttitudeUpdateServerImpl
#knotTimes : long"[]" [0] = null
#splineCoefs : double"[][]" [0] = null
#masterKnotTimes : long"[]" [0] = null
#masterSplineCoefs : double"[][]" [0] = null
#storeUpdates : boolean = true
<<constructor>>+AttitudeUpdateServerImpl()
-reinit() : void
+prepareForRun() : void
+processElementary(ae : AstroElementary) : void
+calculateAndUpdate() : boolean
<<getter>>+getResult() : AttitudeUpdateResult
+calculateUpdate() : boolean
<<getter>>+getAttRanges() : AttitudeUpdateRange"[]"
<<getter>>+getAbsoluteEndTime() : long
<<getter>>+getAbsoluteStartTime() : long
<<getter>>+isStoreUpdates() : boolean
<<setter>>+setStoreUpdates(storeUpdates : boolean) : void
+sendUpdatesToMaster(segmentId : int, data : Object) : void
<<getter>>+getUpdateData() : Object
+updateData(object : Object) : void
AttitudeCollector
-useGatedObsForAttitude : boolean = false
<<constructor>>+AttitudeCollector()
<<getter>>#getServer() : UpdateServer
<<JavaElement>>+take(ae : AstroElementary) : void{JavaAnnotations = "@Override"}

Fig. 14 UML diagram for the distributed attitude update. The calibration update has similar classes

been seen before doing the final computation. The end of the run is signalled, via RMI, by the *RunManager*. At this point the updated spline coefficients are calculated and written to the *Store*. The server also now sends the updated attitude to the *ConvergenceMonitor* so it may be plotted on the website.

7.3 Calibration update

The geometric calibration model deals with the precise placement and orientation of the CCDs in the focal plane. Within the optical system light bounces off six highly polished mirrors before hitting the CCDs in the Focal Plane Assembly. Since there are no on-board calibration devices a distortion in a mirror is indistinguishable from a displacement of a CCD. In both cases the image centroid will not appear where it should be. This also means that any such shift can be modelled in terms of CCD orientation, ignoring the mirrors entirely, and this is precisely what we do in AGIS.

Geometric calibration parameters for the CCDs, such as orientation, scale and mechanical distortions, are defined on timescales of hours or months as needed and are know as *CalibrationEffect*s. This transformation for Gaia is quite involved (see [1]), yet for our purposes we may consider an instantaneous position η^{obs} for the source in the field of view. We define the astrometric calibrations in the following generalised form:

$$\eta_l^{\mathrm{obs}} = \eta_n^0 + \sum_r E_r(l)\,, \tag{4}$$

where l is the observation index and each of the $E_r(l)$ represents one basic *CalibrationEffect*, being a linear combination of calibration functions $\Phi_{rs}(l)$:

$$E_r(l) = \sum_s c_{rs}\Phi_{rs}(l)\,. \tag{5}$$

The coefficients c_{rs} constitute the whole set of calibration parameters. In the calibration update we solve these coefficients by a least-squares system similar to (2).

The functions Φ_{rs} receive the observation index l and it is assumed that this index suffices to derive whatever dependencies are needed to evaluate the corresponding function/effect for this observation. Examples of such dependencies are: the telescope index (preceding/following field of view); CCD row number; CCD strip number; pixel column within the CCD; time; relevant astrometric, photometric, and spectroscopic source parameters; auxiliary parameters (e.g., optical background level, illumination history of the pixel column). In this generic calibration scheme the dependencies are not hardcoded, and we do not know exactly how many calibration parameters there will be in the mission. Furthermore the calibration effects are all specified in an XML file allowing for easy addition (or removal) of specific effects in an AGIS execution.

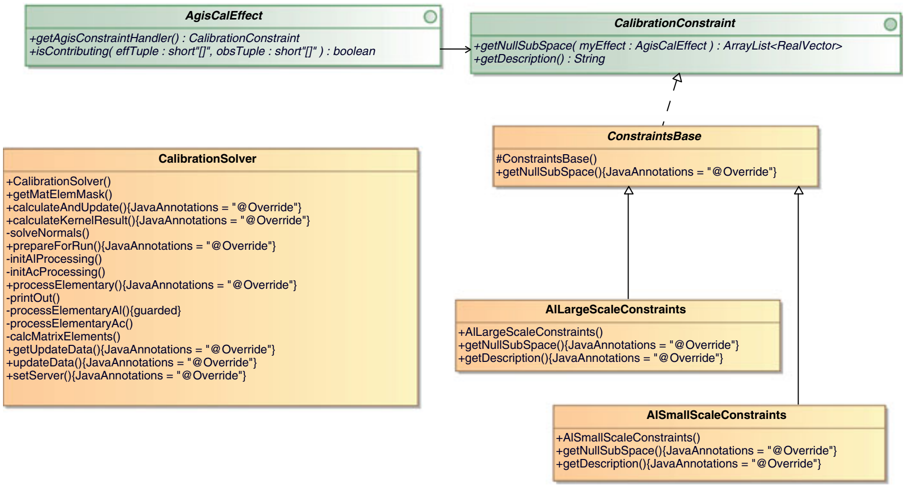

AgisCalEffect
+getAgisConstraintHandler() : CalibrationConstraint
+isContributing(effTuple : short"[]", obsTuple : short"[]") : boolean
CalibrationConstraint
+getNullSubSpace(myEffect : AgisCalEffect) : ArrayList<RealVector>
+getDescription() : String
CalibrationSolver
+CalibrationSolver()
+getMatElemMask()
+calculateAndUpdate(){JavaAnnotations = "@Override"}
+calculateKernelResult(){JavaAnnotations = "@Override"}
-solveNormals()
+prepareForRun(){JavaAnnotations = "@Override"}
-initAlProcessing()
-initAcProcessing()
+processElementary(){JavaAnnotations = "@Override"}
-printOut()
-processElementaryAl(){guarded}
-processElementaryAc()
-calcMatrixElements()
+getUpdateData(){JavaAnnotations = "@Override"}
+updateData(){JavaAnnotations = "@Override"}
+setServer(){JavaAnnotations = "@Override"}
ConstraintsBase
#ConstraintsBase()
+getNullSubSpace(){JavaAnnotations = "@Override"}
AlLargeScaleConstraints
+AlLargeScaleConstraints()
+getNullSubSpace(){JavaAnnotations = "@Override"}
+getDescription(){JavaAnnotations = "@Override"}
AlSmallScaleConstraints
+AlSmallScaleConstraints()
+getNullSubSpace(){JavaAnnotations = "@Override"}
+getDescription(){JavaAnnotations = "@Override"}

Fig. 15 UML for Calibration Effects (*AgisCalEffect*)

Following the terminology introduced in [17], the calibration parameters can be grouped into *calibration units* that can be handled separately because any given observation *l* can only belong to one calibration unit. Within a calibration unit, on the other hand, each observation typically contribute to many different effects, for example to irregularities both on a large scale (e.g., between CCDs) and on a small scale (e.g., between pixel columns). Our estimate is that no calibration unit will have more than about 10,000 parameters, which is negligible compared to the attitude parameters. Still, the memory requirements in the calibration block are larger than in the attitude update because there is no obvious way to exploit the sparseness of the normal matrix within each calibration unit. The CalibrationEffects are depicted using UML in Fig. 15.

From the perspective of distributed processing one must consider that, unlike attitude, here an observation will end up going to many calibration effects, e.g., both the large-scale and the small-scale calibration. We may however process all effects for a row of CCDs on a separate machine. The processing for calibration is not a huge overhead; as for attitude, the main bottleneck is the sending of all observations to the calibration server(s). Unlike the attitude, the calibration server can process the incoming observations more quickly and it has not been an overall bottleneck in the system.

The framework is similar to that of the attitude update. A *CalibrationCollector* is registered with each *DataTrain*, collects the required observation information and sends it to the *CalibrationUpdateServer* via RMI. The server accumulates the equations during the run and performs the final calculation when signalled by the *RunManager* that the run is complete. It writes the updated calibrations to the *Store* and sends them to the *GisConvergenceMonitor* for plotting on the website.

7.4 Global update

The global parameters, nominally some of the Parameterized Post-Newtonian (PPN) relativistic parameters, are estimated using a robust least-squares algorithm similar to (2) but now involving all the observations but only a (very) small set of parameters. The treatment is practically identical to a calibration parameter which spans the entire mission. As such it would be possible to combine this with the calibration update in a later version of the system, but for other reasons it is convenient to separate these terms, for example, to more easily estimate their correlations. As in the case for the attitude and calibration blocks we also have a *GlobalCollector* and a *GlobalUpdateManager* functioning in the same manner as described previously.

We will have sufficient observations and full sky coverage to decouple the global parameters from the astrometric parameters. Currently we only calculate PPN-γ due to solar system body deflection, but other variants will be added in the future, for example, separate and combined values of PPN-γ due to deflection by the major planets. The calculation of additional global parameters can provide a sanity check on the entire solution, i.e., a value wildly departing from the nominal value in the simulation data can only mean we are doing something very wrong somewhere.

7.5 Secondary source update

Nominally the entire data set could be put through AGIS, however we know that many binary stars and other complex objects will not work well with the simple observation model used. Hence only a fraction (between 10 and 50%) of the sources observed by Gaia are processed in AGIS. The selection of these *primary sources* will be done partly based on information from other parts of the processing chain (e.g., detected double stars), but mainly from the goodness-of-fit statistics gathered while performing a trial source update. If the fit is bad for the source, it is not accepted as a primary source but relegated to secondary source status. The selection of primary/secondary sources is itself an iterative process, which must be repeated after more accurate estimates have been obtained of the attitude and calibration parameters.

The AGIS solution, thus based on a 'clean' subset of the sources, provides an accurate celestial reference frame along with a correspondingly accurate attitude and geometric calibration. These outputs will be used to update the remaining fraction of the sources. This secondary star update is effectively identical to the source update block described in Section 7.1 but must only be run once over the data. This secondary solution will still not make sense for all types of objects (e.g., resolved binaries), which will be picked up in other parts of the processing chain.

8 Results

Some run times for the system are given in Table 1. AGIS has been running almost continuously since the end of 2005 on different simulated data sets. The current system requires around 40 iterations to remove initial (random and systematic) catalogue errors of about 100 milliarcseconds, based on the simulated observations. This level of initial errors is well above expected mission levels. After 40 iterations AGIS the source errors have been reduced to a level that is consistent with the observational noise level, i.e., some microarcsec for the brighter sources. Moreover, none of the systematic errors introduced in the starting values remain in the converged solution. A more comprehensive study of the results from AGIS will be provided in another paper.

9 Conclusion

The overall AGIS architecture and many of the components have been described in some detail. This is a software system designed and optimised to perform the Gaia astrometric data reduction involving the solution of a system with hundreds of millions of parameters and hundreds of billions of observations.

Advanced features of the Java language have been employed to make this system work well and remain very portable. Despite skepticism we have found Java reliable and robust, and sufficiently performant for our purposes. More work is needed in the coming years to further optimise AGIS, but a very good system is already in place and well understood.

Acknowledgements The constant work of the Gaia Data Processing and Analysis Consortium (DPAC) has played an important part in this work. We are particularly indebted to CU2 for the production of independently simulated Gaia-like data for use in the system. The data simulations have been done in the supercomputer Mare Nostrum at Barcelona Supercomputing Center – Centro Nacional de Supercomputación (The Spanish National Supercomputing Center). Research and development in Sweden is kindly supported by the Swedish National Space Board (SNSB).

Thanks to Xavier Luri for his guidance and input on early versions of the text.

Our special thanks go to Gaia's former Project Scientist Michael Perryman, whose vision, leadership, and enthusiasm in the early years of the project laid the foundations for the excellent progress that is today seen throughout DPAC and with AGIS in particular.

References

1. Bastian, U., Biermann, M.: Astrometric meaning and interpretation of high-precision time delay integration CCD data. A&A **438**, 745–755 (2005). doi:10.1051/0004-6361:20042372
2. Björck, Å.: Numerical Methods for Least Squares Problems. SIAM, Philadelphia, PA (1996)
3. Bombrun, A., Lindegren, L., Holl, B., Jordan, S.: Complexity of the Gaia astrometric least-squares problem and the (non-)feasibility of a direct solution method. A&A **516**, A77 (2010). doi:10.1051/0004-6361/200913503OPEN:
4. de Boor, C.: A Practical Guide to Splines, Rev. ed., Applied Mathematical Sciences, vol. 27 edn. Springer, Berlin Heidelberg New York (2001)

5. Danelutto, M., Dazzi, P.: A java/jini framework supporting stream parallel computations. In: Parallel Computing: Current and Future Issues of High-End Computing, pp. 681–688 (2006)
6. de Bruijne, J., Kohley, R., Prusti, T.: Gaia: 1,000 million stars with 100 CCD detectors. In: Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, vol. 7731 (2010). doi:10.1117/12.862062
7. Demichev, A., Foster, D., Kalyaev, V., Kryukov, A., Lamanna, M., Pose, V., Da Rocha, R.B., Wang, C.: Ogsa/globus evaluation for data intensive applications. ArXiv Computer Science e-prints. cs/0311009 (2003)
8. ESA: The Hipparcos and Tycho Catalogues. ESA, ESA SP-1200 (1997)
9. Feissel, M., Mignard, F.: The adoption of ICRS on 1 January 1998: meaning and consequences. A&A **331**, L33 (1998)
10. Foster, I.: Globus toolkit version 4: software for service oriented systems. J. Comput. Sci. Technol. **21**(4), 513–520 (2006)
11. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional Computing Series (1994)
12. Hamilton, W.R.: On quaternions. In: Proceedings of the Royal Irish Academy, vol. 3, pp. 1–16. http://www.maths.tcd.ie/pub/HistMath/People/Hamilton/Quatern2/Quatern2.html (1847)
13. Jordan, S.: The Gaia project: Technique, performance and status. Astron. Nachr. **329**, 875–880 (2008). doi:10.1002/asna.200811065. 0811.2345
14. Kane, T.R., Likins, P.W., Levinson, D.A.: Spacecraft Dynamics, 1st edn. McGraw Hill Book Company (1983)
15. Lindegren, L.: Gaia: astrometric performance and current status of the project. In: Klioner, S.A., Seidelmann, P.K., Soffel, M.H. (eds.) IAU Symposium, IAU Symposium, vol. 261, pp. 296–305 (2010). doi:10.1017/S1743921309990548
16. Lindegren, L., Babusiaux, C., Bailer-Jones, C., Bastian, U., Brown, A.G.A., Cropper, M., Høg, E., Jordi, C., Katz, D., van Leeuwen, F., Luri, X., Mignard, F., de Bruijne, J.H.J., Prusti, T.: The Gaia mission: science, organization and present status. In: Jin, W.J., Platais, I., Perryman, M.A.C. (eds.) IAU Symposium, vol. 248, pp. 217–223 (2008). doi:10.1017/S1743921308019133
17. Lindegren, L., O'Mullane, W., Lammers, U., Hobbs, D.: The astrometric core solution for the gaia mission. Astron. Astrophys. (2011, in prepreation)
18. Miller, W.W. III, Sontag, C., Rose, J.F.: OPUS: a CORBA pipeline for java, python, and perl applications. In: Payne, H.E., Jedrzejewski, R.I., Hook, R.N. (eds.) Astronomical Data Analysis Software and Systems XII, Astronomical Society of the Pacific Conference Series, vol. 295, pp. 261–264 (2003)
19. Nieto-Santisteban, M.A., Szalay, A.S., Thakar, A.R., O'Mullane, W.J., Gray, J., Annis, J.: When Database Systems Meet the Grid. ArXiv Computer Science e-prints. cs/0502018 (2005)
20. O'Mullane, W., Lindegren, L.: An object-oriented framework for GAIA data processing. Balt. Astron. **8**, 57–72 (1999)
21. O'Mullane, W., Li, N., Nieto-Santisteban, M., Szalay, A., Thakar, A., Gray, J.: Batch is back: casjobs, serving multi-tb data on the web. In: International Conference on Web Services, also MS technote. http://arxiv.org/pdf/cs.DC/0502072 (2005)
22. O'Mullane, W., Lammers, U., Bailer-Jones, C., Bastian, U., Brown, A., Drimmel, R., Eyer, L., Huc, C., Jansen, F., Katz, D., Lindegren, L., Pourbaix, D., Luri, X., Mignard, F., Torra, J., van Leeuwen, F.: Gaia Data Processing Architecture. ArXiv Astrophysics e-prints. astro-ph/0611885 (2006)
23. Rose, J., Akella, R., Binegar, S., Choo, T.H., Heller-Boyer, C., Hester, T., Hyde, P., Perrine, R., Rose, M.A., Steuerman, K.: The OPUS pipeline: a partially object-oriented pipeline system. In: Shaw, R.A., Payne, H.E., Hayes, J.J.E. (eds.) Astronomical Data Analysis Software and Systems IV, Astronomical Society of the Pacific Conference Series, vol. 77, pp. 429–432 (1995)
24. Schaefer, B.E.: The epoch of the constellations on the Farnese Atlas and their origin in Hipparchus's lost catalogue. J. Hist. Astron. **36**, 167–196 (2005)
25. Turon, C., O'Flaherty, K.S., Perryman, M.A.C. (eds.): The Three-Dimensional Universe with Gaia, vol. 576. ESA Special Publication (2005)
26. Wilkinson, B., Allen, M.: Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers. Prentice Hall (1999)

Astronomy & Astrophysics manuscript no. aa17905-11 © ESO 2011
December 20, 2011

The astrometric core solution for the Gaia mission

Overview of models, algorithms and software implementation

L. Lindegren[1], U. Lammers[2], D. Hobbs[1], W. O'Mullane[2], U. Bastian[3], and J. Hernández[2]

[1] Lund Observatory, Lund University, Box 43, SE-22100 Lund, Sweden
e-mail: `Lennart.Lindegren, David.Hobbs@astro.lu.se`

[2] European Space Agency (ESA), European Space Astronomy Centre (ESAC), P.O. Box (Apdo. de Correos) 78, ES-28691 Villanueva de la Cañada, Madrid, Spain
e-mail: `Uwe.Lammers, William.OMullane, Jose.Hernandez@sciops.esa.int`

[3] Astronomisches Rechen-Institut, Zentrum für Astronomie der Universität Heidelberg, Mönchhofstr. 12–14, DE-69120 Heidelberg, Germany
e-mail: `bastian@ari.uni-heidelberg.de`

Received 17 August 2011 / Accepted 25 November 2011

ABSTRACT

Context. The Gaia satellite will observe about one billion stars and other point-like sources. The astrometric core solution will determine the astrometric parameters (position, parallax, and proper motion) for a subset of these sources, using a global solution approach which must also include a large number of parameters for the satellite attitude and optical instrument. The accurate and efficient implementation of this solution is an extremely demanding task, but crucial for the outcome of the mission.

Aims. We aim to provide a comprehensive overview of the mathematical and physical models applicable to this solution, as well as its numerical and algorithmic framework.

Methods. The astrometric core solution is a simultaneous least-squares estimation of about half a billion parameters, including the astrometric parameters for some 100 million well-behaved so-called primary sources. The global nature of the solution requires an iterative approach, which can be broken down into a small number of distinct processing blocks (source, attitude, calibration and global updating) and auxiliary processes (including the frame rotator and selection of primary sources). We describe each of these processes in some detail, formulate the underlying models, from which the observation equations are derived, and outline the adopted numerical solution methods with due consideration of robustness and the structure of the resulting system of equations. Appendices provide brief introductions to some important mathematical tools (quaternions and B-splines for the attitude representation, and a modified Cholesky algorithm for positive semidefinite problems) and discuss some complications expected in the real mission data.

Results. A complete software system called AGIS (Astrometric Global Iterative Solution) is being built according to the methods described in the paper. Based on simulated data for 2 million primary sources we present some initial results, demonstrating the basic mathematical and numerical validity of the approach and, by a reasonable extrapolation, its practical feasibility in terms of data management and computations for the real mission.

Key words. Astrometry – Methods: data analysis – Methods: numerical – Space vehicles: instruments

1. Introduction

The space astrometry mission Gaia, planned to be launched by the European Space Agency (ESA) in 2013, will determine accurate astrometric data for about one billion objects in the magnitude range from 6 to 20. Accuracies of 8–25 micro-arcsec (μas) are typically expected for the trigonometric parallaxes, positions at mean epoch, and annual proper motions of simple (i.e., apparently single) stars down to 15th magnitude. The astrometric data are complemented by photometric and spectroscopic information collected with dedicated instruments on board the Gaia satellite. The mission will result in an astronomical database of unprecedented scope, accuracy and completeness becoming available to the scientific community around 2021.

The original interferometric concept for a successor mission to Hipparcos, called GAIA (Global Astrometric Interferometer for Astrophysics), was described by Lindegren & Perryman (1996) but has since evolved considerably by the incorporation of novel ideas (Høg 2008) and as a result of industrial studies conducted under ESA contracts (Perryman et al. 2001). The mission, now in the final integration phase with EADS Astrium as prime contractor, is no longer an interferometer but has retained the name Gaia, which is thus no acronym. For some brief but up-to-date overviews of the mission, see Lindegren et al. (2008) and Lindegren (2010). The scientific case is most comprehensively described in the proceedings of the conference The Three-Dimensional Universe with Gaia (Turon et al. 2005).

In parallel with the industrial development of the satellite, the Gaia Data Processing and Analysis Consortium (DPAC; Mignard et al. 2008) is charged with the task of developing and running a complete data processing system for analysing the satellite data and constructing the resulting database ('Gaia Catalogue'). This task is extremely difficult due to the large quantities of data involved, the complex relationships between different kinds of data (astrometric, photometric, spectroscopic) as well as between data collected at different epochs, the need for complex yet efficient software systems, and the interaction and sustained support of many individuals and groups over an extended period of time.

A fundamental part of the data processing task is the astrometric core solution, currently under development in DPAC's Coordination Unit 3 (CU3), 'Core Processing'. Mathematically, the astrometric core solution is a simultaneous determination of a very large number of unknowns representing three kinds of information: (i) the astrometric parameters for a subset of the observed stars, representing the astrometric reference frame; (ii) the instrument attitude, representing the accurate celestial pointing of the instrument axes in that reference frame as a function of time; and (iii) the geometric instrument calibration, representing the mapping from pixels on the CCD detectors to angular directions relative to the instrument axes. Although the astrometric core solution is only made for a subset of the stars, the resulting celestial reference frame, attitude and instrument calibration are fundamental inputs for the processing of all observations. Optionally, a fourth kind of unknowns, the global parameters, may be introduced to describe for example a hypothetical deviation from General Relativity.

We use the term 'source' to denote any astronomical object that Gaia detects and observes as a separate entity. The vast majority of the Gaia sources are ordinary stars, many of them close binaries or the components of wide systems, but some are non-stellar (for example asteroids and quasars). Nearly everywhere in this paper, one can substitute 'star' for 'source' without distortion; however, for consistency with established practice in the Gaia community we use 'source' throughout.

While the total number of distinct sources that will be observed by Gaia is estimated to slightly more than one billion, only a subset of them shall be used in the astrometric core solution. This subset, known as the 'primary sources', is selected to be astrometrically well-behaved (see Sect. 6.2) and consists of (effectively) single stars and extragalactic sources (quasars and AGNs) that are sufficiently stable and point-like. We assume here that the number of primary sources is about 10^8, i.e., roughly one tenth of the total number of objects, although in the end it is possible that an even larger number will be used.

In comparison with many other parts of the Gaia data processing, the astrometric core solution is in principle simple, mainly because it only uses a subset of the observations (namely, those of the primary sources), which can be accurately modelled in a relatively straightforward way. In practice, the problem is however formidable: the total number of unknowns is of the order of 5×10^8, the solution uses some 10^{11} individual observations extracted from some 70 Terabyte (TB) of raw satellite data, and the entangled nature of the data excludes a direct solution. A feasible approach has nevertheless been found, including a precise mathematical formulation, practical solution method, and efficient software implementation. It is the aim of this paper to provide a comprehensive overview of this approach.

Concerning notations we have followed the usual convention to denote all non-scalar entities (vectors, tensors, matrices, quaternions) by boldfaced characters. Lower-case bold italics ($\boldsymbol{a}$) are used for vectors and one-dimensional column matrices; upper-case bold italics ($\boldsymbol{A}$) usually denote two-dimensional matrices. Following Murray (1983) the prime ($'$) signifies both matrix transpose ($\boldsymbol{a}'$, $\boldsymbol{A}'$) and scalar multiplication of vectors; thus $||a|| = (\boldsymbol{a}'\boldsymbol{a})^{1/2}$ defines the magnitude of the vector $\boldsymbol{a}$ as well as the Euclidean norm of the column matrix $\boldsymbol{a}$. Angular brackets denote normalization to unit length, as in $\langle \boldsymbol{x} \rangle = \boldsymbol{x}||\boldsymbol{x}||^{-1}$. In this notation, no special distinction is thus made between vectors as physical entities (also known as geometric, spatial or Euclidean vectors) on one hand, and their numerical representations in some coordinate system as column matrices (also known as list vectors) on the other hand. Moreover, list vectors can of course have any dimension: $\boldsymbol{a} \in \mathbb{R}^n$. In the coordinate system whose axes are aligned with unit vectors $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, the components of the arbitrary vector $\boldsymbol{a}$ are given by $a_x = \boldsymbol{x}'\boldsymbol{a}$, $a_y = \boldsymbol{y}'\boldsymbol{a}$, and $a_z = \boldsymbol{z}'\boldsymbol{a}$; if the coordinate system is represented by the vector triad $\mathsf{S} = [\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$, these components are given by the column matrix $\mathsf{S}'\boldsymbol{a}$ (cf. Appendix A in Murray 1983). This notation, although perhaps unfamiliar to many readers, provides a convenient and unambiguous framework for representing and transforming spatial vectors in different coordinate systems. For a vector-valued function $\boldsymbol{f}(\boldsymbol{x})$, $\partial \boldsymbol{f}/\partial \boldsymbol{x}'$ denotes the $\dim(\boldsymbol{f}) \times \dim(\boldsymbol{x})$ matrix whose (i, j)th element is $\partial f_i/\partial x_j$. Quaternions follow their own algebra (see Appendix A for a brief introduction) and must not be confused with vectors/matrices; quaternions are therefore denoted by bold Roman characters ($\mathbf{a}$). When taking a derivative with respect to the quaternion $\mathbf{a}$, $\partial x/\partial \mathbf{a}$ denotes the 4×1 matrix of derivatives with respect to the quaternion components; $\partial x/\partial \mathbf{a}'$ is the transposed matrix. Tables of acronyms and variables are provided in Appendix E.

2. Outline of the approach

The astrometric principles for Gaia were outlined already in the Hipparcos Catalogue (ESA 1997, Vol. 3, Ch. 23) where, based on the accumulated experience of the Hipparcos mission, a view was cast to the future. The general principle of a global astrometric data analysis was succinctly formulated as the minimization problem:

$$\min_{\boldsymbol{s},\, \boldsymbol{n}} ||\boldsymbol{f}^{\text{obs}} - \boldsymbol{f}^{\text{calc}}(\boldsymbol{s}, \boldsymbol{n})||_{\mathcal{M}}\,, \tag{1}$$

with a slight change of notation to adapt to the present paper. Here $\boldsymbol{s}$ is the vector of unknowns (parameters) describing the barycentric motions of the ensemble of sources used in the astrometric solution, and $\boldsymbol{n}$ is a vector of 'nuisance parameters' describing the instrument and other incidental factors which are not of direct interest for the astronomical problem but are nevertheless required for realistic modelling of the data. The observations are represented by the vector $\boldsymbol{f}^{\text{obs}}$ which could for example contain the measured detector coordinates of all the stellar images at specific times. $\boldsymbol{f}^{\text{calc}}(\boldsymbol{s}, \boldsymbol{n})$ is the observation model, e.g., the expected detector coordinates calculated as functions of the astrometric and nuisance parameters. The norm is calculated in a metric $\mathcal{M}$ defined by the statistics of the data; in practice the minimization will correspond to a weighted least-squares solution with due consideration of robustness issues (see Sect. 3.6).

While Eq. (1) encapsulates the global approach to the data analysis problem, its actual implementation will of course depend strongly on the specific characteristics of the Gaia instrument and mission as well as on the practical constraints on the data processing task.

At the heart of the problem is the modelling of data represented by the function $\boldsymbol{f}^{\text{calc}}(\boldsymbol{s}, \boldsymbol{n})$ in Eq. (1). This is schematically represented in the shaded part of the diagram in Fig. 1. It shows the main steps for calculating the expected CCD output in terms of the various parameters. The data processing, effecting the minimization in Eq. (1), is represented in the right part of the diagram. In subsequent sections we describe in some detail the main elements depicted in Fig. 1. The astrometric (source) parameters are represented by the vector $\boldsymbol{s}$, while the nuisance parameters are of three kinds: the attitude parameters $\boldsymbol{a}$, the geometric calibration parameters $\boldsymbol{c}$, and the global parameters $\boldsymbol{g}$. The observables $\boldsymbol{f}$ are represented by the pixel coordinates (κ, μ) of point source images on Gaia's CCDs. (In the actual implementation of the approach, the minimization problem is formu-

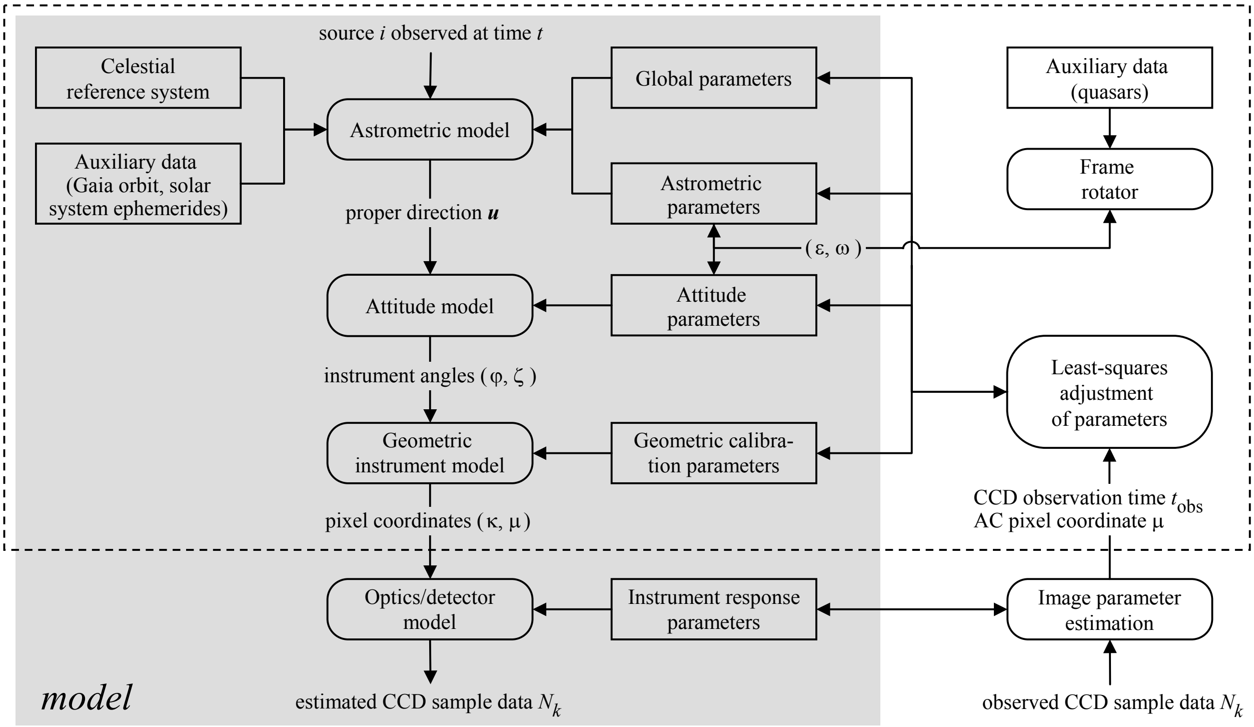

source i observed at time t
Celestial reference system
Auxiliary data (Gaia orbit, solar system ephemerides)
Astrometric model
Global parameters
Astrometric parameters
Auxiliary data (quasars)
Frame rotator
proper direction $\boldsymbol{u}$
(ε, ω)
Attitude model
Attitude parameters
instrument angles (φ, ζ)
Least-squares adjustment of parameters
Geometric instrument model
Geometric calibration parameters
CCD observation time $t_{\rm obs}$
AC pixel coordinate μ
pixel coordinates (κ, μ)
Optics/detector model
Instrument response parameters
Image parameter estimation
model
estimated CCD sample data N_k
observed CCD sample data N_k

Fig. 1. Schematic representation of the main elements of the astrometric data analysis. In the shaded area is the mathematical model that allows to calculate the position of the stellar image in pixel coordinates, and hence the expected CCD data, for any given set of model parameters. To the right are the processes that fit this model to the observed CCD data by adjusting the parameters in the rectangular boxes along the middle. This paper is primarily concerned with the geometrical part of the analysis contained in the dashed box. However, a brief outline of the CCD data modelling and processing (bottom part of the diagram) is given in Sect. 3.5 and Appendix D.2.

lated in terms of the field angles η, ζ rather than in the directly measured pixel coordinates κ, μ; see Sects. 3.3 and 3.4.)

The various elements of the astrometric solution are described in some detail in the subsequent sections. Section 3 provides the mathematical framework needed to model the Gaia observations and setting up the least-squares equations for the astrometric solution. In the interest of clarity and overview we omit in this description certain complications that need to be considered in the final data processing system; these are instead briefly discussed in Appendix D. In Sect. 4 we describe the iterative solution method in general terms, and then provide, in Sects. 5 and 6, the mathematical details of the most important procedures. Section 7 outlines the existing implementation of the solution and presents the results of a demonstration run based on simulated observations of about 2 million primary sources. Appendices A to C provide brief introductions to three mathematical tools that are particularly important for the subsequent development, namely the use of quaternions for representing the instantaneous satellite attitude, the B-spline formalism used to model the attitude as a function of time, and a modified Cholesky algorithm for the decomposition of positive semidefinite normal matrices.

3. Mathematical formulation of the basic observation model

3.1. Reference systems

The high astrometric accuracy aimed for with Gaia makes it necessary to use general relativity for modelling the data. This implies a precise and consistent formulation of the different reference systems used to describe the motion of the observer (Gaia), the motion of the observed object (source), the propagation of light from the source to the observer, and the transformations between these systems. The formulation adopted for Gaia (Klioner 2003, 2004) is based on the parametrized post-Newtonian (PPN) version of the relativistic framework adopted in 2000 by the International Astronomical Union (IAU); see Soffel et al. (2003). In this section only some key concepts from this formulation are introduced.

The orbit of Gaia and the light propagation from the source to Gaia are entirely modelled in the Barycentric Celestial Reference System (BCRS) whose spatial axes are aligned with the International Celestial Reference System (ICRS, Feissel & Mignard 1998). The associated time coordinate is the barycentric coordinate time (TCB). Throughout this paper, all time variables denoted t (with various subscripts) must be interpreted as TCB. The ephemerides of solar-system bodies (including the Sun and the Earth) are also expressed in this reference system. Even the motions of the stars and extragalactic objects are formally modelled in this system, although for practical reasons certain effects of the light-propagation time are conventionally ignored in this model (Sect. 3.2).

In order to model the rotation (attitude) of Gaia and the celestial direction of the light rays as observed by Gaia, it is expedient to introduce also a co-moving celestial reference system having its origin at the centre of mass of the satellite and a coordinate time equal to the proper time at Gaia. This is known as the Centre-of-Mass Reference System (CoMRS) and the associated time coordinate is Gaia Time (T_G). Klioner (2004) demonstrates how the CoMRS can be constructed in the IAU 2000 framework in exact analogy with the Geocentric Celestial Reference System (GCRS), only for a massless particle (Gaia) instead of the Earth. Like the BCRS and GCRS, the CoMRS is kinematically non-rotating, and its spatial axes are aligned with the ICRS. The celestial reference system (either the CoMRS or the ICRS depending on whether the origin is at Gaia or the solar-system barycentre) will in the following be represented by the vector triad $\mathsf{C} = [\boldsymbol{X}\,\boldsymbol{Y}\,\boldsymbol{Z}]$, where $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ are orthogonal unit vectors aligned with the axes of the celestial reference system. That is, $\boldsymbol{X}$ points towards the origin ($\alpha = \delta = 0$), $\boldsymbol{Z}$ towards the north celestial pole ($\delta = +90^\circ$), and $\boldsymbol{Y} = \boldsymbol{Z} \times \boldsymbol{X}$ points to ($\alpha = 90^\circ, \delta = 0$).

In the CoMRS the attitude of the satellite is a spatial rotation in three dimensions, and can therefore be described purely classically, for example using quaternions (Sect. 3.3 and Appendix A). The rotated reference system, aligned with the instrument axes, is known as the Scanning Reference System (SRS). Its spatial $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ axes (Fig. 2) are defined in terms of the two viewing directions of Gaia $\boldsymbol{f}_\mathrm{P}$ (in the centre of the preceding field of view, PFoV) and $\boldsymbol{f}_\mathrm{F}$ (in the centre of the following field of view, FFoV) as

$$\boldsymbol{x} = \langle \boldsymbol{f}_\mathrm{F} + \boldsymbol{f}_\mathrm{P} \rangle\,, \quad \boldsymbol{z} = \langle \boldsymbol{f}_\mathrm{F} \times \boldsymbol{f}_\mathrm{P} \rangle\,, \quad \boldsymbol{y} = \boldsymbol{z} \times \boldsymbol{x}\,. \qquad (2)$$

(The precise definition of $\boldsymbol{f}_\mathrm{P}$ and $\boldsymbol{f}_\mathrm{F}$ is implicit in the geometric instrument model; see Sect. 3.4.) The SRS is represented by the vector triad $\mathsf{S} = [\boldsymbol{x}\,\boldsymbol{y}\,\boldsymbol{z}]$.

For determining the orbit of Gaia and calibrating the onboard clock, it is also necessary to model the radio ranging and other ground-based observations of the Gaia spacecraft in the same relativistic framework. For this, we also need the GCRS. These aspects of the Gaia data processing are, however, not discussed in this paper.

3.2. Astrometric model

The astrometric model is a recipe for calculating the proper direction $\boldsymbol{u}_i(t)$ to a source (i) at an arbitrary time of observation (t) in terms of its astrometric parameters $\boldsymbol{s}_i$ and various auxiliary data, assumed to be known with sufficient accuracy. The auxiliary data include an accurate barycentric ephemeris of the Gaia satellite, $\boldsymbol{b}_\mathrm{G}(t)$, including its coordinate velocity $\mathrm{d}\boldsymbol{b}_\mathrm{G}/\mathrm{d}t$, and ephemerides of all other relevant solar-system bodies.

For the astrometric core solution every source is assumed to move with uniform space velocity relative to the solar-system barycentre. In the BCRS its spatial coordinates at time t_* are therefore given by

$$\boldsymbol{b}_i(t_*) = \boldsymbol{b}_i(t_\mathrm{ep}) + (t_* - t_\mathrm{ep})\boldsymbol{v}_i \qquad (3)$$

where t_ep is an arbitrary reference epoch and $\boldsymbol{b}_i(t_\mathrm{ep})$, $\boldsymbol{v}_i$ define six kinematic parameters for the motion of the source. The six astrometric parameters in $\boldsymbol{s}_i$ are merely a transformation of the kinematic parameters into an equivalent set better suited for the analysis of the observations. The six astrometric parameters are:

α_i the barycentric right ascension at the adopted reference epoch t_ep

δ_i the barycentric declination at epoch t_ep

ϖ_i the annual parallax at epoch t_ep

$\mu_{\alpha*i} = (\mathrm{d}\alpha_i/\mathrm{d}t)\cos\delta_i$ the proper motion in right ascension at epoch t_ep

$\mu_{\delta i} = \mathrm{d}\delta_i/\mathrm{d}t$ the proper motion in declination at epoch t_ep

$\mu_{ri} = v_{ri}\varpi_i/A_\mathrm{u}$ the 'radial proper motion' at epoch t_ep, i.e., the radial velocity of the source (v_{ri}) expressed in the same unit as the transverse components of proper motion (A_u = astronomical unit).

As explained in Sect. 3.1, the astrometric parameters refer to the ICRS and the time coordinate used is TCB. The reference epoch t_ep is preferably chosen to be near the mid-time of the mission in order to minimize statistical correlations between the position and proper motion parameters.

The transformation between the kinematic and the astrometric parameters is non-trivial (Klioner 2003), mainly as a consequence of the practical need to neglect most of the light-propagation time $t - t_*$ between the emission of the light at the source (t_*) and its reception at Gaia (t). This interval is typically many years and its value, and rate of change (which depends on the radial velocity of the source), will in general not be known with sufficient accuracy to allow modelling of the motion of the source directly in terms of its kinematic parameters according to Eq. (3). The proper motion components $\mu_{\alpha*i}$, $\mu_{\delta i}$ and radial velocity v_{ri} correspond to the 'apparent' quantities discussed by Klioner (2003, Sect. 8).

The coordinate direction to the source at time t is calculated with the same 'standard model' as was used for the reduction of the Hipparcos observations (ESA 1997, Vol. 1, Sect. 1.2.8), namely

$$\bar{\boldsymbol{u}}_i(t) = \langle \boldsymbol{r}_i + (t_\mathrm{B} - t_\mathrm{ep})(\boldsymbol{p}_i\mu_{\alpha*i} + \boldsymbol{q}_i\mu_{\delta i} + \boldsymbol{r}_i\mu_{ri}) - \varpi_i\boldsymbol{b}_\mathrm{G}(t)/A_\mathrm{u} \rangle \qquad (4)$$

where the angular brackets signify vector length normalization, and $[\boldsymbol{p}_i\ \boldsymbol{q}_i\ \boldsymbol{r}_i]$ is the 'normal triad' of the source with respect to the ICRS (Murray 1983). In this triad, $\boldsymbol{r}_i$ is the barycentric coordinate direction to the source at time t_ep, $\boldsymbol{p}_i = \langle \boldsymbol{Z} \times \boldsymbol{r}_i \rangle$, and $\boldsymbol{q}_i = \boldsymbol{r}_i \times \boldsymbol{p}_i$. The components of these unit vectors in the ICRS are given by the columns of the matrix

$$\mathsf{C}'[\boldsymbol{p}_i\ \boldsymbol{q}_i\ \boldsymbol{r}_i] = \begin{bmatrix} -\sin\alpha_i & -\sin\delta_i\cos\alpha_i & \cos\delta_i\cos\alpha_i \\ \cos\alpha_i & -\sin\delta_i\sin\alpha_i & \cos\delta_i\sin\alpha_i \\ 0 & \cos\delta_i & \sin\delta_i \end{bmatrix}. \qquad (5)$$

$\boldsymbol{b}_\mathrm{G}(t)$ is the barycentric position of Gaia at the time of observation, and A_u the astronomical unit. t_B is the barycentric time obtained by correcting the time of observation for the Römer delay; to sufficient accuracy it is given by

$$t_\mathrm{B} = t + \boldsymbol{r}_i'\boldsymbol{b}_\mathrm{G}(t)/c\,, \qquad (6)$$

where c is the speed of light.

In Eq. (4) the term containing μ_{ri} accounts for the relative rate of change of the barycentric distance to the source. This term may produce secular variations of the proper motions and parallaxes of some nearby stars, which in principle allow their radial velocities to be determined astrometrically (Dravins et al. 1999). However, for the vast majority of these stars, μ_{ri} can be more accurately calculated by combining the spectroscopically measured radial velocities with the astrometric parallaxes. Thus, although all six astrometric parameters are taken into account when computing the coordinate direction, usually only five of them are considered as unknowns in the astrometric solution.

The standard model can be derived by considering the uniform motion of the source in a purely classical framework, using

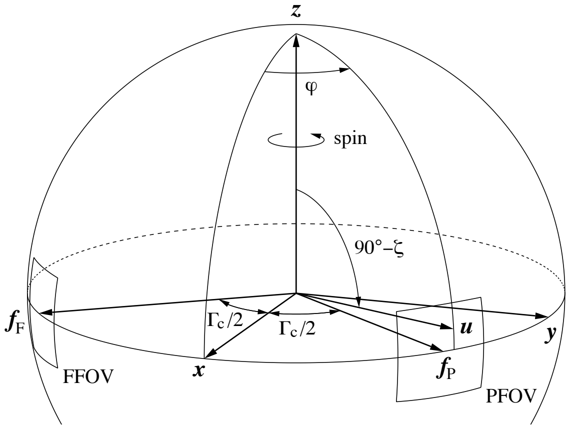


Fig. 2. The Scanning Reference System (SRS) $[\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$ is defined by the viewing directions $\boldsymbol{f}_{\rm P}$ and $\boldsymbol{f}_{\rm F}$ according to Eq. (2). The instrument angles (φ, ζ) are the spherical coordinates in the SRS of the observed (proper) direction $\boldsymbol{u}$ towards the object. See Fig. 4 for further definition of the viewing directions. The field sizes are greatly exaggerated in this drawing; in reality the astrometrically useful field is 0.72° × 0.69° (along × across scan) for each viewing direction. The basic angle is $\Gamma_{\rm c} = \arccos(\boldsymbol{f}_{\rm P}'\boldsymbol{f}_{\rm F})$, nominally equal to 106.5°.

Euclidean coordinates and neglecting the light propagation time from the source to the observer (except for the Römer delay). To the accuracy of Gaia, relativistic and light-propagation effects are by no means negligible, but it can be shown that this model is nevertheless accurate enough to *model* the observations to sub-microarcsec accuracy. It is adopted for this purpose because it closely corresponds to the conventional interpretation of the astrometric parameters. However, when the astrometric parameters are to be *interpreted* in terms of the barycentric space velocity of the source, some of these effects may come into play (Lindegren & Dravins 2003).

The transformation from $\bar{\boldsymbol{u}}_i(t)$ to the observable (proper) direction $\boldsymbol{u}_i(t)$ involves taking into account gravitational light deflection by solar-system bodies and the Lorentz transformation to the co-moving frame of the observer (stellar aberration); the relevant formulae are given by Klioner (2003). This transformation therefore depends also on the global parameters $\boldsymbol{g}$, for example the PPN parameter γ which measures the strength of the gravitational light deflection. The calculation uses some auxiliary data, not subject to improvement by the solution, and which are here denoted $\boldsymbol{h}$; normally they include for example the barycentric ephemerides of the Gaia satellite and of solar-system bodies, along with their masses. The complete transformation can therefore be written symbolically as:

$$\boldsymbol{u}_i(t) = \boldsymbol{u}(\boldsymbol{s}_i, \boldsymbol{g} \,|\, t, \boldsymbol{h}) \,, \tag{7}$$

where the vertical bar formally separates the (updatable) parameters $\boldsymbol{s}_i$ and $\boldsymbol{g}$ from the (fixed) given data t and $\boldsymbol{h}$. Strictly speaking, the function $\boldsymbol{u}$ returns the coordinates of the proper direction in the CoMRS, that is the column matrix $\mathsf{C}'\boldsymbol{u}_i(t)$. The source parameter vector $\boldsymbol{s}$ is the concatenation of the sub-vectors $\boldsymbol{s}_i$ for all the primary sources.

3.3. Attitude model

The attitude specifies the instantaneous orientation of the Gaia instrument in the celestial reference frame, that is the transformation from $\mathsf{C} = [\boldsymbol{X}\ \boldsymbol{Y}\ \boldsymbol{Z}]$ (more precisely, the CoMRS) to $\mathsf{S} = [\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$ (Sect. 3.1) as a function of time. The spacecraft is controlled to follow a specific attitude as a function of time, for example the so-called Nominal Scanning Law (NSL; de Bruijne et al. 2010), designed to provide good coverage of the whole celestial sphere while satisfying a number of other requirements. The NSL is analytically defined for arbitrarily long time intervals by just a few free parameters.

However, the actual attitude will deviate from the intended (nominal) scanning law by up to ∼ 1 arcmin in all three axes, and these deviations vary on time scales from seconds to minutes depending on the level of physical perturbations and the characteristics of the real-time attitude measurements and control law (cf. Appendix D.4). The CCD integration time of (usually) 4.42 s means that the true (physical) attitude cannot be observed, but only a smoothed version of it, corresponding to the convolution of the physical attitude with the CCD exposure function (Appendix D.3). This 'effective' attitude must however be a posteriori estimable, at any instant, to an accuracy compatible with the astrometric goals, i.e., at sub-mas level. For this purpose the effective attitude is modelled in terms of a finite number of attitude parameters, for which it is necessary to choose a suitable mathematical representations of the instantaneous transformation, as well as of its time dependence. For the Gaia data processing, we have chosen to use quaternions (Appendix A) for the former, and B-splines (Appendix B) for the latter representation.

At any time the orientation of the SRS (S) with respect to the CoMRS (C) may be represented by the attitude matrix

$$\boldsymbol{A} \equiv \begin{bmatrix} A_{xx} & A_{xy} & A_{xz} \\ A_{yx} & A_{yy} & A_{yz} \\ A_{zx} & A_{zy} & A_{zz} \end{bmatrix} = \mathsf{S}'\mathsf{C} = \begin{bmatrix} \boldsymbol{x}'\boldsymbol{X} & \boldsymbol{x}'\boldsymbol{Y} & \boldsymbol{x}'\boldsymbol{Z} \\ \boldsymbol{y}'\boldsymbol{X} & \boldsymbol{y}'\boldsymbol{Y} & \boldsymbol{y}'\boldsymbol{Z} \\ \boldsymbol{z}'\boldsymbol{X} & \boldsymbol{z}'\boldsymbol{Y} & \boldsymbol{z}'\boldsymbol{Z} \end{bmatrix} , \tag{8}$$

which is a proper orthonormal matrix, $\boldsymbol{A}\boldsymbol{A}' = \boldsymbol{I}$, $\det(\boldsymbol{A}) = +1$. This is also called the direction cosine matrix, since the rows are the direction cosines of $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ in the CoMRS, and the columns are the direction cosines of $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ in the SRS.

The orthonormality of $\boldsymbol{A}$ implies that the matrix elements satisfy six constraints, leaving three degrees of freedom for the attitude representation. Although one could parametrize each of the nine matrix elements as a continuous function of time, for example using splines, it would in practice not be possible to guarantee that the orthonormality constraints hold at any time. The adopted solution is to represent the instantaneous attitude by a unit quaternion $\mathbf{q}$, which has only four components, $\{q_x, q_y, q_z, q_w\}$, satisfying the single constraint $q_x^2 + q_y^2 + q_z^2 + q_w^2 = 1$. This is easily enforced by normalization.

The attitude quaternion $\mathbf{q}(t)$ gives the rotation from the CoMRS (C) to the SRS (S). Using quaternions, their relation is defined by the transformation of the coordinates of the arbitrary vector $\boldsymbol{v}$ in the two reference systems,

$$\{\mathsf{S}'\boldsymbol{v}, 0\} = \mathbf{q}^{-1}\, \{\mathsf{C}'\boldsymbol{v}, 0\}\, \mathbf{q} \,. \tag{9}$$

In the terminology of Appendix A.3 this is a frame rotation of the vector from C to S. The inverse operation is $\{\mathsf{C}'\boldsymbol{v}, 0\} = \mathbf{q}\, \{\mathsf{S}'\boldsymbol{v}, 0\}\, \mathbf{q}^{-1}$.

Using the B-spline representation in Appendix B, we have

$$\mathbf{q}(t) = \left\langle \textstyle\sum_{n=\ell-M+1}^{\ell} \mathbf{a}_n B_n(t) \right\rangle \,, \tag{10}$$

where $\mathbf{a}_n$ $(n = 0 \ldots N-1)$ are the coefficients of the B-splines $B_n(t)$ of order M defined on the knot sequence $\{\tau_k\}_{k=0}^{N+M-1}$. The function $B_n(t)$ is non-zero only for $\tau_n < t < \tau_{n+M}$, which is why the sum in Eq. (10) only extends over the M terms ending with $n = \ell$. Here, ℓ is the so-called *left index* of t satisfying $\tau_\ell \le t < \tau_{\ell+1}$. The normalization operator $\langle\,\rangle$ guarantees that $\mathbf{q}(t)$ is a unit quaternion for any t in the interval $[\tau_{M-1}, \tau_N]$ over which the spline is completely defined. Although the coefficients $\mathbf{a}_n$ are formally quaternions, they are not of unit length. The attitude parameter vector $\boldsymbol{a}$ consists of the components of the whole set of quaternions $\mathbf{a}_n$.

Cubic splines ($M = 4$) are currently used in this attitude model. Each component of the quaternion (before the normalization in Eq. 10) is then a piecewise cubic polynomial with continuous value, first, and second derivative for any t; the third derivative is discontinuous at the knots. When initializing the solution, it is possible to select any desired order of the spline. Using a higher order, such as $M = 5$ (quartic) or 6 (quintic), provides improved smoothness but also makes the spline fitting less local, and therefore more prone to undesirable oscillatory behaviour. The flexibility of the spline is in principle only governed by the number of degrees of freedom (that is, in practice, the knot frequency), and is therefore independent of the order. One should therefore not choose a higher order than is warranted by the smoothness of the actual effective attitude, which is difficult to model a priori (cf. Appendix D.4). Determining the optimal order and knot frequency may in the end only be possible through analysis of the real mission data.

Equation (7) returns the observed direction to the source relative to the celestial reference system, or $\mathsf{C}'\boldsymbol{u} = [u_X,\ u_Y,\ u_Z]'$. In order to compute the position of the image in the field of view, we need to express this direction in the Scanning Reference System, SRS (Sect. 3.1), or $\mathsf{S}'\boldsymbol{u} = [u_x,\ u_y,\ u_z]'$. The required transformation is obtained by a frame rotation according to Eq. (9),

$$\{\mathsf{S}'\boldsymbol{u},\ 0\} = \mathbf{q}^{-1}\,\{\mathsf{C}'\boldsymbol{u},\ 0\}\,\mathbf{q}\,, \tag{11}$$

whereupon the instrument angles (φ, ζ) are obtained from

$$\mathsf{S}'\boldsymbol{u} \equiv \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} \cos\zeta\cos\varphi \\ \cos\zeta\sin\varphi \\ \sin\zeta \end{bmatrix} \quad\Leftrightarrow\quad \begin{cases} \varphi = \mathrm{atan2}(u_y, u_x) \\ \zeta = \mathrm{atan2}\Big(u_z,\ \sqrt{u_x^2 + u_y^2}\Big) \end{cases} \tag{12}$$

(Fig. 2), where we adopt the convention that $-\pi \le \varphi < \pi$. The field index f and the along-scan field angle η are finally obtained as

$$f = \mathrm{sign}(\varphi)\,, \qquad \eta = \varphi - f\Gamma_\mathrm{c}/2\,, \tag{13}$$

where the basic angle, Γ_c, is here a purely conventional value (as suggested by the subscript). The field-of-view limitations imply that $|\eta| \lesssim 0.5^\circ$ and $|\zeta| \lesssim 0.5^\circ$ for any observation.

Given the time of observation t and the set of source parameters $\boldsymbol{s}$, attitude parameters $\boldsymbol{a}$, and global parameters $\boldsymbol{g}$, the field index f and the field angles (η, ζ) can thus be computed by application of Eqs. (7), (10)–(13). The resulting *computed* field angles are concisely written $\eta(t\,|\,\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{g})$, $\zeta(t\,|\,\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{g})$.

3.4. Geometric instrument model

The geometric instrument model defines the precise layout of the CCDs in terms of the field angles (η, ζ). This layout depends on several different factors, including: the physical geometry of each CCD; its position and alignment in the focal-plane assembly; the optical system including its scale value, distortion and other aberrations; and the adopted (conventional) value of the basic angle Γ_c. Some of these (notably the optical distortion) are different in the two fields of view and may evolve slightly over the mission; on the other hand these variations are expected to be very smooth as a function of the field angles. Other factors, such as the detailed physical geometry of the pixel columns, may be extremely stable but possibly quite irregular on a small scale. As a result, the geometric calibration model must be able to accommodate both smooth and irregular patterns evolving on different time scales in the two fields of view.

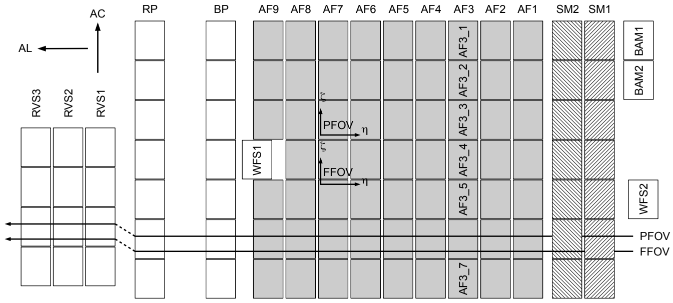


Fig. 3. Layout of the CCDs in Gaia's focal plane. The star images move from right to left in the diagram. The along-scan (AL) and across-scan (AC) directions are indicated in the top left corner. The skymappers (SM1, SM2) provide source image detection, two-dimensional position estimation and field-of-view discrimination. The astrometric field (AF1–AF9) provides accurate AL measurements and (sometimes) AC positions. Additional CCDs used in the blue and red photometers (BP, RP), the radial-velocity spectrometer (RVS), for wavefront sensing (WFS) and basic-angle monitoring (BAM) are not further described in this paper. One of the rows (AF3) illustrates the system for labelling individual CCDs (AF3_1, etc). The nominal paths of two star images, one in the preceding field of view (PFoV) and one in the following field of view (FFoV) are indicated. For purely technical reasons the origin of the field angles (η, ζ) corresponds to different physical locations on the CCDs in the two fields. The physical size of each CCD is 45×59 mm^2.

In the following it should be kept in mind that the astrometric instrument of Gaia is optimised for one-dimensional measurements in the along-scan direction, i.e., of the field angle η, while the requirements in the perpendicular direction (ζ) are much relaxed. This feature of Gaia (and Hipparcos) is a consequence of fundamental considerations derived from the requirements to determine a global reference frame and absolute parallaxes (Lindegren & Bastian 2011). In principle the measurement accuracy in the ζ direction, as well as the corresponding instrument modelling and calibration accuracies, may be relaxed by up to a factor given by the inverse angular size of the field of view (Lindegren 2005), or almost two orders of magnitude for a field of 0.7°. In practice a ratio of the order of 10 may be achieved, in which case the across-scan measurement and calibration errors have a very marginal effect on the overall astrometric accuracy.

The focal plane of Gaia contains a total of 106 CCDs (Laborie et al. 2007), of which only the 14 CCDs in the skymappers (SM1 and SM2) and the 62 in the astrometric field (AF1 through AF9) are used for the astrometric solution (Fig. 3). The CCDs have 4500 pixel lines in the along-scan (AL, constant ζ, decreasing η) direction and 1966 pixel columns in the across-scan (AC, constant η, increasing ζ) direction. They are operated in the Time, Delay and Integrate (TDI) mode (also known as drift-scanning), an imaging technique well-known in astronomy from ground-based programmes such as the Sloan Digital Sky Survey (Gunn et al. 1998). Effectively, the charges are clocked along the CCD columns at the same (average) speed as the motion of the optical images, i.e., 60 arcsec s^{-1} for Gaia. The exposure (integration) time is thus set by the time it takes the image to move across the CCD, or nominally $T \simeq 4.42$ s, if no gate is activated. At this exposure time the central pixels will be saturated for sources brighter than magnitude $G \simeq 12$.[1] Gate activation, or 'gating' for short, is the adopted method to obtain valid measurements of brighter sources. Gating temporarily inhibits charge transfer across a certain TDI line (row of pixels AC), thus effectively zeroing the charge image and reducing the exposure time in proportion to the number of TDI lines following the gate. A range of discrete exposure times is thus available, the shortest one, according to current planning, using only 16 TDI lines (15.7 ms). Gate selection is made by the on-board software based on the measured brightness of the source in relation to the calibrated full-well capacity of the relevant section of the CCD. Measurement errors and the spread in full-well capacities make the gate-activation thresholds somewhat fuzzy, and a given bright source is not necessarily always observed using the same gate. Moreover, a quasi-random selection of the observations of fainter sources will also be gated, viz., if they happen to be read out at about the same time, and on the same CCD, as a gated bright star. The skymappers are operated in a permanently gated mode, so that in practice only the last 2900 TDI lines are used in SM1 and SM2.

The skymappers are crucial for the real-time operation of the instrument, since they detect the sources as they enter the field of view, and allow determination of an approximate two-dimensional position of the images and (together with data from AF1) their speed across the CCDs. This information is used by the on-board attitude computer, in order to determine which CCD pixels should be read out and transmitted to ground (Sect. 3.5). The skymappers also allow to discriminate between the two viewing directions, since sources in the preceding field of view (PFoV) are only seen by SM1 and sources in the following field of view (FFoV) are only seen by SM2. In the astrometric field (as well as in BP, RP and RVS) the two fields of view are superposed.

Most observations acquired in the astrometric field (AF) are purely one-dimensional through the on-chip AC binning of data (Sect. 3.5). However, sources brighter than magnitude $G \simeq 13$ are always observed as two-dimensional images, providing accurate information about the AC pixel coordinate (μ) as well. Some randomly selected fainter images are also observed two-dimensionally ('Calibration Faint Stars', CFS). The bright sources, CFS and SM data provide the AC information necessary for the on-ground three-axis attitude determination.

Because of the TDI mode of observation, AL irregularities of the pixel geometry are smeared out and need not be calibrated, and any measurement of the AL or AC position must effectively be referred back to an 'observation time' half-way through the integration. Correspondingly, the pixel geometry can be represented by a fiducial 'observation line' $[\eta(\mu), \zeta(\mu)]$ traced out in the field angles η, ζ as functions of the AC pixel coordinate μ (Fig. 4). The AC pixel coordinate is defined as a continuous variable with $\mu = 14$ when the image is centred on the first pixel column, $\mu = 15$ at the second pixel column, and so on until $\mu = 1979$ at the last (1966th) pixel column. The offset by 13 pixels allows for the presence of pre-scan pixel data in some observations.

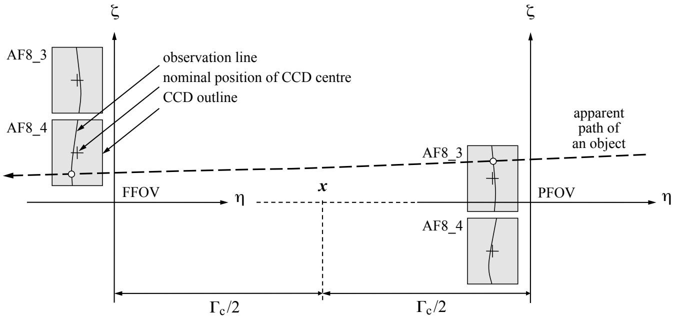


Fig. 4. Schematic illustration of how the field angles (η, ζ) are defined in terms of the CCD layout in Fig. 3. For simplicity only the projections of two CCDs, AF8_3 and AF8_4, into the Scanning Reference System (SRS) are shown (not to scale). The field angles have their origins at the respective viewing direction $\boldsymbol{f}_\mathrm{P}$ or $\boldsymbol{f}_\mathrm{F}$ (Fig. 2), which are defined in relation to the nominal centres of the CCDs (crosses); the actual configuration of the detectors is described by fiducial observation lines according to Eq. (15). The dashed curve shows the apparent path of a stellar image across the two fields of view. Its intersection with the observation lines define the instants of observations.

Nominally, the observation line corresponds to the $(K/2)$th pixel line projected backwards through the optical instrument onto the Scanning Reference System on the sky, where K is the number of active AL pixel lines in the observation (normally $K = 4500$).[2] For a gated observation K is much smaller and the observation line is therefore correspondingly displaced towards the CCD readout register. A separate set of geometrical calibration parameters is therefore needed for each gate being used. In the calibration updating (Sect. 5.3) all the calibration parameters are however solved together, with the overlap due to the fuzzy gate-activation thresholds providing the necessary connection between the different gates.

Let n be an index identifying the different CCDs used for astrometry, i.e., for each of the 76 CCDs in the SM and AF part of the focal plane. Furthermore, let g be a gate index such that $g = 0$ is used for non-gated observations and $g = 1, \ldots, 12$ for gated observations of progressively brighter sources. In each field of view (index f) the nominal observation lines $\left[\eta^0_{fng}(\mu),\, \zeta^0_{fng}(\mu)\right]$ could in principle be calculated from the nominal characteristics of the focal plane assembly (FPA) and ray tracing through the nominal optical system. However, since the nominal observation lines are only used as a reference for the calibration of the actual observation lines, a very simplistic transformation from linear coordinates to angles can be used without introducing approximation errors in the resulting calibration model. The nominal observation lines are therefore defined by the transformation

$$\left.\begin{aligned} \eta^0_{fng}(\mu) \equiv \eta^0_{ng} &= -Y_\mathrm{FPA}[n, g]/F \\ \zeta^0_{fng}(\mu) &= -\Big(X_\mathrm{FPA}[n] - (\mu - \mu_\mathrm{c})p_\mathrm{AC} - X^\mathrm{centre}_\mathrm{FPA}[f]\Big)\Big/F \end{aligned}\right\} \qquad (14)$$

where X_FPA, Y_FPA are physical coordinates (in mm) in the focal plane,[3] $X_\mathrm{FPA}[n]$ is the physical AC coordinate of the nom-

[1] G is the broad-band magnitude measured on Gaia's unfiltered CCDs. $G \simeq V$ for unreddened early-type stars, while $G \simeq V - 2$ for M stars. See Jordi et al. (2010) for a precise characterization of G.

[2] In reality the definition of the fiducial observation line is a bit more complex, as some of the pixel lines are blocked out by an aluminium mask.

[3] For consistency with notations adopted by the ESA project team and the industrial contractor, and extensively used, e.g., for on-ground calibrations, $+X_\mathrm{FPA}$ is oriented along $-\zeta$ and $+Y_\mathrm{FPA}$ along $-\eta$.

inal centre of the nth CCD, $Y_{\rm FPA}[n,g]$ the physical AL coordinate of the nominal observation line for gate g on the nth CCD, and $X_{\rm FPA}^{\rm centre}[f]$ the physical AC coordinate of the nominal field centre for field index f. $\mu_{\rm c} = 996.5$ is the AC pixel coordinate of the CCD centre, $p_{\rm AC} = 30\ \mu$m the physical AC pixel size, and $F = 35$ m the nominal equivalent focal length of the telescope. While η^0_{ng} is independent of f, in the AC direction the origins are offset by about ±221 arcsec between the two fields of view, corresponding to the physical coordinates $X_{\rm FPA}^{\rm centre}[f] = (-37.5\ {\rm mm})f$.

It is emphasized that the nominal observation lines are purely conventional reference quantities, and need not be recomputed, e.g., once a more accurate estimate of F becomes available.

Because of possible changes in the instrument during the mission, the actual observation lines will be functions of time. The time dependence is quantified by introducing independent sets of calibration parameters for successive, non-overlapping time intervals. Different groups of parameters may develop on different time scales, and the resulting formulation can be quite complex. For the sake of illustration, let us distinguish between three categories of geometric calibration parameters:

1. Large-scale AL calibration, $\Delta\eta$. This may (minutely but importantly) change due to thermal variations in the optics, the detectors, and their supporting mechanical structure. These variations could occur on short time scales (of the order of a day), and would in general be different in the two fields of view. They are modelled as low-order polynomials in the across-scan pixel coordinate μ.
2. Small-scale AL calibration, $\delta\eta$. This is mainly related to physical defects or irregularities in the CCDs themselves, for example ‘stitching errors’ introduced by the photolithography process used to manufacture the CCDs. These irregularities are expected to be stable over very long time scales, possibly throughout the mission, and should be identical in both fields of view. They require a spatially detailed modelling, perhaps with a resolution of just one or a few AC pixels.
3. Large-scale AC calibration, $\Delta\zeta$. Although the physical origin is the same as for $\Delta\eta$, the AC components can be assumed to be constant on long time scales, since the calibration requirement in the AC direction is much more relaxed than in the AL direction. They are modelled as low-order polynomials in the field angles, separately in each field of view.

Let index j identify the ‘short’ time intervals needed for the large-scale AL calibration, and index k identify the ‘long’ time intervals applicable to the small-scale AL calibration and large-scale AC calibration. That is, an observation at time t belongs to some short time interval j and some long time interval k, where j and k are readily computed from the known t and the corresponding sequences of division times.[4] Assuming that quadratic polynomials in μ are sufficient for the large-scale calibrations, and that full AC pixel resolution is required for the small-scale AL calibration, the observation lines at time t are modelled as

$$\left.\begin{aligned}\eta_{fng}(\mu,t) &= \eta^0_{ng} + \sum_{r=0}^{2}\Delta\eta_{rfngj}L^*_r\left(\frac{\mu-13.5}{1966}\right) + \delta\eta_{ngkm}\\ \zeta_{fng}(\mu,t) &= \zeta^0_{fng}(\mu) + \sum_{r=0}^{2}\Delta\zeta_{rfngk}L^*_r\left(\frac{\mu-13.5}{1966}\right)\end{aligned}\right\} \quad (15)$$

where $L^*_r(x)$ is the shifted Legendre polynomial of degree r (orthogonal on $[0,1]$), i.e., $L^*_0(x) = 1$, $L^*_1(x) = 2x-1$, $L^*_2(x) = 6x^2-6x+1$, etc; $\Delta\eta_{rfngj}$ are the large-scale AL parameters, $\delta\eta_{ngkm}$ the small-scale AL parameters (with $m = \text{round}(\mu)$ the index of the nearest pixel column), and $\Delta\zeta_{rfngk}$ the large-scale AC parameters.

In order to render all the geometric calibration parameters uniquely determinable, a number of constraints are rigorously enforced by the astrometric solution. Effectively, they define the origins of the field angles, i.e., the viewing directions $\boldsymbol{f}_{\rm P}$ and $\boldsymbol{f}_{\rm F}$, to coincide with the average nominal field angles of the CCD centres. The necessary constraints are:

$$\sum_f\sum_{n\in{\rm AF}}\Delta\eta_{0fn0j} = 0 \quad \text{for each } j, \quad (16)$$

$$\sum_m \delta\eta_{n0km}L^*_r\left(\frac{m-13.5}{1966}\right) = 0 \quad \text{for each combination } rnk, \quad (17)$$

$$\sum_{n\in{\rm AF}}\Delta\zeta_{0fn0k} = 0 \quad \text{for each combination } fk, \quad (18)$$

Note that $g = 0$ throughout in Eq. (16)–(18). That the constraints are only defined in terms of the non-gated observations ($g = 0$) implies that the observation lines for the gated observations must be calibrated relative to the non-gated observations. This is possible since any given bright source will not always be observed with the same gate.

Constraint (16) effectively defines the zero point of the AL field angle η by requiring that $\overline{\Delta\eta} = 0$ when averaging over the CCDs and between the two fields of view. Constraint (17) implies that the small-scale AL corrections $\delta\eta$ do not have any components that could be described by $\Delta\eta$ instead; it therefore ensures a unique division between the large-scale and small-scale components. Constraint (18) effectively defines the zero point of the AC field angle ζ by requiring that $\overline{\Delta\zeta} = 0$, separately in each field of view, when averaged over the CCDs. The sums over n in Eqs. (16) and (18) are restricted to the CCDs in the astrometric field (AF), since the skymapper (SM) measurements are less accurate.

The basic angle $\Gamma_{\rm c}$ introduced in Sect. 3.3 is a fixed reference value, and any real variations of the angle between the two lines of sight will therefore show up as a variation in $\Delta\eta$ with opposite signs in the two fields of view. The offset of the actual basic angle with respect to the conventional value $\Gamma_{\rm c}$ may be defined as the average difference of $\Delta\eta$ between the two fields of view, where the average is computed over the astrometric CCDs, for gate $g = 0$, and over μ; the result for time period j is

$$\Delta\Gamma_j = \frac{1}{62}\sum_{n\in{\rm AF}}\sum_f f\Delta\eta_{0fn0j}\,. \quad (19)$$

Although this is obviously a useful quantity to monitor, it does not appear as a parameter in the geometric calibration model. A number of additional quantities representing the mean scale offset, the mean field orientation, etc., can similarly be computed from the large-scale calibration parameters for the purpose of monitoring.

Equation (15) encapsulates a specific formulation of the geometric instrument model, with certain assumptions about the shape, dependencies, and time scales of possible variations. While this particular model is currently believed to be sufficient to describe the behaviour of the actual instrument to the required accuracy, it is very likely that modifications will be needed after a first analysis of the flight data. Moreover, in the course of

[4] Technically, the use of independent parameter values in successive time intervals represents a spline of order 1 (i.e., degree 0), the separation times constitute the knot sequence, and j or k correspond to the ‘left index’ (Appendix B.2).

the data analysis one may want to try out alternative models, or examine possible systematics resulting from the pre-processing (location estimator). In order to facilitate this, a much more flexible *generic calibration model* has been implemented. In the generic model, the 'observed' field angles (representing the true observation lines) for any observation l are written

$$\left.\begin{aligned} \eta_l^{\rm obs} &= \eta_{ng}^0 + \sum_{r=0}^{N_{\rm AL}-1} E_r^{\rm AL}(l) \\ \zeta_l^{\rm obs} &= \zeta_{fng}^0(\mu) + \sum_{r=0}^{N_{\rm AC}-1} E_r^{\rm AC}(l) \end{aligned}\right\} \tag{20}$$

where each of the $E_r(l)$ (for brevity dropping superscript AL/AC) represents a basic calibration effect, being a linear combination of elemental calibration functions Φ_{rs}:

$$E_r(l) = \sum_{s=0}^{K_r-1} c_{rs}\Phi_{rs}(l)\,. \tag{21}$$

$N_{\rm AL}$ and $N_{\rm AC}$ are the number of effects considered along and across scan. The whole set of coefficients c_{rs} forms the calibration vector $\boldsymbol{c}$.

In the generic formulation, the multiple indices f, n, g and the variables μ and t are replaced by the single observation index l. This allows maximum flexibility in terms of how the calibration model is implemented in the software. The functions Φ_{rs} receive the observation index l, and it is assumed that this index suffices to derive from it all quantities needed to evaluate the calibration functions for this observation. Examples of quantities that can be derived from the observation index are: the FoV index f, the CCD and gate indices n and g, the AC pixel coordinate μ, time t, and any relevant astrometric, photometric or spectroscopic parameter of the source (magnitude, colour index, etc.). Intrinsically real-valued quantities such as t can be subdivided into non-overlapping intervals with different sets of calibration parameters applicable to each interval. The basic calibration model (15) can therefore be implemented as a particular instance of the generic model, with for example $L_r^*(x)$ ($r = 0, 1, 2$) constituting three of the calibration functions (with μ derived from l). Once the calibration functions have been coded, the entire calibration model can be conveniently specified (and changed) through an external configuration file alone using, e.g., XML structures.

Some elemental calibration functions may be introduced for diagnostic purposes rather than actual calibration. An example of this is any function depending on the colour or magnitude of the source. The origin of such effects is briefly explained in Appendix D.1 and D.2. Magnitude- and colour-dependent variations of the instrument response are expected to be fully taken into account by the signal modelling on the CCD data level, as outlined in Sect. 3.5 (notably by the LSF and CDM calibrations), and should not show up in the astrometric solution. Thus, nonzero results for such 'non-geometric' diagnostic calibration parameters indicate that the signal modelling should be improved. In the final solution the diagnostic calibration parameters should ideally be zero.

The parameters of the generic calibration model must satisfy a number of constraints similar to Eqs. (16)–(18). These can be cast in the general form

$$\boldsymbol{C}'\boldsymbol{c} = \boldsymbol{0}\,, \tag{22}$$

where the matrix $\boldsymbol{C}$ contains one column, with known coefficients, for each constraint. The columns are, by design, linearly independent.

3.5. Signal (CCD data) model

As suggested in Fig. 1, the modelling of CCD data at the level of individual pixels (i.e., the photon counts) is not part of the geometrical model of the observations with which we are concerned in this paper. However, the processing of the photon counts, effectively by fitting the CCD data model, produces the 'observations' that are the input to the astrometric core solution. In order to clarify the exact meaning of these observations we include here a brief overview of the signal model.

The pixel size, 10 μm $\simeq$ 59 mas in the along-scan (AL) direction and 30 μm $\simeq$ 177 mas in the across-can (AC) direction, roughly matches the theoretical diffraction image for the 1.45×0.50 m^2 telescope pupil of Gaia (effective wavelength $\sim$ 650 nm). Around each detected object, only a small rectangular window (typically 6–18 pixels long in the AL direction and 12 pixels wide in the AC direction) is actually read out and transmitted to the ground. Moreover, for most of the observations in the astrometric field (AF), on-chip binning in the serial register is used to sum the charges over the window in the AC direction. This effectively results in a one-dimensional image of 6–18 AL 'samples', where the signal (N_k) in each sample k is the sum of 12 AC pixels. The exact time t_k when a sample was transferred to the serial register, expressed on the TCB scale, is in principle known from the time correlation of the on-board clock with ground-based time signals. Because of the known one-to-one relation between the TDI period counter k and t_k, we may use k as a proxy for t_k in subsequent calculations.

For single stars, and in the absence of the effects discussed in Appendix D.2, the sample values in the window are modelled as a stochastic variable (Poisson photon noise plus electronic readout noise) with expected value

$$\lambda_k \equiv {\rm E}\,(N_k) = \beta + \alpha L(k-\kappa) \tag{23}$$

where β, α and κ are the so-called image parameters representing the (uniform) background level, the amplitude (or flux) of the source, and the AL location (pixel coordinate) of the image centroid. The continuous, non-negative function $L(x)$ is the Line Spread Function (LSF) appropriate for the observation. $L(x)$ depends, for example, on the spectrum of the source and on the position of the image in the focal plane. The (in general non-integer) pixel coordinate κ is expressed on the same scale as the (integer) TDI index k, and may be translated to the equivalent TCB $t(\kappa)$ by means of the known relation between k and t_k. The *CCD observation time* is defined as $t(\kappa - K/2)$, where K is the number of TDI periods used for integrating the image (see Appendix D.3). Formally, the CCD observation time is the instant of time at which the centre of the source image passed across the CCD fiducial line halfway between the first and the last TDI line used in the integration (this will depend on the gating).

Fitting the CCD signal model to the one-dimensional sample values N_k thus gives, as the end result of observation l, an estimate of the AL pixel coordinate κ of the image in the pixel stream, which is then transformed to the observation time t_l. The fitting procedure also provides an estimate of the uncertainty in κ, which can be expressed in angular measure as a formal standard deviation of the AL measurement, $\sigma_l^{\rm AL}$. It is derived by error propagation through the fitted signal model, taking into account the dominant noise sources, photon noise and readout noise.

For some observations, AC information is also provided through two-dimensional sampling of the pixel window around the detected object. This applies to all SM observations, AF observations of bright ($G \lesssim 13$) sources, and AF observations of Calibration Faint Stars (Sect. 3.4). The modelling of the two-dimensional images follows the same principles as outlined above for the one-dimensional (AL only) case, only that the LSF is replaced by a two-dimensional Point Spread Function (PSF) and that there is one more location parameter to estimate, namely the AC pixel coordinate μ. The astrometric result in this case consists of the observation time t_l, the observed AC coordinate μ_l, and their formal standard uncertainties $\sigma_l^{\rm AL}$ and $\sigma_l^{\rm AC}$ (both expressed as angles).

The estimation errors for different images are, for the subsequent analysis, assumed to be statistically independent (and therefore uncorrelated). This is a very good approximation to the extent that they only depend on the photon and readout noises. However, modelling errors at the various stages of the processing (in particular CTI effects in the signal modelling [Appendix D.3] and attitude irregularities [Appendix D.4]) are likely to introduce errors that are correlated at least over the nine AF observations in a field transit. The resulting correlations as such are not taken into account in the astrometric solution (i.e., the weight matrix of the least-squares equations is taken to be diagonal), but the sizes of the modelling errors are estimated, and are employed to reduce the statistical weights of the observations as described in Sect. 3.6. The AL and AC estimates of a given (two-dimensional) image are roughly independent at least in the limit of small optical aberrations.

3.6. Synthesis model

By synthesis of the models described in the preceding sections, we are now in a position to formulate very precisely the core astrometric data analysis problem as outlined in Sect. 2. The unknowns are represented by the vectors $\boldsymbol{s}$, $\boldsymbol{a}$, $\boldsymbol{c}$, and $\boldsymbol{g}$ of respectively the source, attitude, calibration, and global parameters. For any observation l the observed quantities are the observation time t_l and, where applicable, the observed AC pixel coordinate μ_l, with their formal uncertainties $\sigma_l^{\rm AL}$, $\sigma_l^{\rm AC}$. We then have the global minimization problem

$$\min_{\boldsymbol{s},\boldsymbol{a},\boldsymbol{c},\boldsymbol{g}} Q = \sum_{l\in{\rm AL}} \frac{(R_l^{\rm AL})^2\, w_l^{\rm AL}}{(\sigma_l^{\rm AL})^2 + (\epsilon_l^{\rm AL})^2} + \sum_{l\in{\rm AC}} \frac{(R_l^{\rm AC})^2\, w_l^{\rm AC}}{(\sigma_l^{\rm AC})^2 + (\epsilon_l^{\rm AC})^2}\,, \quad (24)$$

where

$$R_l^{\rm AL}(\boldsymbol{s},\boldsymbol{a},\boldsymbol{c},\boldsymbol{g}) = \eta_{fng}(\mu_l, t_l\,|\,\boldsymbol{c}) - \eta(t_l\,|\,\boldsymbol{s},\boldsymbol{a},\boldsymbol{g})\,, \quad (25)$$

$$R_l^{\rm AC}(\boldsymbol{s},\boldsymbol{a},\boldsymbol{c},\boldsymbol{g}) = \zeta_{fng}(\mu_l, t_l\,|\,\boldsymbol{c}) - \zeta(t_l\,|\,\boldsymbol{s},\boldsymbol{a},\boldsymbol{g}) \quad (26)$$

are the residuals in the field angles, taken as functions of the unknowns[5], and $l \in$ AL refers to observations with a valid AL component, etc. The applicable indices f, n, g are of course known for every observation l. In Eq. (24), $\epsilon_l^{\rm AL}$ and $\epsilon_l^{\rm AC}$ represent all AL and AC error sources extraneous to the formal uncertainties; they include in particular modelling errors in the source behaviour (e.g., for unrecognized binaries), attitude and calibration, which have to be estimated as functions of time and source in the course of the data analysis process. $w_l^{\rm AL}$ and $w_l^{\rm AC}$ are weight factors, always in the range 0 to 1; for most observations they are equal to 1, but 'bad' data (outliers) are assigned smaller weight factors. The determination of these factors is discussed in Sect. 5.1.2.

For the sake of conciseness we shall hereafter consider the AL and AC components of an observation to have separate observation indices l, so that for example R_l stands for either $R_l^{\rm AL}$ or $R_l^{\rm AC}$, as the case may be. This allows the two sums in Eq. (24) to be contracted and written concisely as

$$Q(\boldsymbol{s},\boldsymbol{a},\boldsymbol{c},\boldsymbol{g}) = \sum_l \frac{R_l^2\, w_l}{\sigma_l^2 + \epsilon_l^2} = \sum_l R_l^2\, W_l\,, \quad (27)$$

where $W_l = w_l/(\sigma_l^2 + \epsilon_l^2)$ is the statistical weight of the observation.

The excess noise ϵ_l represents modelling errors and should ideally be zero. However, it is unavoidable that some sources do not behave exactly according to the adopted astrometric model (Sect. 3.2), or that the attitude sometimes cannot be represented by the spline model in Sect. 3.3 to sufficient accuracy. The excess noise term in Eq. (27) is introduced to allow these cases to be handled in a reasonable way, i.e., by effectively reducing the statistical weight of the observations affected. It should be noted that these modelling errors are assumed to affect *all* the observations of a particular star, or *all* the observations in a given time interval. (By contrast, the downweighting factor w_l is intended to take care of isolated outliers, for example due to a cosmic-ray hit in one of the CCD samples.) This is reflected in the way the excess noise is modelled as the sum of two components,

$$\epsilon_l^2 = \epsilon_i^2 + \epsilon_a^2(t_l)\,, \quad (28)$$

where ϵ_i is the excess noise associated with source i (if $l \in i$, that is, l is an observation of source i), and $\epsilon_a(t)$ is the excess attitude noise, being a function of time. For a 'good' primary source, we should have $\epsilon_i = 0$, and for a 'good' stretch of attitude data we may have $\epsilon_a(t) = 0$. Calibration modelling errors are not explicitly introduced in Eq. (28), but could be regarded as a more or less constant part of the excess attitude noise. The estimation of ϵ_i is described in Sect. 5.1.2, and the estimation of $\epsilon_a(t)$ in Sect. 5.2.5.

Three separate functions are needed to describe the excess attitude noise, corresponding to AL observations, AC observations in the preceding field of view (ACP), and AC observations in the following field of view (ACF). We distinguish between these functions by letting the subscript a in $\epsilon_a(t)$ stand for either AL, ACP or ACF.

4. Solving the global minimization problem

Assuming 10^8 primary sources, the number of unknowns in the global minimization problem, Eq. (24), is about 5×10^8 for the sources ($\boldsymbol{s}$), $4{\times}10^7$ for the attitude ($\boldsymbol{a}$, assuming a knot interval of 15 s for the 5 yr mission; cf. Sect. 5.2.1), 10^6 for the calibration $\boldsymbol{c}$, and less than 100 global parameters ($\boldsymbol{g}$). The number of elementary observations (l) considered is about 8×10^{10}. However, the size of the data set, and the large number of parameters, would not by themselves be a problem if the observations, or sources, could be processed sequentially. The difficulty is caused by the strong connectivity among the observations: each source is effectively observed relative to a large number of other sources simultaneously in the field of view, or in the complementary field of view some 106.5° away on the sky, linked together by the attitude and calibration models. The complexity of the astrometric solution in terms of the connectivity between the sources provided by the attitude modelling was analysed by Bombrun et al. (2010), who concluded that a direct solution is infeasible, by

[5] Note that in Eq. (25) the quantity μ_l is just a given value (observed or computed); in the case of one-dimensional images the observed μ_l is replaced by an approximate value derived from current knowledge on the source and attitude.

many orders of magnitude, with today's computational capabilities. The study neglected the additional connectivity due to the calibration model, which makes the problem even more unrealistic to attack by a direct method. Note that this impossibility is not a defect, but a virtue of the mathematical system under consideration: it guarantees that a unique, coherent and completely independent global solution for the whole sky can be derived from the system.

The natural alternative to a direct solution is to use some iterative method. This is in fact the standard way to deal with large, sparse systems of equations. The literature in the field is vast and a plethora of methods exist for various kinds of applications. However, the iterative method adopted for Gaia did not spring from a box of ready-made algorithms. Rather, it was developed over several years in parallel with the software system in which it could be implemented and tested. Originally based on an intuitive and somewhat simplistic approach, the algorithm has developed through a series of experiments, insights and improvements into a rigorous, efficient and well-understood procedure, completely adapted to its particular application. In this section we first describe the approach in broad outline, then provide the mathematical background for its better understanding and further development.

4.1. Outline of the iterative solution

The numerical approach to the Gaia astrometry is a block-iterative least-squares solution, referred to as the Astrometric Global Iterative Solution (AGIS). In its simplest form, four blocks are evaluated in a cyclic sequence until convergence. The blocks map to the four different kinds of unknowns outlined in Sect. 3, namely:

S: the source (star) update, in which the astrometric parameters $\boldsymbol{s}$ of the primary sources are improved;
A: the attitude update, in which the attitude parameters $\boldsymbol{a}$ are improved;
C: the calibration update, in which the calibration parameters $\boldsymbol{c}$ are improved;
G: the global update, in which the global parameters $\boldsymbol{g}$ are improved.

The G block is optional, and will perhaps only be used in some of the final solutions, since the global parameters can normally be assumed to be known a priori to high accuracy.

The blocks must be iterated because each one of them needs data from the three other processes. For example, when computing the astrometric parameters in the S block, the attitude, calibration and global parameters are taken from the previous iteration. The resulting (updated) astrometric parameters are used the next time the A block is run, and so on. This iterative approach to the astrometric solution was proposed early on in the Hipparcos project as an alternative to the 'three-step method' subsequently adopted for the original Hipparcos reductions; see ESA (1997, Vol. 3, p. 488) and references therein. Indeed, the later re-reduction of the Hipparcos raw data by van Leeuwen (2007) used a very similar iterative method, and yielded significantly improved results mainly by virtue of the much more elaborate attitude modelling implemented as part of the approach.

While the block-iterative solution as outlined above is intuitive and appealing in its simplicity, it is from a mathematical standpoint not obvious that it must converge; and if it does indeed converge, it is not obvious how many iterations are required, whether the order of the blocks in each iteration matter, and whether the converged results do in fact constitute a solution to the global minimization problem. These are fundamental questions for the validity of the adopted iterative approach, and we therefore take some care in the following subsections to establish its theoretical foundations (Sects. 4.2–4.5). Section 5 then describes each of the S, A, C and G blocks in some detail. In addition to these blocks, separate processes are required for the alignment of the astrometric solution with the ICRS, the selection of primary sources, and the calculation of standard uncertainties; these auxiliary processes are discussed in Sect. 6.

4.2. Least-squares approach

Strictly speaking, Eq. (24) is not a least-squares problem, because of the weight factors $w_l^{\rm AL}$, $w_l^{\rm AC}$ (as well as the excess noises $\epsilon_l^{\rm AL}$, $\epsilon_l^{\rm AC}$), which depend on the AL and AC residuals and hence on the unknowns $(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{g})$. In Eq. (1) this dependence is formally included in the unspecified metric $\mathcal{M}$, which therefore is not simply a (weighted) Euclidean norm.

In principle, the minimization problem (24) can be solved by finding a point where the partial derivatives of the objective function Q with respect to all the unknowns are simultaneously zero. In practice, however, the partial derivatives are not computed completely rigorously, and the problem solved is therefore a slightly different one from what is outlined above. In order to understand precisely the approximations involved, it is necessary to consider how different kinds of non-linearities enter the problem.

The functions $\eta_{fng}(\mu, t\,|\,\boldsymbol{c})$ and $\zeta_{fng}(\mu, t\,|\,\boldsymbol{c})$ appearing in Eqs. (25)–(26) are strictly linear in the calibration parameters $\boldsymbol{c}$, by virtue of the basic geometric calibration model in Eq. (15) or the generic model in Eqs. (20)–(21). On the other hand, the functions $\eta(t\,|\,\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{g})$ and $\zeta(t\,|\,\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{g})$ are non-linear in $\boldsymbol{s}$, $\boldsymbol{a}$, and $\boldsymbol{g}$ due to the complex transformations involved (Sects. 3.2–3.3). However, thanks to the data processing prior to the astrometric solution, the initial errors in these parameters are already so small that the corresponding errors in η and ζ are only some 0.1 arcsec ($\sim 10^{-6}$ rad). Second-order terms are therefore typically less than 10^{-12} rad $\simeq 0.2\,\mu$as, that is negligible in comparison with the noise of a single AL observation (some 100 μas). This means that the partial derivatives of the residuals $R_l^{\rm AL}$ and $R_l^{\rm AC}$ with respect to all the unknowns do not change in the course of the solution. (In practice they are in fact recomputed in each iteration, although that is mainly done because it is more convenient than to store and retrieve the values; nevertheless, this takes care of any remaining non-linearity, however small.) The non-linearities of the underlying astrometric, attitude, calibration and global models are therefore not an issue for the minimization problem as such.

The weight factors w_l represent a different kind of non-linearity, potentially much more important for the final solution. These factors are introduced to make the solution robust against outliers, by reducing their influence on Q and hence on the estimated parameter values (Sect. 5.1.2). Ideally, outlying observations should not contribute at all to the solution (by having $w_l = 0$), while 'normal' observations should receive full weight ($w_l = 1$). In reality there will however be a transition region where the weight factors are between 0 and 1. Since the weight factors are in practice determined by the normalized residuals, $\hat{R}_l \equiv R_l(\sigma_l^2 + \epsilon_l^2)^{-1/2}$, which in turn depend on the parameter values $(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{g})$, it follows that the partial derivatives contain extra terms of the form $\hat{R}_l^2 \partial w_l/\partial\boldsymbol{s}$, $\hat{R}_l^2 \partial w_l/\partial\boldsymbol{a}$, etc., that are non-zero for some observations. Analogous considerations apply to the excess noise terms ϵ_l: they too are estimated by means of the residuals (Sects. 5.1.2 and 5.2.5), and therefore in principle introduce additional terms in the partial derivatives.

We take the somewhat pragmatic approach of neglecting the terms depending on the partial derivatives of w_l and ϵ_l with respect to the unknowns when seeking the solution to the global minimization problem. The consequences of this approximation can be appreciated by observing that the down-weighting ($w_l < 1$) only kicks in when the absolute value of the residual exceeds a few times the total standard uncertainty, or some 0.5–1 mas for typical observations of bright sources. Similarly the estimated ϵ_l are only sensitive to changes of the residuals of a similar size. Experience with AGIS runs on simulated data show that the typical changes of the residuals fall below this level already in the first few iterations. Thereafter the weight factors and the estimated excess noises do not change significantly. Neglecting the derivatives of w_l and ϵ_l in the global minimization problem is therefore equivalent to solving the weighted least-squares problem with w_l and ϵ_l fixed at whatever values they settle to after the initial iterations. This is a reasonable assumption given that the statistical weight of any observation, and the size of the modelling errors, are not a priori expected to depend on the actual values of the parameters. The precise solution does of course depend on how w_l and ϵ_l are estimated, but that is an unavoidable consequence of any practical data analysis approach.

4.3. Normal equations

The minimization of Q in Eq. (27) is thus solved by the weighted least-squares method, assuming fixed weights W_l that are however determined as part of the solution. The normal equations for the sources are given by $\frac{1}{2}\partial Q/\partial \boldsymbol{s} = \boldsymbol{0}$, and similarly for the other unknowns. Linearising around any reference point $(\boldsymbol{s}^{\rm ref}, \boldsymbol{a}^{\rm ref}, \boldsymbol{c}^{\rm ref}, \boldsymbol{g}^{\rm ref})$ within the linear regime of parameter space, i.e., setting $\boldsymbol{s} = \boldsymbol{s}^{\rm ref} + \boldsymbol{x}_s$, and similarly for the other unknowns, and expanding to first order in the differential vector $\boldsymbol{x} = (\boldsymbol{x}_s', \boldsymbol{x}_a', \boldsymbol{x}_c', \boldsymbol{x}_g')'$, we find the normal equations as

$$\left[\sum_l \frac{\partial R_l}{\partial \boldsymbol{x}} \frac{\partial R_l}{\partial \boldsymbol{x}'} W_l\right] \boldsymbol{x} = -\sum_l \frac{\partial R_l}{\partial \boldsymbol{x}} R_l(\boldsymbol{s}^{\rm ref}, \boldsymbol{a}^{\rm ref}, \boldsymbol{c}^{\rm ref}, \boldsymbol{g}^{\rm ref}) W_l \,. \quad (29)$$

This can be written in matrix form as

$$\boldsymbol{N}\boldsymbol{x} = \boldsymbol{b}\,, \quad (30)$$

where $\boldsymbol{N}$ is a symmetric matrix. We now proceed to analyse the structure of this linear system of equations in terms of the previously mentioned block updates S, A, C, G.

The matrix $\boldsymbol{N}$ and the vectors (column matrices) $\boldsymbol{x}$, $\boldsymbol{b}$ can be partitioned into sub-matrices and sub-vectors corresponding to the different parameter vectors $\boldsymbol{s}$, $\boldsymbol{a}$, $\boldsymbol{c}$, and $\boldsymbol{g}$:

$$\begin{bmatrix} \boldsymbol{N}_{ss} & \boldsymbol{N}_{sa} & \boldsymbol{N}_{sc} & \boldsymbol{N}_{sg} \\ \boldsymbol{N}_{as} & \boldsymbol{N}_{aa} & \boldsymbol{N}_{ac} & \boldsymbol{N}_{ag} \\ \boldsymbol{N}_{cs} & \boldsymbol{N}_{ca} & \boldsymbol{N}_{cc} & \boldsymbol{N}_{cg} \\ \boldsymbol{N}_{gs} & \boldsymbol{N}_{ga} & \boldsymbol{N}_{gc} & \boldsymbol{N}_{gg} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_s \\ \boldsymbol{x}_a \\ \boldsymbol{x}_c \\ \boldsymbol{x}_g \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_s \\ \boldsymbol{b}_a \\ \boldsymbol{b}_c \\ \boldsymbol{b}_g \end{bmatrix}, \quad (31)$$

where $\boldsymbol{N}_{as} = \boldsymbol{N}_{sa}'$, etc. Of importance here is that $\boldsymbol{N}_{ss}$ and $\boldsymbol{N}_{aa}$ have a particularly simple structure. Since $\boldsymbol{s}$ is sub-divided into vectors $\boldsymbol{s}_i$ of length 5 for the individual primary sources (i), it is natural to sub-divide $\boldsymbol{N}_{ss}$ into blocks of 5×5 elements. From Eq. (29) it follows that the (i, j)th such block is given by

$$[\boldsymbol{N}_{ss}]_{ij} = \sum_l \frac{\partial R_l}{\partial \boldsymbol{s}_i} \frac{\partial R_l}{\partial \boldsymbol{s}_j'} W_l = \begin{cases} \displaystyle\sum_{l\in i} \frac{\partial R_l}{\partial \boldsymbol{s}_i} \frac{\partial R_l}{\partial \boldsymbol{s}_i'} W_l & \text{if } i = j, \\ \boldsymbol{0} & \text{if } i \neq j, \end{cases} \quad (32)$$

where $l \in i$ signifies that the sum is taken over the observations of source i. The result for $i \neq j$ follows because no observation l is associated with more than one primary source. $\boldsymbol{N}_{ss}$ is consequently block-diagonal, and $\boldsymbol{N}_{ss}^{-1}\boldsymbol{b}_s$ can trivially be computed for arbitrary vector $\boldsymbol{b}_s$ by looping through the sources and solving the corresponding 5×5 system for each source.[6] This is exactly what is done in the source update block (S).

The vector of attitude unknowns is naturally sub-divided into vectors $\boldsymbol{a}_n$ of length 4, containing the elements of the quaternions $\mathbf{a}_n$ that serve as coefficients in the B-spline representation, Eq. (10). If $\boldsymbol{N}_{aa}$ is correspondingly sub-divided into blocks of 4×4 elements, it follows from Eq. (29) that the (n, m)th such block is given by

$$[\boldsymbol{N}_{aa}]_{nm} = \sum_l \frac{\partial R_l}{\partial \boldsymbol{a}_n} \frac{\partial R_l}{\partial \boldsymbol{a}_m'} W_l \,. \quad (33)$$

With ℓ denoting the left index of t_l (Sect. 3.3), we have $\partial R_l/\partial \boldsymbol{a}_n = \boldsymbol{0}$ whenever $n < \ell - M + 1$ or $n > \ell$, where M is the order of the spline. It follows that $[\boldsymbol{N}_{aa}]_{nm} = \boldsymbol{0}$ if $|n - m| > M - 1$ (cf. Appendix B.1). The non-zero blocks in $\boldsymbol{N}_{aa}$ are therefore confined to the diagonal and $M - 1$ blocks above and below the diagonal (Fig. 5). Thus, $\boldsymbol{N}_{aa}^{-1}\boldsymbol{b}_a$ can be efficiently computed for arbitrary $\boldsymbol{b}_a$ since the Cholesky decomposition of the matrix does not produce any additional fill-in (Appendix C). This system is solved in the attitude update block (A).

In the geometric instrument model (Sect. 3.4), each CCD/gate and time interval combination (index ngk for example in Eq. 15) has its own set of unknowns. Moreover, a given observation l can only refer to one CCD/gate combination. By the same reasoning as above, the sub-matrix $\boldsymbol{N}_{cc}$ is therefore block-diagonal, and $\boldsymbol{N}_{cc}^{-1}\boldsymbol{b}_c$ can be computed for arbitrary $\boldsymbol{b}_c$ by looping over the CCD/gates combinations. This is done in the calibration update block (C). Although the number of calibration parameters per CCD/gate combination can be fairly large ($\sim 10^4$), the resulting systems are well within the bounds that can readily be handled by direct matrix methods, even without taking into account their sparseness.

By definition all the global parameters affect each observation, and the sub-matrix $\boldsymbol{N}_{gg}$ is therefore full. However, since the number of global parameters is never large, $\boldsymbol{N}_{gg}^{-1}\boldsymbol{b}_g$ can easily be computed, which is done in the global update block (G).

In the description above we have implicitly assumed that each of the diagonal blocks $\boldsymbol{N}_{ss}$, $\boldsymbol{N}_{aa}$, $\boldsymbol{N}_{cc}$, and $\boldsymbol{N}_{gg}$ is non-singular, and even well-conditioned in order to avoid numerical instability. This is equivalent to the statement that each of the blocks S, A, C and G is a well-posed problem: for example, that the determination of the source parameters is ‘easy’ if we assume that we know the attitude, calibration and global parameters. This will in practice be guaranteed by the choice of primary sources (which will make $[\boldsymbol{N}_{ss}]_{ii}$ well-conditioned for every i) for the S block, and by the adopted attitude, calibration and global parametrizations, including the constraints necessary to render the updates unique – in particular the quaternion length normalization in Eq. (10) for the attitude model, and Eq. (22) for the calibration model.

Turning now to the off-diagonal sub-matrices of $\boldsymbol{N}$, it is natural to sub-divide for example the sub-matrix $\boldsymbol{N}_{as}$ into blocks of 4×5 elements corresponding to the 4 components of the quater-

[6] Here, and elsewhere in this paper, an expression like $\boldsymbol{A}^{-1}\boldsymbol{b}$ is shorthand notation for solving the system $\boldsymbol{A}\boldsymbol{y} = \boldsymbol{b}$. The inverse matrix $\boldsymbol{A}^{-1}$ is (usually) not computed, but only the solution vector $\boldsymbol{y} = \boldsymbol{A}^{-1}\boldsymbol{b}$ itself.

nion and the 5 astrometric parameters. The (n, i)th block,

$$[\boldsymbol{N}_{as}]_{ni} = \sum_l \frac{\partial R_l}{\partial \boldsymbol{a}_n} \frac{\partial R_l}{\partial \boldsymbol{s}_i'} W_l \,, \tag{34}$$

is non-zero only if source i was observed in the time interval $[\tau_n, \tau_{n+M}]$, which is the support of the B-spline $B_n(t)$. $\boldsymbol{N}_{as}$ is therefore very sparse, but it also has no simple structure because the distribution of the non-zero blocks is linked to the scanning law. Fortunately, as will be explained in Sect. 4.5, there is no need to explicitly compute, much less store, this sub-matrix, nor any of the other off-diagonal sub-matrices in Eq. (31).

4.4. Rank of the normal equations

From the nature of the astrometric observations, which are in effect differential within the (combined) field of view, and the modelling of the primary sources, which does not assume that any of their positions or proper motions are known a priori, it is clear that there is no unique astrometric solution to the problem as outlined above. The fundamental reason for this is that any (small) change in the orientation of the celestial reference system, as well as the introduction of a (small) inertial spin of the system, would leave all observations invariant. In principle the non-uniqueness of the solution is not a problem as such, since the resulting system of positions and proper motions are afterwards aligned with the ICRS by a special process (Sect. 6.1). However, it does imply that the normal matrix $\boldsymbol{N}$ is in principle singular,[7] which may have consequences for the numerical solution of the normal equations. We say singular 'in principle' because arithmetic rounding errors will in practice prevent it from becoming truly singular, although it remains extremely ill-conditioned.

More precisely, we expect $\boldsymbol{N}$ to have rank $n-6$, if $n = \dim(\boldsymbol{N})$ is the total number of unknowns. The null space of the matrix,

$$\mathcal{N}(\boldsymbol{N}) = \{\boldsymbol{v} \in \mathbb{R}^n \,|\, \boldsymbol{N}\boldsymbol{v} = \boldsymbol{0}\} \tag{35}$$

therefore has rank six. Indeed, it is easy to find six linearly independent vectors $\boldsymbol{v}_0, \ldots, \boldsymbol{v}_5$ that span the null space: the first three are found by introducing small changes in the orientation of the celestial reference system around each of its principal axes, and deriving the corresponding changes in $\boldsymbol{s}$ and $\boldsymbol{a}$ ($\boldsymbol{c}$ and $\boldsymbol{g}$ being independent of the reference system); the last three are correspondingly found by introducing a small inertial spin of the reference system around each axis (see Sect. 6.1.5 for details). Introducing the $n \times 6$ matrix $\boldsymbol{V} = (\boldsymbol{v}_0, \ldots, \boldsymbol{v}_5)$ we have

$$\boldsymbol{N}\boldsymbol{V} = \boldsymbol{0} \,. \tag{36}$$

The singularity can in principle be removed by adding the six constraints $\boldsymbol{V}'\boldsymbol{x} = \boldsymbol{0}$, but in practice that is not necessary. It suffices to derive *one* particular solution $\tilde{\boldsymbol{x}}$ to the normal equations, then the whole solution space can be written $\tilde{\boldsymbol{x}} + \boldsymbol{V}\boldsymbol{z}$ for arbitrary $\boldsymbol{z} \in \mathbb{R}^6$. The vector $\boldsymbol{z}$ is effectively determined by the frame rotator (Sect. 6.1), yielding the unique solution that best agrees with the adopted definition of the ICRS.

Quite apart from numerical rounding errors, it is not completely true that $\boldsymbol{N}$ is mathematically singular. Stellar aberration and parallax introduce some absolute knowledge about the reference system via the barycentric ephemeris of Gaia, which is expressed in ICRS and is not part of the adjustment process. However, since stellar aberration is at most 10^{-4} rad, the orientation error would have to be of the order of 1 mas for the aberration effect to change by 0.1 μas (say). So, although absent in principle, the indeterminacy of the reference frame orientation and spin exists in practice. Since the orientation can be determined to much higher accuracy than 1 mas (by means of the optical counterparts of radio sources), the contribution of stellar aberration to the absolute frame knowledge can be neglected in practice. The same holds, a fortiori, for the much smaller parallax effect.

4.5. The simple iteration step

Consider the system

$$\boldsymbol{K}\boldsymbol{d} = \boldsymbol{b} \,, \qquad \boldsymbol{K} = \begin{bmatrix} \boldsymbol{N}_{ss} & \varnothing & \varnothing & \varnothing \\ \boldsymbol{N}_{as} & \boldsymbol{N}_{aa} & \varnothing & \varnothing \\ \boldsymbol{N}_{cs} & \boldsymbol{N}_{ca} & \boldsymbol{N}_{cc} & \varnothing \\ \boldsymbol{N}_{gs} & \boldsymbol{N}_{ga} & \boldsymbol{N}_{gc} & \boldsymbol{N}_{gg} \end{bmatrix} . \tag{37}$$

where $\boldsymbol{b}$ is the same right-hand side as in Eq. (30), and each $\varnothing$ stands for a zero-filled sub-matrix of the appropriate dimensions. Although this system is of the same size as Eq. (30), is it fundamentally different in that it can be directly solved through a sequence of smaller systems,

$$\left.\begin{aligned} \boldsymbol{N}_{ss}\boldsymbol{d}_s &= \boldsymbol{b}_s \\ \boldsymbol{N}_{aa}\boldsymbol{d}_a &= \boldsymbol{b}_a - \boldsymbol{N}_{as}\boldsymbol{d}_s \\ \boldsymbol{N}_{cc}\boldsymbol{d}_c &= \boldsymbol{b}_c - \boldsymbol{N}_{cs}\boldsymbol{d}_s - \boldsymbol{N}_{ca}\boldsymbol{d}_a \\ \boldsymbol{N}_{gg}\boldsymbol{d}_g &= \boldsymbol{b}_g - \boldsymbol{N}_{gs}\boldsymbol{d}_s - \boldsymbol{N}_{ga}\boldsymbol{d}_a - \boldsymbol{N}_{gc}\boldsymbol{d}_c \end{aligned}\right\} \tag{38}$$

where each sub-system is of the form S, A, C, G described above, allowing it to be solved with relative ease. (Naturally, the resulting solution $\boldsymbol{d}$ is also quite different from the $\boldsymbol{x}$ in Eq. 30!) By means of Eqs. (29) and (34) the right-hand side in the second sub-system becomes

$$\begin{aligned} \boldsymbol{b}_a - \boldsymbol{N}_{as}\boldsymbol{d}_s &= -\sum_l \frac{\partial R_l}{\partial \boldsymbol{a}} \left[R_l(\boldsymbol{s}^{\text{ref}}, \boldsymbol{a}^{\text{ref}}, \boldsymbol{c}^{\text{ref}}, \boldsymbol{g}^{\text{ref}}) + \frac{\partial R_l}{\partial \boldsymbol{s}'} \boldsymbol{d}_s \right] W_l \\ &= -\sum_l \frac{\partial R_l}{\partial \boldsymbol{a}} R_l(\boldsymbol{s}^{\text{ref}} + \boldsymbol{d}_s, \boldsymbol{a}^{\text{ref}}, \boldsymbol{c}^{\text{ref}}, \boldsymbol{g}^{\text{ref}}) W_l \,, \end{aligned} \tag{39}$$

where the linearity of $R_l(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{g})$ has been used in a Taylor expansion around the reference values. This shows that the off-diagonal matrix $\boldsymbol{N}_{as}$ is not needed in order to compute the right-hand side of the second sub-system in Eq. (38), if only the residuals are computed *after* the source parameters have been updated by the solution of the first sub-system. Similarly, we find that the right-hand side in the third sub-system can be obtained from the residuals after updating both the source and attitude parameters, and so on. The off-diagonal sub-matrices in $\boldsymbol{K}$ are therefore not needed, provided that the sub-vectors of unknowns are successively updated before the new right-hand sides are computed.

From the above it is clear that a single AGIS iteration, consisting of the successive application of the four update blocks S, A, C and G, is mathematically equivalent to an updating of the unknowns by $\boldsymbol{d} = \boldsymbol{K}^{-1}\boldsymbol{b}$. In the context of iterative solution algorithms, the matrix $\boldsymbol{K}$ is referred to as the preconditioner of the normal equations system (Axelsson 1996).

As previously noted, we assume that the block-diagonal matrices $\boldsymbol{N}_{ss}$, $\boldsymbol{N}_{aa}$, $\boldsymbol{N}_{cc}$, $\boldsymbol{N}_{gg}$ are all non-singular, and in fact positive definite, for a proper formulation of the S, A, C, and G blocks. This ensures that the preconditioner $\boldsymbol{K}$ is non-singular, even though $\boldsymbol{N}$ is not.

In the course of the iterations, new right-hand sides of the normal equations will be computed, while the matrix remains

[7] More precisely, $\boldsymbol{N}$ is properly semidefinite, so that $\boldsymbol{x}'\boldsymbol{N}\boldsymbol{x} \geq 0$ for all $\boldsymbol{x} \neq \boldsymbol{0}$, with equality for some $\boldsymbol{x} \neq \boldsymbol{0}$.

essentially unchanged. From now on, let us express the residuals R_l not as functions of the parameter values $\boldsymbol{s}$, $\boldsymbol{a}$, $\boldsymbol{c}$, $\boldsymbol{g}$ but in terms of the differential parameter vector $\boldsymbol{x}$ relative to the reference values. The original right-hand side, denoted $\boldsymbol{b}^{(0)}$, corresponds to the initial differential parameter vector $\boldsymbol{x}^{(0)} = \boldsymbol{0}$. If $\boldsymbol{x}$ (without a superscript) denotes the exact solution of Eq. (30), the initial error vector is $\boldsymbol{e}^{(0)} = \boldsymbol{x}^{(0)} - \boldsymbol{x} = -\boldsymbol{x}$. Although this vector is of course not known, we do know $-\boldsymbol{N}\boldsymbol{e}^{(0)} = \boldsymbol{N}\boldsymbol{x} = \boldsymbol{b}^{(0)}$. Solving the preconditioner system (37) gives $\boldsymbol{d}^{(0)} = \boldsymbol{K}^{-1}\boldsymbol{b}^{(0)}$ and the updated parameter vector $\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \boldsymbol{d}^{(0)}$. The new error vector $\boldsymbol{e}^{(1)} = \boldsymbol{x}^{(1)} - \boldsymbol{x}$ is again not known, but $-\boldsymbol{N}\boldsymbol{e}^{(1)} = \boldsymbol{b}^{(0)} - \boldsymbol{N}\boldsymbol{d}^{(0)} = \boldsymbol{b}^{(1)}$ is obtained by inserting the updated parameters in the right-hand side of Eq. (29), by the same argument as in Eq. (39). Generalizing, we have the so-called *simple iteration scheme*

$$\left.\begin{aligned} \boldsymbol{b}^{(k)} &= -\sum_l \frac{\partial R_l}{\partial \boldsymbol{x}} R_l(\boldsymbol{x}^{(k)}) W_l \\ \boldsymbol{d}^{(k)} &= \boldsymbol{K}^{-1}\boldsymbol{b}^{(k)} \\ \boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(k)} + \boldsymbol{d}^{(k)} \end{aligned}\right\} \quad k = 0,\, 1,\, \dots \tag{40}$$

For the successive right-hand sides we find by recursion

$$\boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} - \boldsymbol{N}\boldsymbol{d}^{(k)} = (\boldsymbol{I} - \boldsymbol{N}\boldsymbol{K}^{-1})\boldsymbol{b}^{(k)} = \tilde{\boldsymbol{B}}^{k+1}\boldsymbol{b}^{(0)}\,, \tag{41}$$

where

$$\tilde{\boldsymbol{B}} = \boldsymbol{I} - \boldsymbol{N}\boldsymbol{K}^{-1}\,. \tag{42}$$

For the successive updates and errors we find

$$\boldsymbol{d}^{(k+1)} = \boldsymbol{K}^{-1}\boldsymbol{b}^{(k+1)} = (\boldsymbol{I} - \boldsymbol{K}^{-1}\boldsymbol{N})\boldsymbol{d}^{(k)} = \boldsymbol{B}^{k+1}\boldsymbol{d}^{(0)}\,, \tag{43}$$

$$\boldsymbol{e}^{(k+1)} = \boldsymbol{e}^{(k)} + \boldsymbol{d}^{(k)} = (\boldsymbol{I} - \boldsymbol{K}^{-1}\boldsymbol{N})\boldsymbol{e}^{(k)} = \boldsymbol{B}^{k+1}\boldsymbol{e}^{(0)}\,. \tag{44}$$

where

$$\boldsymbol{B} = \boldsymbol{I} - \boldsymbol{K}^{-1}\boldsymbol{N} \tag{45}$$

is called the iteration matrix (Axelsson 1996). Equations (41)–(45) are of great theoretical interest, as explained below, although none of them is used in the actual computations.

The convergence behaviour of the simple iteration scheme can largely be understood by means of these relations and especially in terms of the properties of the iteration matrix $\boldsymbol{B}$ governing the sequence of updates $\boldsymbol{d}$ and errors $\boldsymbol{e}$, and the adjunct matrix $\tilde{\boldsymbol{B}}$ governing the sequence of right-hand sides. It is well known (e.g., Axelsson 1996) that the simple iteration scheme in Eq. (40) converges (that is $\boldsymbol{x}^{(k)} \to \boldsymbol{x}$) for arbitrary starting approximation if and only if $\rho(\boldsymbol{B}) < 1$. Here, $\rho(\boldsymbol{B})$ is the spectral radius of $\boldsymbol{B}$, i.e., the largest absolute value of its eigenvalues. Under this condition we have $\boldsymbol{d}^{(k)} \to \boldsymbol{0}$ and $\boldsymbol{e}^{(k)} \to \boldsymbol{0}$ for $k \to \infty$. Also the right-hand side $\boldsymbol{b}^{(k)} = \boldsymbol{K}\boldsymbol{d}^{(k)} \to \boldsymbol{0}$ under the same condition.[8]

As discussed in Sect. 4.4, the normal matrix $\boldsymbol{N}$ is singular and its null space spanned by the $n \times 6$ matrix $\boldsymbol{V}$. Therefore,

$$\boldsymbol{B}\boldsymbol{V} = \boldsymbol{V} - \boldsymbol{K}^{-1}\boldsymbol{N}\boldsymbol{V} = \boldsymbol{V} \tag{46}$$

which shows that $\boldsymbol{B}$ has a six-fold eigenvalue equal to 1, with the corresponding eigenvectors spanning $\mathcal{N}(\boldsymbol{N})$. The spectral radius of $\boldsymbol{B}$ is therefore not less than 1, and the simple iteration scheme does not converge for arbitrary initial errors.

This is not a real problem, for the following reason. First, let us note that $\boldsymbol{B}$ is not a full-rank matrix. Indeed, a direct calculation of Eq. (45) using the expression for $\boldsymbol{K}$ in Eq. (37) shows that the first n_s columns of $\boldsymbol{B}$ are zero.[9] Thus (at least) n_s of its eigenvalues are identically zero. A corresponding set of orthogonal unit vectors is given by the columns of the $n \times n_s$ matrix

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{I} \\ \varnothing \end{bmatrix} \begin{matrix} \}\ n_s \text{ rows} \\ \}\ n - n_s \text{ rows} \end{matrix}\,, \tag{47}$$

so that $\boldsymbol{B}\boldsymbol{Z} = \boldsymbol{0}$. We assume that the remaining $n - n_s - 6$ eigenvalues satisfy $0 < |\lambda| < 1$. Thus, if $\boldsymbol{\Lambda}$ is the diagonal matrix containing these eigenvalues and $\boldsymbol{U}$ a matrix of size $n \times (n - n_s - 6)$ whose columns are made up of the corresponding eigenvectors, we have $\boldsymbol{B}\boldsymbol{U} = \boldsymbol{U}\boldsymbol{\Lambda}$. The columns of $\boldsymbol{Z}$, $\boldsymbol{U}$ and $\boldsymbol{V}$ together span $\mathbb{R}^n$, and it is therefore possible to decompose the solution vector as

$$\boldsymbol{x} = \boldsymbol{Z}\boldsymbol{x}_s + \boldsymbol{U}\boldsymbol{y} + \boldsymbol{V}\boldsymbol{z} \tag{48}$$

where $\boldsymbol{x}_s \in \mathbb{R}^{n_s}$ is the 'source' part of $\boldsymbol{x}$, $\boldsymbol{y} \in \mathbb{R}^{n-n_s-6}$ and $\boldsymbol{z} \in \mathbb{R}^6$. Since $-\boldsymbol{e}^{(0)} = \boldsymbol{x}$ we find

$$-\boldsymbol{e}^{(1)} = -\boldsymbol{B}\boldsymbol{e}^{(0)} = \boldsymbol{B}\boldsymbol{Z}\boldsymbol{x}_s + \boldsymbol{B}\boldsymbol{U}\boldsymbol{y} + \boldsymbol{B}\boldsymbol{V}\boldsymbol{z} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{y} + \boldsymbol{V}\boldsymbol{z}\,, \tag{49}$$

and by recursion

$$-\boldsymbol{e}^{(k)} = \boldsymbol{U}\boldsymbol{\Lambda}^k\boldsymbol{y} + \boldsymbol{V}\boldsymbol{z}\,. \tag{50}$$

The first term clearly vanishes for $k \to \infty$ if $\rho(\boldsymbol{\Lambda}) < 1$, as we have assumed. The second term, which is the projection of $\boldsymbol{x}$ on the null space of $\boldsymbol{N}$, remains unchanged by the iterations. The update vector can be written

$$\boldsymbol{d}^{(k)} = \boldsymbol{e}^{(k+1)} - \boldsymbol{e}^{(k)} = \boldsymbol{U}\boldsymbol{\Lambda}^k(\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{y}\,, \tag{51}$$

which vanishes under the same condition. The same is then true for the right-hand side $\boldsymbol{b}^{(k)} = \boldsymbol{K}\boldsymbol{d}^{(k)}$. Effectively, this means that we can ignore the singularity of $\boldsymbol{N}$ in the simple iteration scheme; it will converge to some valid solution of the (singular) normal equations, and the convergence process can be monitored by means of the vectors $\boldsymbol{d}^{(k)}$ and $\boldsymbol{b}^{(k)}$. After convergence, the required null-space components can be found and added by means of the frame rotator.

To summarize, we have shown that the simple iteration scheme converges in the desired way, provided that the spectral radius of $\boldsymbol{B}$, not counting the eigenvalues related to the null space of $\boldsymbol{N}$, is less than 1. It is well known (e.g., Golub & van Loan 1996) that this condition is satisfied for any symmetric positive definite $\boldsymbol{N}$, using the Gauss–Seidel preconditioner. However, the spectral radius may be very close to 1, implying very slow convergence. In the case of AGIS the situation is more complex, and convergence is in practice demonstrated through simulations (Sect. 7), but the theoretical background outlined above is of great help when interpreting the results.

[8] $\boldsymbol{B}$ and $\tilde{\boldsymbol{B}} = \boldsymbol{K}\boldsymbol{B}\boldsymbol{K}^{-1}$ have the same spectral radius; indeed, their eigenvalues are the same, as can be seen from the characteristic polynomial $\det(z\boldsymbol{I} - \boldsymbol{B}) = \det(\boldsymbol{K})(z\boldsymbol{I} - \boldsymbol{B})\det(\boldsymbol{K}^{-1}) = \det(z\boldsymbol{I} - \boldsymbol{K}\boldsymbol{B}\boldsymbol{K}^{-1})$ being the same for the two matrices (A. Bombrun, private communication).

[9] That the first n_s columns in the iteration matrix are zero means that the results of the next iteration are independent of the current source parameters $\boldsymbol{x}_s^{(k)}$. This may seem surprising at first, but is a simple consequence of the fact that each iteration starts with the source update block (S). In this block, the updated source parameters depend on the previous attitude, calibration and global parameters, but not on the previous source parameters.

4.6. Order of the block processes

In the preceding sections we have assumed that the four blocks are always executed in the sequence SACG, and that the updated parameters resulting from each block is used in the subsequent blocks. The particular preconditioner $\boldsymbol{K}$ in Eq. (37) incorporates these assumptions. We will now discuss variants of this scheme and show that they can be mathematically represented by different preconditioners.

Each AGIS iteration always starts with the source update block (S). The main reason for starting with S rather than A, for example, is that the observations can be inspected and analysed one source at a time in order to estimate the quality of the data, identify possible outliers, and check whether the source is suitable as a primary source. The identification of outliers, in particular, requires several 'inner' iterations of the source observation equations (Sect. 5.1.2), which can be done with relative ease because of the small number of data points involved.

The order of the remaining blocks ACG adopted in the preceding sections is more or less arbitrary. It is easy to see through an analogy with Eqs. (37)–(38) that the sequence SCAG (for example) is mathematically represented by the alternative preconditioner

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{N}_{ss} & \varnothing & \varnothing & \varnothing \\ \boldsymbol{N}_{as} & \boldsymbol{N}_{aa} & \boldsymbol{N}_{ac} & \varnothing \\ \boldsymbol{N}_{cs} & \varnothing & \boldsymbol{N}_{cc} & \varnothing \\ \boldsymbol{N}_{gs} & \boldsymbol{N}_{ga} & \boldsymbol{N}_{gc} & \boldsymbol{N}_{gg} \end{bmatrix}, \tag{52}$$

and that other permutations of the sequence are similarly obtained by transposing the corresponding off-diagonal blocks. Changing the preconditioner means changing the iteration matrix (45) and therefore possibly also its eigenvalues, which in turn govern the rate of convergence.

From a mathematical viewpoint, the choice of the starting block (S in our case) determines the initial update $\boldsymbol{d}^{(0)}$ (cf. footnote 9) and therefore influences all subsequent updates. However, this particular choice is not expected to have much influence on the convergence rate after a number of iterations, since the S block has no special status in the periodically repeated sequence …SACGSACGSACGSA…. Thus we may surmise that the different iteration matrices for the cyclically permuted sequences SACG, ACGS, CGSA and GSAC do in fact have the same set of eigenvalues. For symmetry reasons it can also be surmised that the reversed sequences GCAS, SGCA, ASGC and CASG have the same eigenvalues as the original sequences. Thus, there are in fact only three sets of sequences with possibly distinct convergence behaviour, represented by SACG (or SGCA), SAGC (or SCGA), and SCAG (or SGAC) if we take S to be the starting block. There is no obvious way of knowing a priori which of the three possibilities is to be preferred, or even if they are significantly different.

Apart from the various permutations discussed above, there is another way to modify the AGIS iteration scheme, which can also be described in terms of the preconditioner. This modification is related to the practical organization of the flow of data to the different block processes. The current implementation of the simple iteration scheme differs somewhat from the description in Sect. 4.5, and the block updates are in fact practically organized as follows:

1. Initialize the iteration counter, $k = 0$
2. Choose starting values for all unknowns: $\boldsymbol{s}^{(0)}$, $\boldsymbol{a}^{(0)}$, $\boldsymbol{c}^{(0)}$, $\boldsymbol{g}^{(0)}$.
3. Estimate $\boldsymbol{s}^{(k+1)}$ using $\boldsymbol{a}^{(k)}$, $\boldsymbol{c}^{(k)}$, $\boldsymbol{g}^{(k)}$.
4. Estimate $\boldsymbol{a}^{(k+1)}$ using $\boldsymbol{s}^{(k+1)}$, $\boldsymbol{c}^{(k)}$, $\boldsymbol{g}^{(k)}$.
5. Estimate $\boldsymbol{c}^{(k+1)}$ using $\boldsymbol{s}^{(k+1)}$, $\boldsymbol{a}^{(k)}$, $\boldsymbol{g}^{(k)}$.
6. Estimate $\boldsymbol{g}^{(k+1)}$ using $\boldsymbol{s}^{(k+1)}$, $\boldsymbol{a}^{(k)}$, $\boldsymbol{c}^{(k)}$.
7. Increment k and go to Step 3.

The crucial difference compared with Eq. (38) is that the C block in Step 5 does not use the updated attitude but the old one, and that the G block in Step 6 similarly uses the 'old' attitude and calibration parameters. This has the practical advantage that the A, C and G blocks can be carried out in parallel, with big savings in terms of the amounts of data that have to be exchanged between the processes (see Sect. 7). Schematically, this can be represented as S[ACG], where the blocks in brackets are (or can be) executed in parallel (and so the order of the bracketed blocks does not matter). The corresponding preconditioner is

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{N}_{ss} & \varnothing & \varnothing & \varnothing \\ \boldsymbol{N}_{as} & \boldsymbol{N}_{aa} & \varnothing & \varnothing \\ \boldsymbol{N}_{cs} & \varnothing & \boldsymbol{N}_{cc} & \varnothing \\ \boldsymbol{N}_{gs} & \varnothing & \varnothing & \boldsymbol{N}_{gg} \end{bmatrix}. \tag{53}$$

Yet other variants may be considered, for example S[AC]G, where Step 6 uses the updated attitude and calibration parameters, with preconditioner

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{N}_{ss} & \varnothing & \varnothing & \varnothing \\ \boldsymbol{N}_{as} & \boldsymbol{N}_{aa} & \varnothing & \varnothing \\ \boldsymbol{N}_{cs} & \varnothing & \boldsymbol{N}_{cc} & \varnothing \\ \boldsymbol{N}_{gs} & \boldsymbol{N}_{ga} & \boldsymbol{N}_{gc} & \boldsymbol{N}_{gg} \end{bmatrix}, \tag{54}$$

and [SACG], for which

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{N}_{ss} & \varnothing & \varnothing & \varnothing \\ \varnothing & \boldsymbol{N}_{aa} & \varnothing & \varnothing \\ \varnothing & \varnothing & \boldsymbol{N}_{cc} & \varnothing \\ \varnothing & \varnothing & \varnothing & \boldsymbol{N}_{gg} \end{bmatrix}. \tag{55}$$

This last case is known as the (block) Jacobi method, while the use of a full triangular preconditioner as in Eq. (37) is known as the (block) Gauss–Seidel method (Axelsson 1996). As we have seen, the currently implemented simple iteration scheme is intermediate between the Jacobi and Gauss–Seidel methods.

Intuitively, we expect the Gauss–Seidel method to converge more quickly than the Jacobi or any intermediate method, simply because each block then uses the most recent (and, presumably, best) estimates of the parameters. However, our practical experience with AGIS shows that the 'difficult' part of the problem is to disentangle the source and attitude parameters. For example, the calibration parameters are generally found to converge much faster than the source and attitude parameters. Thus it does not seem to matter much if Step 5 above uses $\boldsymbol{a}^{(k+1)}$ or $\boldsymbol{a}^{(k)}$, i.e., whether the sub-matrix $\boldsymbol{N}_{ca}$ is included or not in the preconditioner. A similar argument can be made concerning the G block, provided some measures are taken to decorrelate the global parameters from the source parameters (Sect. 5.4). Thus, the various intermediate methods are probably nearly as good as the Gauss–Seidel method, in terms of the convergence rate, and the precise scheme may then rather be determined by practical considerations. With the present data processing architecture, the favoured scheme is S[ACG] as described above.

4.7. Accelerated iteration, conjugate gradients and the hybrid iteration scheme

The 'simple iteration' (SI) scheme described above was the starting point for a long development towards a fully functional scheme with much improved convergence properties. The main stages in this development were the 'accelerated simple iteration' (ASI), the conjugate gradients (CG), and finally the fully flexible 'hybrid scheme' (A/SI-CG) to be used in the final implementation of AGIS. As much of this development has at most historical interest, only a brief outline is given here.

Already in the very early implementation of the simple iteration scheme it was observed that convergence was slower than (naively) expected, and that after some iterations, the updates always seemed to go in the same direction, forming a geometrically (exponentially) decreasing series. With the hindsight of the analysis in Sect. 4.5 this behaviour is very easily understood: the persistent pattern of updates is roughly proportional to the eigenvector of the largest eigenvalue of the iteration matrix, and the (nearly constant) ratio of the sizes of successive updates is the corresponding eigenvalue. From this realization it was natural to test an acceleration method based on a Richardson-type extrapolation of the updates. The idea is simply that if the updates in two successive iterations are roughly proportional to each other, $\boldsymbol{d}^{(k+1)} \simeq \lambda\boldsymbol{d}^{(k)}$, with $|\lambda| < 1$, then we can infer that the next update is again a factor λ smaller than $\boldsymbol{d}^{(k+1)}$, and so on. The sum of all the updates after iteration k can therefore be estimated as $\boldsymbol{d}^{(k+1)} + \lambda\boldsymbol{d}^{(k+1)} + \lambda^2\boldsymbol{d}^{(k+1)} + \cdots = (1-\lambda)^{-1}\boldsymbol{d}^{(k+1)}$. Thus, in iteration $k+1$ we apply an acceleration factor $1/(1-\lambda)$ based on the current estimate of the ratio λ. This accelerated simple iteration (ASI) scheme is seen to be a variant of the well-known successive overrelaxation method (Axelsson 1996). The factor λ is estimated by statistical analysis of the parallax updates for a small fraction of the sources; the parallax updates are used for this analysis, since they are unaffected by a possible change in the frame orientation between successive iterations. With this simple device, the number of iterations for full convergence was reduced roughly by a factor 2.

Both the simple iteration and the accelerated simple iteration belongs to a much more general class of solution methods known as Krylov subspace approximations. The sequence of updates $\boldsymbol{d}^{(k)}$, $k = 0\dots K-1$ generated by the first K simple iterations constitute the basis for the K-dimensional subspace of the solution space, known as the Krylov subspace for the given matrix and right-hand side (e.g., Greenbaum 1997; van der Vorst 2003). Krylov methods compute approximations that, in the kth iteration, belongs to the k-dimensional Krylov subspace. But whereas the simple and accelerated iteration schemes, in the kth iteration, use updates that are just proportional to the kth basis vector, more efficient algorithms generate approximations that are (in some sense) optimal linear combinations of all k basis vectors. Conjugate gradients (CG) is one of the best-known such methods, and possibly the most efficient one for general symmetric positive-definite matrices. (e.g., Axelsson 1996; Björck 1996; van der Vorst 2003). Its implementation within the AGIS framework is more complicated, but has been considered in detail by Bombrun et al. (2011). As it provides significant advantages over the SI and ASI schemes in terms of convergence speed, this algorithm has been chosen as the baseline method for the astrometric core solution of Gaia (see below however). From practical experience, we have found that CG is roughly a factor 2 faster than ASI, or a factor 4 faster than the SI scheme. Like SI, the CG algorithm uses a preconditioner and can be formulated in terms of the S, A, C and G blocks, so the subsequent description of these blocks remains valid. In the terminology of Bombrun et al. (2011) the process of solving the preconditioner system $\boldsymbol{Kd} = \boldsymbol{b}$ is the *kernel* operation common to all these solution methods, which only differ in how the updates are applied according to the various *iteration schemes*.

The CG algorithm assumes that the normal matrix is constant in the course of the iterations. This is not strictly true if the observation weights are allowed to change as functions of the residuals, as will be required for efficient outlier elimination (Sect. 5.1.2). Using the CG algorithm together with the weight-adjustment scheme described below could therefore lead to instabilities, i.e., a reduced convergence rate or even non-convergence. On the other hand, the SI scheme is extremely stable with respect to all such modifications in the course of the iterations, as can be expected from the interpretation of the SI scheme as the successive and independent application of the different solution blocks. The finally adopted algorithm is therefore a *hybrid scheme* combining SI (or ASI) and CG, where SI is used initially, until the weights have settled, after which CG is turned on. A temporary switch back to SI, with an optional re-adjustment of the weights, may be employed after a certain number of CG iterations; this could avoid some problems due to the accumulation of numerical rounding errors in CG.

5. Updating processes

In this section we describe in some detail each of the updating blocks S, A, C and G that form the basis (or kernel process) for the AGIS iteration loop.

5.1. Source updating (S)

5.1.1. The normal equations

The astrometric model for the sources is given in Sect. 3.2. In the source update block (S) the source parameters $\boldsymbol{s}$ are improved by solving the first line in Eq. (38). According to Eqs. (29) and (32) this can be done for one source (i) at a time by solving the following system of equations for the update $\boldsymbol{d}_i$ of the five astrometric parameters in $\boldsymbol{s}_i$:

$$\left[\sum_{l\in i}\frac{\partial R_l}{\partial \boldsymbol{s}_i}\frac{\partial R_l}{\partial \boldsymbol{s}_i'}W_l\right]\boldsymbol{d}_i = -\sum_{l\in i}\frac{\partial R_l}{\partial \boldsymbol{s}_i}R_l(\boldsymbol{s}_i)W_l\,. \qquad (56)$$

Here $W_l = w_l/(\sigma_l^2 + \epsilon_l^2)$, with σ_l denoting the given formal standard uncertainty of observation l, expressed as an angle. w_l and ϵ_l are, respectively, the downweighting factor and excess noise introduced in Sect. 3.6. In Eq. (56) the dependence of R_l on $\boldsymbol{a}$, $\boldsymbol{c}$ and $\boldsymbol{g}$ has been suppressed, since the system is solved with these parameters fixed.

In matrix notation, the normal equations (56) can be written

$$\boldsymbol{A}_i'\boldsymbol{W}_i\boldsymbol{A}_i\,\boldsymbol{d}_i = \boldsymbol{A}_i'\boldsymbol{W}_i\boldsymbol{h}_i\,, \qquad (57)$$

where

$$\boldsymbol{A}_i = \left[-\frac{\partial R_l}{\partial \boldsymbol{s}_i'}\right]_{l\in i} \qquad (58)$$

is an $n_i \times 5$ matrix with $n_i = \sum_{l\in i} 1$ the number of observations of source i (typically $n_i \sim 800$),

$$\boldsymbol{h}_i = [R_l(\boldsymbol{s}_i)]_{l\in i} \qquad (59)$$

is a column matrix of length n_i, and $\boldsymbol{W}_i$ is a diagonal matrix containing the weight factors W_l. Although the residuals R_l are in principle non-linear functions of $\boldsymbol{s}_i$, this non-linearity can be neglected if the parameters are close enough to the final solution, i.e., if the resulting update $\boldsymbol{d}_i$ is small enough. The partial derivatives in $\boldsymbol{A}_i$ can then be regarded as fixed throughout the updating process, and $\boldsymbol{A}_i$ and $\boldsymbol{h}_i$ can immediately be computed when entering the source updating. The weight matrix $\boldsymbol{W}_i$, on the other hand, depends on w_l and ϵ_l, which are modified as part of the process as explained below.

For given $\boldsymbol{W}_i$ the system of equations (57) is solved using the Cholesky algorithm (Appendix C). At the end of the source updating, the full (5×5) inverse matrix is computed, providing a first-order estimate of the covariance of the astrometric parameters in $\boldsymbol{s}_i$ (see Sect. 6.3).

We note that Eq. (57) corresponds to the overdetermined system of observation equations

$$\boldsymbol{A}_i\,\boldsymbol{d}_i \simeq \boldsymbol{h}_i \quad \text{(weight matrix } \boldsymbol{W}_i) \, . \tag{60}$$

After solution, the updated residuals are contained in the vector $\boldsymbol{R}_i = \boldsymbol{h}_i - \boldsymbol{A}_i\boldsymbol{d}_i$, and the contribution of the source to the objective function Q is given by

$$Q_i = \boldsymbol{R}_i'\boldsymbol{W}_i\boldsymbol{R}_i \, . \tag{61}$$

5.1.2. The inner iteration: identifying outliers and estimating the excess source noise

Since the weight matrix $\boldsymbol{W}_i$ depends on the downweighting factors w_l and the excess noises ϵ_l, which in turn depend on the updated residuals $\boldsymbol{R}_i$, the normal equations (57) are non-linear and must be solved by iteration. We refer to this as the *inner iteration* of the source update, to distinguish it from the AGIS iteration discussed in Sect. 4.5.

Using Eq. (28) we can write

$$W_l = \frac{w_l}{\sigma_l^2 + \epsilon_a^2(t_l) + \epsilon_i^2} = \frac{w_l}{\tilde{\sigma}_l^2 + \epsilon_i^2} \, , \tag{62}$$

where $\tilde{\sigma}_l = \left[\sigma_l^2 + \epsilon_a^2(t_l)\right]^{1/2}$ is the formal standard uncertainty of the observation adjusted for the excess attitude noise, which, when entering the source update, is assumed to be known from a previous attitude update (Sect. 5.2.5). In the very first source update the excess attitude noise must be set to the mean errors of the pre-AGIS attitude parameter estimates which are used for starting up the iterations. The downweighting factors depend on the normalized residuals, i.e.,

$$w_l = w\left(\frac{R_l}{\sqrt{\tilde{\sigma}_l^2 + \epsilon_i^2}}\right) \, , \tag{63}$$

where the function $w(z)$ is such that $w(z) = 1$ for $|z| \lesssim 3$ and gradually decreasing to 0 for larger $|z|$ (see Eq. 66).

The inner iteration actually consists of two nested procedures: an outer one which determines the downweighting factors w_l, and an inner one which determines the excess source noise ϵ_i for a fixed set of w_l. We begin by considering the inner procedure.

For fixed downweighting factors, the source update aims to minimize Q_i in Eq. (61) with respect to the unknown $\boldsymbol{d}_i$ and ϵ_i. But it is immediately seen that Q_i can be made arbitrarily small just by making ϵ_i large enough. Consequently, we cannot use unconstrained minimization to solve this problem. The necessary constraint is provided by the condition that Q_i, under the assumption that the excess noises have been correctly estimated and the outliers properly removed, should follow the chi-square distribution with $\nu = n_i - n_{\text{out}} - 5$ degrees of freedom. Here n_i is the number of observations of source i, n_{out} the number of outliers and 5 the number of astrometric parameters estimated. In particular, the expected value is $\mathrm{E}(Q_i) = \nu$. The number of outliers n_{out} is estimated by counting the number of observations with $w_l < 0.2$. This limit was empirically found to give a reasonable estimate of the actual number of outliers in a variety of simulated cases.

For given ϵ_i the weight matrix is now known, Eq. (57) can be trivially solved and the residuals $\boldsymbol{R}_i$ computed. We thus define the function $Q(\epsilon_i^2) = \boldsymbol{R}_i'\boldsymbol{W}_i\boldsymbol{R}_i$. The excess source variance is then taken to be

$$\epsilon_i^2 = \begin{cases} 0 & \text{if } Q(0) \le \nu, \\ \text{solution of } Q(\epsilon_i^2) = \nu & \text{otherwise.} \end{cases} \tag{64}$$

In the second case, the non-linear equation $Q(y) = \nu$ is iteratively solved by a series of improvements $\Delta y = (1 - Q(y)/\nu)Q(y)/Q'(y)$, starting from $y = 0$. Typically, 2–3 iterations are sufficient.[10]

This procedure returns a positive ϵ_i as soon as $Q(0) > \nu$. If the resulting ϵ_i is much smaller than the typical σ_l of the source, it is probably not significant. The significance of the ϵ_i can more easily be judged from an auxiliary statistic that can be computed almost for free. Under the null hypothesis ($\epsilon_i = 0$) we know that $Q(0) \sim \chi^2_\nu$, so the expected value is ν and the variance 2ν. Thus we may take

$$D = \frac{Q(0) - \nu}{\sqrt{2\nu}} \tag{65}$$

as a measure of the significance of the estimated ϵ_i, with $D > 2$ indicating a probably significant value.

Having determined ϵ_i, and a corresponding set of residuals $\boldsymbol{R}_i$, the downweighting factors can immediately be computed from Eq. (63). However, having changed the downweighting factors (and possibly ν) it is now necessary to repeat the estimation of ϵ_i and $\boldsymbol{d}_i$ with the new set w_l. Typically, four such iterations are found to be sufficient. The only remaining problem is how to start the iterations, that is the initial selection of the downweights w_l. It is not possible to start by assuming $w_l = 1$ for all the observations, since we must take into account that some small fraction of the data could be utterly wrong. Such gross outliers, if not removed already from the start, would severely slow down or even prevent the convergence of the inner iterations. The adopted solution is to make an extremely robust estimation of the standard deviation of the initial residuals (contained in $\boldsymbol{h}_i$), from which the initial downweightings are obtained. This robust standard deviation ς_i is calculated as half the intersextile range of the elements in $\boldsymbol{h}_i$, whereupon the initial $w_l = w(h_l/\varsigma_i)$.

After convergence of the inner iteration, the statistical weight of the source W_i is computed according to Eq. (119). This quantity, together with ϵ_i and the magnitude, are the most important indicators for the selection of primary sources (Sect. 6.2.2).

The weight function $w(z)$ currently used is the following:

$$w(z) = \begin{cases} 1 & \text{if } |z| \le 2 \\ 1 - 1.773735t^2 + 1.141615t^3 & \text{if } 2 \le |z| < 3 \\ \exp(-|z|/3) & \text{otherwise,} \end{cases} \tag{66}$$

where $t = |z| - 2$ and the numerical constants have been chosen to make a smooth transition at $|z| = 3$. The exponential decay for $|z| > 3$ provides a dramatic weight reduction for large residuals; e.g., at 10 sigmas we have $w(10) \simeq 0.036$, while at 100 sigmas we have $w(100) \simeq 3 \times 10^{-15}$.

[10] This iteration formula can be derived by matching the rational approximation $Q(y) \simeq a/(b + y)$ to the value and derivative of $Q(y)$ at the current point y. Compared with the standard Newton–Raphson method, which uses a linear approximation around the current y, the present formula converges much quicker due to the more reasonable behaviour of the rational approximation especially for large y.

Among the many different weight functions proposed in the literature, the so-called Huber estimator (Huber 1981) using

$$w_{\rm H}(z) = \begin{cases} 1 & \text{if } |z| \le c \\ c(2|z| - c)z^{-2} & \text{otherwise} \end{cases} \tag{67}$$

has been considered as an alternative to Eq. (66), e.g., with $c = 2$. This gives a much slower weight decay for large residuals, e.g., $w_{\rm H}(10) = 0.36$ and $w_{\rm H}(100) = 0.0396$. Future experiments may decide which weight function will finally be used for AGIS.

5.1.3. Calculation of partial derivatives

The calculation of the partial derivatives in Eq. (58) is done as follows. From Eqs. (25) and (26) we have, by means of (13),

$$-\frac{\partial R_l^{\rm AL}}{\partial \boldsymbol{s}_i'} = \frac{\partial \varphi_l}{\partial \boldsymbol{s}_i'}\,, \qquad -\frac{\partial R_l^{\rm AC}}{\partial \boldsymbol{s}_i'} = \frac{\partial \zeta_l}{\partial \boldsymbol{s}_i'}\,. \tag{68}$$

In analogy with Eq. (5) we introduce the auxiliary vectors $\boldsymbol{m}_l = \langle \boldsymbol{z} \times \boldsymbol{u}_l \rangle$ and $\boldsymbol{n}_l = \boldsymbol{u}_l \times \boldsymbol{m}_l$, which together with $\boldsymbol{u}_l$ form the normal triad $[\boldsymbol{m}_l\ \boldsymbol{n}_l\ \boldsymbol{u}_l]$ with respect to the SRS; its components in the SRS are given by the columns of the matrix

$$\mathsf{S}'[\boldsymbol{m}_l\ \boldsymbol{n}_l\ \boldsymbol{u}_l] = \begin{bmatrix} -\sin\varphi_l & -\sin\zeta_l\cos\varphi_l & \cos\zeta_l\cos\varphi_l \\ \cos\varphi_l & -\sin\zeta_l\sin\varphi_l & \cos\zeta_l\sin\varphi_l \\ 0 & \cos\zeta_l & \sin\zeta_l \end{bmatrix}. \tag{69}$$

By differentiation of the last column we obtain

$$\frac{\partial \boldsymbol{u}_l}{\partial \boldsymbol{s}_i'} = \boldsymbol{m}_l \frac{\partial \varphi_l}{\partial \boldsymbol{s}_i'} \cos\zeta_l + \boldsymbol{n}_l \frac{\partial \zeta_l}{\partial \boldsymbol{s}_i'}\,, \tag{70}$$

and thus

$$-\frac{\partial R_l^{\rm AL}}{\partial \boldsymbol{s}_i'} = \boldsymbol{m}_l' \frac{\partial \boldsymbol{u}_l}{\partial \boldsymbol{s}_i'} \sec\zeta_l\,, \qquad -\frac{\partial R_l^{\rm AC}}{\partial \boldsymbol{s}_i'} = \boldsymbol{n}_l' \frac{\partial \boldsymbol{u}_l}{\partial \boldsymbol{s}_i'}\,. \tag{71}$$

These expressions can be evaluated in any coordinate system, but perhaps most conveniently in the SRS using $\mathsf{S}'\boldsymbol{m}_l$ and $\mathsf{S}'\boldsymbol{n}_l$ from Eq. (69), and

$$\left\{ \frac{\partial(\mathsf{S}'\boldsymbol{u}_l)}{\partial \boldsymbol{s}_i'}, 0 \right\} = \mathbf{q}_l^{-1} \left\{ \frac{\partial(\mathsf{C}'\boldsymbol{u}_l)}{\partial \boldsymbol{s}_i'}, 0 \right\} \mathbf{q}_l \tag{72}$$

from Eq. (11). To compute the partial derivatives of $\mathsf{C}'\boldsymbol{u}_i$, we first obtain from Eq. (4) the derivatives[11] of the coordinate direction as:

$$\frac{\partial \bar{\boldsymbol{u}}_l}{\partial \alpha_{*i}} = \boldsymbol{p}_i\,, \quad \frac{\partial \bar{\boldsymbol{u}}_l}{\partial \delta_i} = \boldsymbol{q}_i\,, \quad \frac{\partial \bar{\boldsymbol{u}}_l}{\partial \varpi_i} = -(\boldsymbol{I} - \boldsymbol{r}_i\boldsymbol{r}_i')\boldsymbol{b}_{\rm G}(t_l)/A_{\rm u}\,,$$
$$\frac{\partial \bar{\boldsymbol{u}}_l}{\partial \mu_{\alpha*i}} = \boldsymbol{p}_i\tau_l\,, \quad \frac{\partial \bar{\boldsymbol{u}}_l}{\partial \mu_{\delta i}} = \boldsymbol{q}_i\tau_l\,,$$
$$\frac{\partial \bar{\boldsymbol{u}}_l}{\partial \mu_{ri}} = (\boldsymbol{I} - \boldsymbol{r}_i\boldsymbol{r}_i')\boldsymbol{b}_{\rm G}(t_l)\varpi_i\tau_l/A_{\rm u} - (\boldsymbol{p}_i\mu_{\alpha*i} + \boldsymbol{q}_i\mu_{\delta i})\tau_l^2\,, \tag{73}$$

where $\tau_l = t_{{\rm B}l} - t_{\rm ep}$ for brevity, and we have used the normal triad in the celestial reference system, Eq. (5). Although usually only the first five derivatives are needed (see however Sect. 6.3), the last equation gives, for completeness, the derivative with respect to the sixth astrometric parameter μ_{ri}: the first term corresponds to the secular change in parallax and the second to the perspective acceleration.

The rigorous transformation from $\bar{\boldsymbol{u}}_l$ to $\boldsymbol{u}_l$ is quite complex, but by far the largest difference ($\sim 10^{-4}$ rad) is due to stellar aberration. By comparison, gravitational light deflection by the Sun is $\sim 2 \times 10^{-8}$ rad. While the rigorous transformation is required to compute the vector $\boldsymbol{u}_l$ itself, some simplifications can be accepted when computing the partial derivatives. Indeed, for this purpose it is sufficient to consider the classical stellar aberration formula,

$$\boldsymbol{u}_l \simeq \langle \bar{\boldsymbol{u}}_l + \boldsymbol{v}_{\rm G}(t_l)c^{-1} \rangle\,, \tag{74}$$

accurate to first order in $v_{\rm G}/c$, where $\boldsymbol{v}_{\rm G} = {\rm d}\boldsymbol{b}_{\rm G}/{\rm d}t$ is the barycentric coordinate velocity of Gaia and c the speed of light. To a relative precision better than 10^{-6} we then have

$$\frac{\partial \boldsymbol{u}_l}{\partial \boldsymbol{s}_i'} \simeq \left[\left(1 - \frac{\bar{\boldsymbol{u}}_l'\boldsymbol{v}_{\rm G}}{c} \right) \boldsymbol{I} - \frac{\bar{\boldsymbol{u}}_l\boldsymbol{v}_{\rm G}'}{c} \right] \frac{\partial \bar{\boldsymbol{u}}_l}{\partial \boldsymbol{s}_i'}\,, \tag{75}$$

where $\boldsymbol{I}$ is the 3×3 identity matrix.

5.2. Attitude updating (A)

5.2.1. The normal equations

The attitude model using B-splines to represent the components of the attitude quaternion as functions of time is described in Sect. 3.3. (For reference purposes, conventions for notation and some important properties of splines and B-splines are explained in Appendix B.1.) In the attitude update process (A) the attitude parameters $\boldsymbol{a}$ are improved by solving the second sub-system in Eq. (38). Recalling that $\boldsymbol{a}$ is divided into sub-vectors of length 4, representing the quaternions $\mathbf{a}_n$ in Eq. (10), the nth set of four equations can be written, using Eqs. (29) and (33),

$$\sum_{m=n-M+1}^{n+M-1} \left[\sum_{l \in L_n \cap L_m} \frac{\partial R_l}{\partial \mathbf{a}_n} \frac{\partial R_l}{\partial \mathbf{a}_m'} W_l \right] \boldsymbol{d}_m = -\sum_{l \in L_n} \frac{\partial R_l}{\partial \mathbf{a}_n} R_l(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{g}) W_l\,. \tag{76}$$

Here L_n stands for the set of observations occurring within the support of $B_n(t)$, i.e., $L_n = \{ l \,|\, \tau_n \le t_l < \tau_{n+M} \}$, where M is the order of the spline. On the right-hand side, it is understood that the residuals R_l are calculated for the most recent source parameters $\boldsymbol{s}$ (i.e., from the preceding source updating), while the attitude, calibration and global parameters are the not-yet-updated ones, as explained in Sect. 4.6. The weights W_l are the ones calculated in the source updating.

The structure of the symmetric matrix $\boldsymbol{N}_{aa}$ is shown in Fig. 5. If each quaternion component is represented by a spline of order M with N degrees of freedom (so $n = 0 \ldots N-1$), the total number of unknowns is $4N$ and the average bandwidth of $\boldsymbol{N}_{aa}$ (counting non-zero elements from the diagonal up) is $4(M-1) + 2.5$. Including the right-hand side, the total number of reals that need to be stored for the normal equations is therefore $\simeq (16M-2)N$ or $62N$ for cubic splines. With a knot interval of about 15 s, about 3 MB is required to store the attitude normal equations for one day of observations. Thus it is completely realistic to store the attitude normal equations for the entire mission in primary memory. Cholesky factorization of the normal equations does not produce any more non-zero elements in the matrix; the factorization and solution can therefore use the same storage as the normal equations. Moreover, since the number of arithmetic operations grows only linearly with N, it is computationally feasible to solve the normal equations for any stretch of data.

[11] As indicated by the asterisk in the first derivative in Eq. (73), the differential in right ascension is a true arc, thus: $\partial\alpha_{*i} \equiv (\partial\alpha_i)\cos\delta_i$. The corresponding update in right ascension, i.e., the first element of the $\boldsymbol{d}_i$ obtained by solving (57), is therefore $\Delta\alpha_i \cos\delta_i$.

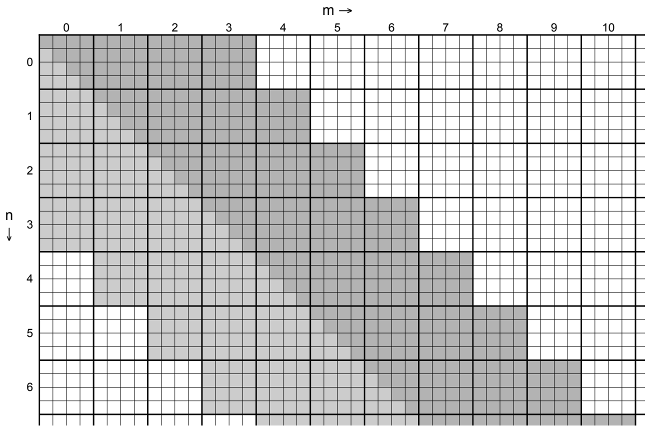


Fig. 5. Structure of the attitude normal equations matrix N_{aa} for a cubic spline (order $M = 4$). The blocks of size 4×4 are indexed by (n, m) as in Eq. (33). The grey cells represent non-zero elements. Since the matrix is symmetric, the elements below the diagonal (in lighter grey) need not be stored.

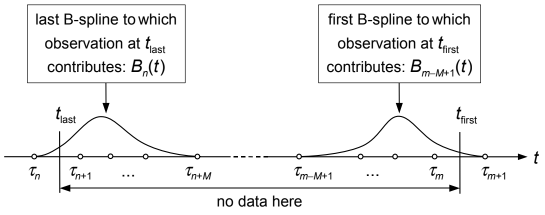

last B-spline to which observation at $t_{\rm last}$ contributes: $B_n(t)$
first B-spline to which observation at $t_{\rm first}$ contributes: $B_{m-M+1}(t)$
$t_{\rm last}$
$t_{\rm first}$
t
τ_n
τ_{n+1}
...
τ_{n+M}
τ_{m-M+1}
...
τ_m
τ_{m+1}
no data here

Fig. 6. A natural break in the definition of the attitude spline occurs if there is a gap in the observations containing at least M knots, where M is the order of the spline. $t_{\rm last}$ is the time of the last observation before the gap, $t_{\rm first}$ the time of the first observation after the gap.

5.2.2. Segmentation of the data

Even though it is feasible to treat the complete set of normal equations for the attitude updating as a single system, it is desirable for several reasons to divide up the data temporally. This allows one to set up a very straightforward and efficient distributed attitude updating, simply by handing out the processing of different time segments to different computing nodes. Also the inspection of residuals in order to detect stretches of bad fit (caused, for example, by micrometeoroid impacts), and the subsequent reprocessing of these stretches, is greatly facilitated if it can be done on shorter data segments.

The spline model is capable of interpolating sensibly (if not accurately) over short data gaps. However, if the data gap contains at least M knots (with $M = 4$ for cubic splines), the two splines on each side of the gap become completely disconnected. This is illustrated in Fig. 6, where n and m are the left indices of, respectively, the last observation before the gap ($t_{\rm last}$) and the first observation after the gap ($t_{\rm first}$). $B_n(t)$ and $B_{m-M+1}(t)$ are the last and first B-splines whose coefficients depend on the observations before and after the gap. Clearly the two segments of the attitude spline are disconnected if $n < m - M + 1$ or $m - n \geq M$. We call this a natural attitude break.

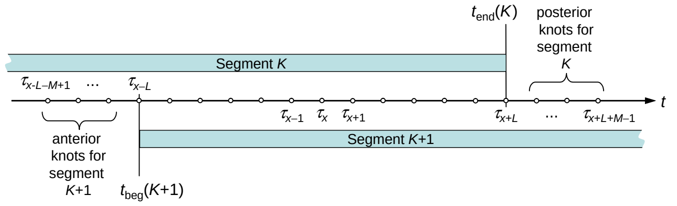


Fig. 7. Illustrating the assignment of knots for the attitude update solutions in two consecutive segments K and $K+1$, with a breakpoint at knot index x and using $2L$ overlapping knot intervals.

In the absence of natural breaks, artificial ones can be introduced at suitable intervals by a simple method and without any significant loss of accuracy. The idea is to make separate solutions for overlapping segments, as illustrated in Fig. 7. The segments use a common knot sequence $\{\tau_k\}$ that may extend over the whole length of the mission. Each segment K defines an attitude spline in the interval $[t_{\rm beg}(K), t_{\rm end}(K)]$, based on data with observation times in that same interval. The endpoints coincide with certain knots in such a way that $t_{\rm end}(K) = \tau_{x+L}$ and $t_{\rm beg}(K+1) = \tau_{x-L}$, where τ_x is the cross-over knot between segments K and $K+1$ and $2L$ the number of overlapping knot intervals. The anterior and posterior knots for each segment are also taken from the common knot sequence. The local character of the splines means that the resulting fit around τ_x is practically the same for the two segments, provided L is large enough. For cubic splines ($M = 4$) it is found that $L = 12$ is sufficient.

Each segment gives a system of normal equations (76) for the updates $\boldsymbol{d}_n$ to the attitude parameters $\mathbf{a}_n$ for a certain range of index n. For example, with reference to Fig. 7, in segment K updates are computed up to and including $n = x+L-1$, while in segment $K+1$ updates are computed starting with $n = x-L-M+1$. At least in the middle part of the overlap region, the updates for a given n should be essentially the same in the two segments. It therefore does not matter much which of the results is chosen. The mid-point is at index $n = x - M/2$ if M is even, or half-way between $x - (M+1)/2$ and $x - (M-1)/2$ if M is odd. We may therefore agree to use the solution from segment K to update $\mathbf{a}_n$ up to and including index $n = x - \lfloor M/2 \rfloor$, while the solution from segment $K+1$ is used starting with $n = x - \lfloor M/2 \rfloor + 1$. The important thing is that no n is missed by the updating, nor updated twice.

Once all the coefficients $\mathbf{a}_n$ have been updated from the different segmented solutions, the segmentation loses its meaning and can in principle be forgotten. For example, when evaluating the attitude at a specific time t, it does not matter to which segment that instant belonged. In the next iteration of the attitude update, a different segmentation can in principle be used.

The overlapping segments mean that a fraction of the observations need to be processed twice in the attitude updating. The fraction equals the ratio of the overlap length to the mean length of the segment, and increases the shorter the segments are made. For example, with a segment length of one day (it would not seem reasonable to have shorter segments) and a mean knot interval of 15 s, the fractional overlap for $L = 12$ is only 0.4%.

5.2.3. Calculation of partial derivatives

For the partial derivatives we obtain in analogy with Eq. (71)

$$-\frac{\partial R_l^{\rm AL}}{\partial \mathbf{q}_l} = \boldsymbol{m}_l' \, \frac{\partial \boldsymbol{u}_l}{\partial \mathbf{q}_l} \sec \zeta_l \,, \qquad -\frac{\partial R_l^{\rm AC}}{\partial \mathbf{q}_l} = \boldsymbol{n}_l' \, \frac{\partial \boldsymbol{u}_l}{\partial \mathbf{q}_l} \,, \tag{77}$$

where the derivatives with respect to the attitude quaternion should be interpreted componentwise. Differentiating Eq. (11) and using that $\mathrm{d}(\mathsf{C}'\boldsymbol{u}_l) = \mathbf{0}$, we have

$$\{\mathrm{d}(\mathsf{S}'\boldsymbol{u}_l),\, 0\} = 2\{\mathsf{S}'\boldsymbol{u}_l,\, 0\}\mathbf{q}_l^{-1}\mathrm{d}\mathbf{q}_l\,, \tag{78}$$

which after some manipulation gives

$$-\frac{\partial R_l^{\mathrm{AL}}}{\partial \mathbf{q}_l} = -2\sec\zeta_l\,\mathbf{q}_l\left\{\mathsf{S}'\boldsymbol{n}_l, 0\right\}, \quad -\frac{\partial R_l^{\mathrm{AC}}}{\partial \mathbf{q}_l} = 2\mathbf{q}_l\left\{\mathsf{S}'\boldsymbol{m}_l, 0\right\}. \tag{79}$$

The derivatives with respect to the spline coefficients $\mathbf{a}_n$ are obtained after multiplying the above expressions by $B_n(t_l)$, assuming that the normalization factor in Eq. (10) is close to unity (Sect. 5.2.4).

5.2.4. Constraining the attitude updating

Since the attitude is represented by a unit quaternion, its components should at all times satisfy $q_x^2 + q_y^2 + q_z^2 + q_w^2 = 1$. All four components are nevertheless needed, for while the magnitude of any of them can be inferred from the other three components, its sign cannot. The redundancy of the representation manifests itself in that the length of the quaternion cannot be determined from the observations. Indeed, as can be seen from Eqs. (11)–(12), applying an arbitrary non-zero scale factor to the attitude quaternion $\mathbf{q}$ has no effect on the computed instrument angles, and is therefore unobservable. The attitude parameters $\mathbf{a}_n$ are therefore also undefined with respect to a certain scale value. As a consequence, the normal matrix $\boldsymbol{N}_{aa}$ computed from Eq. (76) is singular, and constraints are needed for computing a unique solution.

The normalization in Eq. (10) was introduced to guarantee that the calculated quaternion is always of unit length, although, as we have seen, this is not strictly necessary for some of the subsequent calculations.[12]

Naively, one might expect that the coefficients $\mathbf{a}_n$ could be scaled independently for each B-spline, i.e., that different scale factors could apply to each index n. This is not the case, however. At any time, the (non-normalized) attitude quaternion is a linear combination of M adjacent coefficients $\mathbf{a}_n$, and unless all four coefficients are scaled by exactly the same factor, the result will not be a simple scaling of the quaternion.[13] Applying this argument to every observation time t_l, it is readily seen that the same scaling factor must be used for all the coefficients in any attitude segment without natural or artificial breaks. In principle, therefore, the attitude normal matrix for such a segment has a rank defect of 1 (that is, the rank is one less than the number of attitude parameters), and would only need a single constraint equation to become non-singular.

Numerical experiments, using Singular Value Decomposition (SVD; see, e.g., Golub & van Loan 1996) of the matrix $\boldsymbol{N}_{aa}$ computed from simulated observations over successively longer time intervals, indeed show the expected rank defect of 1 for intervals up to several hundred knots. That is, there is a clear gap (of several orders of magnitude) between the smallest singular value and the second smallest one. For longer time intervals the gap gradually closes and the problem thus becomes ill-posed (Hansen 1998). Thus, any reasonably long time interval will in practice require some form of regularization rather than the application of just a single constraint.

The adopted solution method is a variant of the well known Tikhonov regularization (Hansen 1998). The objective function in Eq. (27) is modified to include a term depending on the deviation of the normalization factor in Eq. (10) from unity for each observation. We write the deviation as

$$D_l \equiv 1 - \left\| \sum_{n=\ell-M+1}^{\ell} \mathbf{a}_n B_n(t_l) \right\|^2 \tag{80}$$

and the modified objective function as

$$Q(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{g}) = \sum_l \left(R_l^2 + \lambda^2 D_l^2\right) W_l\,, \tag{81}$$

where λ is a small but non-zero regularization parameter. We have found that $\lambda = 10^{-3}$ to 10^{-2} gives a solution that is always numerically stable, and quite insensitive to the precise value of λ. As a result, the normal equations (76) become

$$\sum_{m=n-M+1}^{n+M-1} \left[\sum_{l\,\in\,L_n\cap L_m} \left(\frac{\partial R_l}{\partial \mathbf{a}_n}\frac{\partial R_l}{\partial \mathbf{a}_m'} + \lambda^2 \frac{\partial D_l}{\partial \mathbf{a}_n}\frac{\partial D_l}{\partial \mathbf{a}_m'} \right) W_l \right] \boldsymbol{d}_m = -\sum_{l\,\in\,L_n} \left(\frac{\partial R_l}{\partial \mathbf{a}_n} R_l + \lambda^2 \frac{\partial D_l}{\partial \mathbf{a}_n} D_l \right) W_l\,. \tag{82}$$

The required partial derivatives, obtained from Eq. (80), are

$$-\frac{\partial D_l}{\partial \mathbf{a}_n} = 2 \sum_{m=\ell-M+1}^{\ell} \mathbf{a}_m B_m(t_l) B_n(t_l) \simeq 2\mathbf{q}(t_l) B_n(t_l)\,, \tag{83}$$

where the approximation makes use of the fact that the normalization factor in Eq. (10) is close to unity.

5.2.5. Estimating the excess attitude noise

The excess attitude noise $\epsilon_a(t)$ introduced in Eq. (28) accounts for modelling errors in the attitude representation. Such errors could be caused for example by (unmodelled) micrometeoroid impacts, ‘clanks’ due to sudden redistributions of satellite inertia, propellant sloshing, thruster noise, or mechanical vibrations (Appendix D.4). Due to the cubic spline representation, any localized effect that cannot be fitted by the spline will result in systematic residuals that span over a few consecutive knot intervals. Indeed, discontinuities in the rate (e.g., from micrometeoroid impacts) or angle (e.g., from clanks) produce characteristic patterns of residuals that can be used to identify such events. A significant effort will be devoted to the possibly semi-manual and interactive process of finding these events. When identified, they can be handled for example by modifying the knot sequence (Sect. 5.2.6). But even after this process, the model will be imperfect due to for example high-frequency thruster noise.[14] Similarly, there will be a large number of impacts that are too small to be individually recognized; collectively they add some unmodelled attitude errors, which $\epsilon_a(t)$ may account for. However, it should be noted that $\epsilon_a(t)$ does *not* include any

[12] On the other hand we have implicitly assumed $\|\mathbf{q}\| = 1$, for example in deriving Eq. (79).

[13] Note that the normalization in Eq. (10) is effected by applying a normalization factor that is a continuous function of time. Since any given spline coefficient $\mathbf{a}_n$ is used in a finite time interval (namely, the support of the corresponding B-spline), one cannot obtain a correct normalization for all t by scaling the spline coefficients by some value depending on n.

[14] ‘High-frequency’ here means roughly the range $1/2\Delta\tau \simeq 0.03$ Hz to 0.2 Hz, where $\Delta\tau$ is the typical spline knot interval (∼15 s); lower frequencies are absorbed by the spline and higher frequencies are smoothed out by the integration across the CCD.

component of the observation noise (principally from CCD photon noise), nor is it an estimation of the attitude uncertainty (cf. Sect. 6.3).

Three components of the excess noise, designated $\epsilon_{\rm AL}(t)$, $\epsilon_{\rm ACP}(t)$, and $\epsilon_{\rm ACF}(t)$, need to be derived independently of each other, representing modelling errors in the AL attitude, the AC attitude of the preceding field of view, and the AC attitude in the following field of view. The three components are statistically nearly independent thanks to the way the attitude measurements are made, and the fact that the basic angle is not far from 90°.

The algorithm to estimate $\epsilon_a(t)$ (for a = AL, ACP or ACF) is based on a simple statistical processing of the residuals R_l derived in the source updating. The time line t is divided into 'buckets' $[t_j,\, t_{j+1})$ such that each bucket (j) will contain a sufficient number of observations, also in the AC direction. The size (duration) of a bucket should be several knot intervals for the attitude spline, but the boundaries t_j need not in any other way be related to the attitude knot sequence. One set of buckets is needed for each attitude component (AL, ACP, ACF). Let $l \in ja$ signify that observation l belongs to bucket j and attitude component a. After having completed the source updating for all primary sources, the excess attitude noise in bucket j is estimated as

$$\epsilon_a^2(t_j \le t < t_{j+1}) = \max\left(0,\ \underset{l\in ja}{F_{0.68}}(R_l^2 - \sigma_l^2 - \epsilon_i^2)\right), \tag{84}$$

where $F_{0.68}()$ is the 68% quantile (68th percentile) of the argument values. It is important to note that the downweighting factors w_l determined during the source updating are not used here to eliminate possible outliers; this function is instead taken care of by using the quantile to compute a robust estimate of the typical excess variance in the attitude bucket. This means that if the 'outliers' detected by the source update were actually caused by a stretch of bad attitude, then this will be recognized by a large value of the quantile in Eq. (84), and consequently by an increased ϵ_a^2.

In the subsequent attitude update, the downweighting factors w_l are re-computed based on the residual from the previous source update but with a value for the total variance, $\sigma_\ell^2 + \sigma_i^2 + \epsilon_a^2$, using the newly estimated ϵ_a^2. Thus, only the 'true' outliers – that are not due to the bad attitude – are now downweighted. The data may thus contribute to the attitude updating even if they had been flagged as outliers in the preceding source updating.

The functions $\epsilon_a(t)$ are obviously an extremely useful diagnostic for the progress of the AGIS iterations as well as (after convergence) for the quality of the attitude modelling and data. They can be plotted as a function of time, and the quantity of data is such that human inspection is feasible. They are also needed for setting the detection threshold for micrometeoroid impacts.

The accumulation of statistics in the buckets is best done in parallel with the source updating, when the residuals are readily at hand. One remaining problem is how to compute the quantile in Eq. (84) in an efficient way, without having to store billions of residuals. Indeed, exact calculation of quantiles would require to store all the values $R_l^2 - \sigma_l^2 - \epsilon_i^2$ in a bucket before the quantile can be computed. However, if we are content with an approximate estimate of the quantile, there are a number of sequential estimation algorithms available that only need to store a much smaller amount of data per bucket, see for example Greenwald & Khanna (2001), Gilbert et al. (2002) and references therein. We have chosen to use the Incremental Quantile estimation algorithm due to Chambers et al. (2006).

5.2.6. Initialization of the attitude parameters

An approximate estimate of the attitude is already provided by the initial data treatment (IDT) preceding the astrometric solution. This may be given as a discrete time series, for example one quaternion every second of time. The first time the attitude update is executed for a certain time interval, a regular knot sequence is set up and the B-spline coefficients $\mathbf{a}_n$ in Eq. (10) are determined by a least-squares fit. For a given time series of attitude estimates, this is a linear problem and therefore easily solved. The resulting initial attitude $\boldsymbol{a}^{(0)}$ is used in the first source update (S) and subsequently improved by the attitude update process (A) as part of the AGIS iteration scheme.

By default, a regular knot sequence is adopted, i.e., the knot interval $\Delta\tau = \tau_{n+1} - \tau_n$ is taken to be more or less constant. Given the endpoints $t_{\rm beg}$, $t_{\rm end}$ of a data segment, the knots are set up at regular intervals respecting a given maximum value of $\Delta\tau$ (of the order of 5 to 20 s). The assignment of knots must also take into account the need for anterior and posterior knots, as discussed in Appendix B, and in the case of segmented data, the overlapping knots as discussed in Sect. 5.2.2.

Occasionally the knot sequence needs to be redefined as a result of the adjustment process. Possible causes could be:

- If the spline is not flexible enough to accurately model the data, it may be necessary to decrease the maximum allowed $\Delta\tau$.
- Conversely, overfitting of the data may require the maximum allowed $\Delta\tau$ to be increased.
- Locally, a scarcity of accurate data or a short gap could make it necessary to remove some knots or introduce a natural break in the attitude representation (Sect. 5.2.2).
- Very locally, a bad fit may result from a micrometeoroid hit causing an almost instantaneous change in the angular velocity of the satellite. This may be dealt with by introducing multiple knots at the appropriate instants (Appendix D.4 and B.3).

Having redefined the knot sequence, it is necessary to reinitialize the spline coefficients $\mathbf{a}_n$, which must now refer to the new knot sequence. This is most easily done by evaluating $\mathbf{q}(t)$ for a regular time series, with a sampling interval much smaller than $\Delta\tau$ (e.g., 1 s), and fitting the new spline to the time series.

5.3. Calibration updating (C)

The geometric instrument model is given in Sect. 3.4. We assume here the generic calibration model in Eqs. (20)–(21), in which the parameters are indexed by rs. In the calibration update block (C) the calibration parameters $\boldsymbol{c}$ are improved by solving the third sub-system in Eq. (38), i.e., the normal equations

$$\left[\sum_l \frac{\partial R_l}{\partial \boldsymbol{c}} \frac{\partial R_l}{\partial \boldsymbol{c}'} W_l\right] \boldsymbol{d}_c = -\sum_l \frac{\partial R_l}{\partial \boldsymbol{c}} R_l(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{c}, \boldsymbol{g}) W_l\,. \tag{85}$$

The residuals in the right-hand side are computed from the parameters values in the current or preceding iteration according to the discussion in Sect. 4.6. Because the calibration model is linear, the partial derivatives are uniquely determined by the observation index l,

$$\frac{\partial R_l}{\partial c_{rs}} = \frac{\partial \eta_l^{\rm obs}}{\partial c_{rs}} = \begin{cases} \Phi_{rs}(l) & \text{if } l \in rs, \\ 0 & \text{otherwise.} \end{cases} \tag{86}$$

In the normal matrix, the element with subscripts $(rs)_1$ and $(rs)_2$ is non-zero only if there is at least one observation l such that

$l \in (rs)_1$ and $l \in (rs)_2$. Depending on how the calibration parameters are grouped into sets with no common observations (for example according to the CCD/gate combination; cf. Sect. 4.3), the normal matrix will therefore be block-diagonal, which the calibration updating takes advantage of in order to save computations. It also facilitates distributed processing.

Since the weights W_l are fixed from the preceding source and attitude updating processes, the update $\boldsymbol{d}_c$ can be calculated in a single direct solution, using the robust Cholesky decomposition (Appendix C). However, due to the degeneracy between for example the large-scale and small-scale AL calibration parameters, this will produce an arbitrary feasible solution $\tilde{\boldsymbol{d}}_c$, which does not necessarily satisfy the constraints in Eq. (22). The constrained update is obtained as

$$\boldsymbol{d}_c = \tilde{\boldsymbol{d}}_c - \boldsymbol{C}(\boldsymbol{C}'\boldsymbol{C})^{-1}\boldsymbol{C}'\tilde{\boldsymbol{d}}_c\,, \tag{87}$$

whereupon the updated $\boldsymbol{c}$ can be computed.

The above-mentioned degeneracy among the calibration parameters means that the normal matrix calculated according to Eq. (85) is singular, which seems to contradict our assumption (Sect. 4.3) that $\boldsymbol{N}_{cc}$ is positive-definite. However, if $\boldsymbol{C}$ spans the null space of $\boldsymbol{N}_{cc}$, as it should for a properly formulated set of constraints, then it can be seen that Eq. (87) gives the same result as solving the updates with the modified normal matrix $\boldsymbol{N}_{cc} + \lambda^2\boldsymbol{C}\boldsymbol{C}'$, which is positive definite for any $\lambda \neq 0$. Thus, the procedure outlined above is equivalent to solving the constrained system with positive-definite matrix.

5.4. Global updating

An arbitrary number of global parameters may be solved for in the AGIS system. Global parameters should be defined in such a way that their default values, equal to zero, correspond to the baseline solution. By not solving for the globals, we implicitly set them to zero, resulting in the baseline solution. For example, we have a very high confidence in General Relativity, which in the parametrized post-Newtonian (PPN) formalism implies the parameter $\gamma = 1$. The global parameter related to the gravitational deflection of light should therefore not be γ itself, but for example the parameter g_0 in

$$\gamma = 1 + g_0\,. \tag{88}$$

That is, $g_0 = 0$ corresponds to the baseline case of General Relativity. The global parameter vector is $\boldsymbol{g} = (g_0,\ g_1,\ \dots)'$.

The normal equations for the update $\boldsymbol{d}_g$ to the global parameter vector are

$$\left[\sum_l \frac{\partial R_l}{\partial \boldsymbol{g}}\frac{\partial R_l}{\partial \boldsymbol{g}'}W_l\right]\boldsymbol{d}_g = -\sum_l \frac{\partial R_l}{\partial \boldsymbol{g}}R_l(\boldsymbol{s},\boldsymbol{a},\boldsymbol{c},\boldsymbol{g})W_l\,, \tag{89}$$

where the sums are taken over all the observations l, and the statistical weights W_l follow from the preceding source and attitude updates. The partial derivatives in Eq. (89) are computed in exact analogy with Eqs. (71)–(72) for the source updating, only with $\boldsymbol{g}$ replacing $\boldsymbol{s}_i$. The calculation of $\partial\boldsymbol{u}_l/\partial\boldsymbol{g}'$ is not detailed here.

In the simple iteration scheme (Sect. 4.5), the inclusion of g_0 representing the PPN parameter γ considerably slows down the convergence of the astrometric solution. As explained by Hobbs et al. (2010), this behaviour is caused by the relatively strong correlation between the gravitational light deflection by the Sun (proportional to $1+\gamma$, and directed away from the Sun) and trigonometric parallax (directed towards the solar-system barycentre, never far from the Sun). Hobbs et al. (2010) found that the convergence rate could be restored by the introduction of a pseudo-parameter g_1 representing a global shift of all parallaxes. (The update to this parameter is solved in each iteration but never applied – its value remains at zero and the converged values of all the other parameters are unaffected; hence the prefix 'pseudo'.) It was later found that this artefact is not needed when using the conjugate gradients scheme, which gives roughly the same rate of convergence whether or not g_1 is included.

6. Auxiliary processes

In this section we describe some auxiliary processes that are not necessarily part of the astrometric solution as such, but nevertheless needed in order to construct the astrometric catalogue. They concern the definition of the reference system for the source positions and proper motions by means of the frame rotator (Sect. 6.1), the selection of primary sources (Sect. 6.2), and the computation of the standard uncertainties and correlations of the astrometric parameters (Sect. 6.3).

6.1. Frame rotator

As explained in Sect. 4.4, the measurement principle of Gaia results in a system of positions and proper motions that is essentially undefined with respect to an arbitrary (small) offset in the orientation and spin of the reference frame. As a consequence, the normal matrix $\boldsymbol{N}$ is in principle singular with a rank defect of 6.

While the solution of rank-deficient problems in general requires special attention to the singularities, for example by adding constraints to avoid numerical instability, no such complication arises here because of the way AGIS works. Basically, a solution is found by iterating between the source and attitude updatings (the calibration and global updatings play no role here because they are to first order independent of the reference frame). When the sources are updated, the reference frame is in reality set by the (then assumed) attitude; similarly, when the attitude is updated, the frame is set by the (then assumed) source parameters. In terms of the matrix formulation of Sect. 4.5 this is equivalent to the statement that the preconditioner $\boldsymbol{K}$ is nonsingular. The end result is that AGIS converges to a solution with both the source and attitude parameters expressed in the same, but largely arbitrary, reference frame.

The intention is however that the final source parameters (positions and proper motions) shall be expressed in a celestial reference frame that represents, as closely as possible, the International Celestial Reference System (ICRS). For consistency, it is moreover necessary that the attitude parameters are expressed in exactly the same frame as the source parameters. It is the task of the frame rotator to accomplish this. A similar process was used to align the Hipparcos Catalogue with the extragalactic reference frame (Lindegren & Kovalevsky 1995).

In the following we start with the rigorous definition of the rotation correction, then derive a linear approximation applicable to the small corrections that we have in practice. Finally, we discuss the determination of the rotation parameters and their application in the AGIS iteration scheme.

6.1.1. Relation between the ICRS and AGIS frames

Ideally, the astrometric solution should result in parameters that are expressed in the BCRS (Sect. 3.1), whose axes are aligned with the ICRS here represented by $\mathsf{C} = [\boldsymbol{X}\ \boldsymbol{Y}\ \boldsymbol{Z}]$. However, due to the in principle undefined reference frame of AGIS, the astrometric solution is in effect expressed relative to a slightly different triad, which we denote $\tilde{\mathsf{C}} = \left[\tilde{\boldsymbol{X}}\ \tilde{\boldsymbol{Y}}\ \tilde{\boldsymbol{Z}}\right]$. The two reference systems, which for simplicity will be referred to as the ICRS and AGIS frames, are related by a time-dependent spatial rotation given by the quaternion $\mathbf{f}(t)$; thus the coordinates of the arbitrary (fixed) vector $\boldsymbol{v}$ are transformed according to the frame rotation formula

$$\{\mathsf{C}'\boldsymbol{v},\ 0\} = \mathbf{f}(t)^{-1}\,\{\tilde{\mathsf{C}}'\boldsymbol{v},\ 0\}\,\mathbf{f}(t) \tag{90}$$

(cf. Eq. A.15). Due to the kinematical constraints of the AGIS solution, $\mathbf{f}(t)$ describes a uniform spin motion of the two frames with respect to each other.

For consistency with Lindegren & Kovalevsky (1995) we parametrize $\mathbf{f}(t)$ by means of two vectors $\boldsymbol{\varepsilon}$ and $\boldsymbol{\omega}$ representing *corrections* to the orientation and spin of the AGIS frame. More precisely, the parameters of the frame rotator are the six coordinates of the vectors in the AGIS frame at some adopted frame rotator epoch $t_{\rm fr}$ (not necessarily the same as the reference epoch $t_{\rm ep}$ of the astrometric parameters). These coordinates are denoted $\varepsilon_{\tilde{X}}$, $\varepsilon_{\tilde{Y}}$, $\varepsilon_{\tilde{Z}}$, $\omega_{\tilde{X}}$, $\omega_{\tilde{Y}}$, and $\omega_{\tilde{Z}}$; according to our kinematical assumption they are strictly constant numbers. For the arbitrary epoch t the frame rotator quaternion is, therefore,

$$\mathbf{f}(t) = \mathbf{Q}[(t-t_{\rm fr})\tilde{\mathsf{C}}'\boldsymbol{\omega}]\mathbf{Q}(\tilde{\mathsf{C}}'\boldsymbol{\varepsilon})\,, \tag{91}$$

where $\mathbf{Q}$ is the function introduced by Eq. (A.12). Equations (90) and (91) provide the basis for the rigorous transformation of any data between the two frames, given the rotation parameters $\tilde{\mathsf{C}}'\boldsymbol{\varepsilon} = [\varepsilon_{\tilde{X}}\ \varepsilon_{\tilde{Y}}\ \varepsilon_{\tilde{Z}}]'$ and $\tilde{\mathsf{C}}'\boldsymbol{\omega} = [\omega_{\tilde{X}}\ \omega_{\tilde{Y}}\ \omega_{\tilde{Z}}]'$.

While the above expressions are strictly valid for arbitrarily large rotation parameters, we have in practice $\|\boldsymbol{\varepsilon}\|$, $\|(t-t_{\rm fr})\boldsymbol{\omega}\| < 20$ mas $\simeq 10^{-7}$ rad, at least in the final iterations of AGIS. This means that second-order terms are completely negligible ($< 0.002\,\mu$as). To first order we have

$$\mathbf{f}(t) \simeq \mathbf{Q}[\tilde{\mathsf{C}}'\boldsymbol{\varepsilon} + (t-t_{\rm fr})\tilde{\mathsf{C}}'\boldsymbol{\omega}]\,, \tag{92}$$

and the vector part of Eq. (90) becomes, to the same approximation,

$$\begin{bmatrix} v_X \\ v_Y \\ v_Z \end{bmatrix} = \begin{bmatrix} v_{\tilde{X}} \\ v_{\tilde{Y}} \\ v_{\tilde{Z}} \end{bmatrix} + \begin{bmatrix} 0 & -v_{\tilde{Z}} & +v_{\tilde{Y}} \\ +v_{\tilde{Z}} & 0 & -v_{\tilde{X}} \\ -v_{\tilde{Y}} & +v_{\tilde{X}} & 0 \end{bmatrix} \begin{bmatrix} \varepsilon_{\tilde{X}} + (t-t_{\rm fr})\,\omega_{\tilde{X}} \\ \varepsilon_{\tilde{Y}} + (t-t_{\rm fr})\,\omega_{\tilde{Y}} \\ \varepsilon_{\tilde{Z}} + (t-t_{\rm fr})\,\omega_{\tilde{Z}} \end{bmatrix}. \tag{93}$$

6.1.2. Transformation of the astrometric parameters

Let $\tilde{\alpha}$, $\tilde{\delta}$, $\tilde{\mu}_{\alpha*}$, $\tilde{\mu}_{\delta}$ be the position and proper motion parameters for a source as derived in AGIS, that is referring to $\tilde{\mathsf{C}}$. For brevity we omit here the source index (i), and do not consider the parallax ϖ_i and radial proper motion μ_{ri} which are independent of the frame orientation. In analogy with Eq. (5) we have the normal triad $[\tilde{\boldsymbol{p}}\ \tilde{\boldsymbol{q}}\ \boldsymbol{r}]$ with respect to the AGIS frame, where $\boldsymbol{r}$ is the barycentric direction to the source at time $t_{\rm ep}$, $\tilde{\boldsymbol{p}} = \langle \tilde{\boldsymbol{X}} \times \boldsymbol{r}\rangle$ and $\tilde{\boldsymbol{q}} = \boldsymbol{r}\times\tilde{\boldsymbol{p}}$; its coordinates in the AGIS frame are given by the columns of the matrix

$$\tilde{\mathsf{C}}'\,[\tilde{\boldsymbol{p}}\ \ \tilde{\boldsymbol{q}}\ \ \boldsymbol{r}] = \begin{bmatrix} -\sin\tilde{\alpha} & -\sin\tilde{\delta}\cos\tilde{\alpha} & \cos\tilde{\delta}\cos\tilde{\alpha} \\ \cos\tilde{\alpha} & -\sin\tilde{\delta}\sin\tilde{\alpha} & \cos\tilde{\delta}\sin\tilde{\alpha} \\ 0 & \cos\tilde{\delta} & \sin\tilde{\delta} \end{bmatrix}. \tag{94}$$

At the source reference epoch $t_{\rm ep}$ the direction cosines of $\boldsymbol{r}$ are related by the frame rotation in Eq. (91); thus

$$\{\mathsf{C}'\boldsymbol{r},\ 0\} = \mathbf{f}(t_{\rm ep})^{-1}\,\{\tilde{\mathsf{C}}'\boldsymbol{r},\ 0\}\,\mathbf{f}(t_{\rm ep})\,. \tag{95}$$

From $\mathsf{C}'\boldsymbol{r} = [r_x\ r_y\ r_z]'$ the position parameters in the ICRS frame are obtained as

$$\alpha = \operatorname{atan2}(r_y,\ r_x)\,, \qquad \delta = \operatorname{atan2}\left(r_z,\ \sqrt{r_x^2 + r_y^2}\right). \tag{96}$$

The transformation of the proper motion components is a bit more complicated, as they are expressed with respect to axes that are physically (slightly) different in the two frames, viz., $\tilde{\boldsymbol{p}}$, $\tilde{\boldsymbol{q}}$ in the AGIS frame, and $\boldsymbol{p}$, $\boldsymbol{q}$ in the ICRS frame. However, the time derivative (at epoch $t_{\rm ep}$) of the barycentric direction to the source is a fixed vector in space, known as the proper motion vector. In a kinematically non-rotating system it can be written

$$\boldsymbol{\mu} = \boldsymbol{p}\mu_{\alpha*} + \boldsymbol{q}\mu_{\delta} = \tilde{\boldsymbol{p}}\tilde{\mu}_{\alpha*} + \tilde{\boldsymbol{q}}\tilde{\mu}_{\delta} - \boldsymbol{\omega}\times\boldsymbol{r}\,, \tag{97}$$

where the last term is the correction for the spin of the AGIS frame. The coordinates of the proper motion vector in the two frames,

$$\mathsf{C}'\boldsymbol{\mu} = \mathsf{C}'\boldsymbol{p}\mu_{\alpha*} + \mathsf{C}'\boldsymbol{q}\mu_{\delta} \tag{98}$$

and

$$\tilde{\mathsf{C}}'\boldsymbol{\mu} = \tilde{\mathsf{C}}'\tilde{\boldsymbol{p}}\tilde{\mu}_{\alpha*} + \tilde{\mathsf{C}}'\tilde{\boldsymbol{q}}\tilde{\mu}_{\delta} - \left(\tilde{\mathsf{C}}'\boldsymbol{\omega}\right)\times\left(\tilde{\mathsf{C}}'\boldsymbol{r}\right) \tag{99}$$

are related by a frame rotation analogous to Eq. (95),

$$\{\mathsf{C}'\boldsymbol{\mu},\ 0\} = \mathbf{f}(t_{\rm ep})^{-1}\,\{\tilde{\mathsf{C}}'\boldsymbol{\mu},\ 0\}\,\mathbf{f}(t_{\rm ep})\,. \tag{100}$$

From Eq. (97) the proper motion components in the ICRS frame are then

$$\mu_{\alpha*} = \boldsymbol{p}'\boldsymbol{\mu} = (\mathsf{C}'\boldsymbol{p})'\,(\mathsf{C}'\boldsymbol{\mu})\,, \quad \mu_{\delta} = \boldsymbol{q}'\boldsymbol{\mu} = (\mathsf{C}'\boldsymbol{q})'\,(\mathsf{C}'\boldsymbol{\mu})\,. \tag{101}$$

For given $(\tilde{\alpha}, \tilde{\delta}, \tilde{\mu}_{\alpha*}, \tilde{\mu}_{\delta})$ and $(\tilde{\mathsf{C}}'\boldsymbol{\varepsilon}, \tilde{\mathsf{C}}'\boldsymbol{\omega})$, the sequence of calculations is therefore:

1. Calculate $\mathbf{f}(t_{\rm ep})$ by Eq. (91);
2. Calculate $\tilde{\mathsf{C}}'[\tilde{\boldsymbol{p}}\ \tilde{\boldsymbol{q}}\ \boldsymbol{r}]$ by Eq. (94);
3. Calculate $\mathsf{C}'\boldsymbol{r}$ by Eq. (95) and $\mathsf{C}'\boldsymbol{\mu}$ by Eq. (100);
4. Calculate α and δ by Eq. (96) and $\mathsf{C}'\boldsymbol{p}$ and $\mathsf{C}'\boldsymbol{q}$ by Eq. (5);
5. Calculate $\mu_{\alpha*}$ and μ_{δ} by Eq. (101).

As we have not employed the approximations in Eqs. (92)–(93), these transformations are rigorous.

6.1.3. Transformation of the attitude parameters

In analogy with Eq. (9) the attitude quaternion $\tilde{\mathbf{q}}(t)$ derived in AGIS defines the transformation from the AGIS frame $\tilde{\mathsf{C}}$ to the SRS S as a function of time; thus for the arbitrary vector $\boldsymbol{v}$

$$\{\mathsf{S}'\boldsymbol{v}, 0\} = \tilde{\mathbf{q}}(t)^{-1}\,\{\tilde{\mathsf{C}}'\boldsymbol{v}, 0\}\,\tilde{\mathbf{q}}(t)\,. \tag{102}$$

Solving $\{\tilde{\mathsf{C}}'\boldsymbol{v}, 0\}$ and inserting into Eq. (90) yields

$$\{\mathsf{C}'\boldsymbol{v}, 0\} = \mathbf{f}(t)^{-1}\tilde{\mathbf{q}}(t)\ \{\mathsf{S}'\boldsymbol{v}, 0\}\ \tilde{\mathbf{q}}(t)^{-1}\mathbf{f}(t)\,. \tag{103}$$

Comparison with Eq. (9) shows that the corrected attitude is given by

$$\mathbf{q}(t) = \mathbf{f}(t)^{-1}\tilde{\mathbf{q}}(t)\,. \tag{104}$$

In practice the AGIS attitude $\tilde{\mathbf{q}}(t)$ is expressed in terms of B-splines by means of coefficients $\tilde{\mathbf{a}}_n$ as in Eq. (10). The result of the time-dependent transformation by $\mathbf{f}(t)^{-1}$ in Eq. (104) cannot, in general, be exactly represented by means of B-splines. However, since the transformation is changing extremely slowly in comparison with the duration of the support of each B-spline (~ 1 min), and also the changes of $\mathbf{a}$ from knot to knot are very small, we make a negligible error by transforming the coefficients instead of the attitude quaternion. Thus we use

$$\mathbf{a}_n = \mathbf{f}(\bar{t}_n)^{-1}\tilde{\mathbf{a}}_n \, , \tag{105}$$

where

$$\bar{t}_n = \frac{1}{2}(\tau_n + \tau_{n+M}) \tag{106}$$

is the time half-way through the support of $B_n(t)$.

6.1.4. Determination of the frame rotator parameters

The parameters $\tilde{\mathsf{C}}'\boldsymbol{\varepsilon} = [\varepsilon_{\tilde{X}},\ \varepsilon_{\tilde{Y}},\ \varepsilon_{\tilde{Z}}]'$ and $\tilde{\mathsf{C}}'\boldsymbol{\omega} = [\omega_{\tilde{X}},\ \omega_{\tilde{Y}},\ \omega_{\tilde{Z}}]'$ are determined by a weighted least-squares solution, using as input the differences in positions and proper motions, for a subset of the sources, between the AGIS results and a priori data. Three kinds of sources may be used for this purpose:

- A subset $S_{\rm NR}$ of the primary sources can be assumed to define a kinematically non-rotating celestial frame. Typically this subset will contain some 10^5 to 10^6 quasars and point-like galactic nuclei, mainly identified from ground-based surveys and photometric criteria. This subset effectively determines $\boldsymbol{\omega}$.
- A subset $S_{\rm P}$ of $S_{\rm NR}$ in addition have positions $(\hat{\alpha}, \hat{\delta})$ accurately determined with respect to the ICRS by means that are completely independent of Gaia. Typically it will contain the optical counterparts of extragalactic objects with accurate positions from radio interferometry (VLBI). Due to the cosmological acceleration effect described below it is necessary to assign an epoch $t_{\rm P}$ to each such position. This subset effectively determines $\boldsymbol{\varepsilon}$.
- The third subset $S_{\rm PM}$ consists of primary sources that do not a priori belong to the non-rotating subset, but have positions and/or proper motions that are accurately determined with respect to the ICRS independent of Gaia. This could include radio stars observed by VLBI, or stars whose absolute proper motions have been determined by some other means. The astrometric parameters of a source in this subset are denoted $\hat{\alpha}$, $\hat{\delta}$, $\hat{\mu}_{\alpha*}$, $\hat{\mu}_\delta$ and refer to the epoch $t_{\rm PM}$ (the parallax is irrelevant here, as it is identical in both frames). It is not expected that this subset will contribute very significantly to the determination of $\boldsymbol{\varepsilon}$ and/or $\boldsymbol{\omega}$, but they are included in the discussion below since they may provide important consistency checks.

In the following we derive the appropriate observation equations for the different kinds of sources. The derivation assumes that the frame rotator parameters are numerically small so that the linear approximation in Eq. (93) applies. For the (weighted) least-squares estimation of the frame rotator parameters it is, furthermore, necessary to assign the appropriate statistical weights to the observations and to have procedures for identifying and handling outliers. These issues are however not discussed here.

In the frame rotator solution, the six parameters must be complemented by three more parameters a_X, a_Y, a_Z taking into account the acceleration of the solar-system barycentre in a cosmological frame (Bastian 1995; Gwinn et al. 1997; Kopeikin & Makarov 2006). Such an acceleration, by the vector $\boldsymbol{\alpha}$, will cause a systematic 'streaming' (dipole) pattern of the apparent proper motions of extragalactic objects, described by

$$\boldsymbol{\mu}_0 = (\boldsymbol{I} - \boldsymbol{r}\boldsymbol{r}')\boldsymbol{a} \, . \tag{107}$$

Here $\boldsymbol{r}$ is the direction to the source and $\boldsymbol{a} = \boldsymbol{\alpha}/c$ to first order in c^{-1}, where c is the speed of light. The galactocentric acceleration of the solar-system barycentre by $\|\boldsymbol{\alpha}\| \simeq 2 \times 10^{-10}$ m s^{-2} is expected to produce a proper motion pattern with an amplitude of $\|\boldsymbol{a}\| \simeq 4$ μas yr^{-1}, which Gaia should be able to detect given a sufficient number of quasars among the primary sources.[15] The additional parameters introduced in the frame rotator solution are the components of $\boldsymbol{a}$ in the ICRS, or $[a_X\ a_Y\ a_Z] = \mathsf{C}'\boldsymbol{a}$; they may be expressed in the same unit as the proper motions.

Observation equations for a source in $S_{\rm NR}$. A kinematically non-rotating source should only have an apparent proper motion due to the cosmological acceleration. Equating $\boldsymbol{\mu}$ in (97) with $\boldsymbol{\mu}_0$ from Eq. (107) and taking the scalar products with $\tilde{\boldsymbol{p}}$ and $\tilde{\boldsymbol{q}}$ results in the two observation equations

$$\left.\begin{aligned} \tilde{\boldsymbol{p}}'\boldsymbol{a} + \tilde{\boldsymbol{q}}'\omega &= \tilde{\mu}_{\alpha*} \\ \tilde{\boldsymbol{q}}'\boldsymbol{a} - \tilde{\boldsymbol{p}}'\omega &= \tilde{\mu}_\delta \end{aligned}\right\} \, , \tag{108}$$

where we have used the scalar triple product rule[16] for the terms including $\boldsymbol{\omega}$. These equations are linear in the unknown acceleration and spin parameters, and the coefficients are the known coordinates of $\tilde{\boldsymbol{p}}$ and $\tilde{\boldsymbol{q}}$ in either $\tilde{\mathsf{C}}$ or C (to the adopted approximation the coefficients are the same in the two frames).

Observation equations for a source in $S_{\rm P}$. In order to compare positions it is necessary to choose an epoch at which to make the comparison. At the chosen epoch of comparison, t, the barycentric coordinate direction of the source is, to first order in the proper motion,

$$\begin{aligned} \bar{\boldsymbol{u}}_{\rm B}(t) &= \boldsymbol{r}(t_{\rm P}) + (t - t_{\rm P})\boldsymbol{\mu}_0 \\ &= \boldsymbol{r}(t_{\rm ep}) + (t - t_{\rm ep})\,(\tilde{\boldsymbol{p}}\tilde{\mu}_{\alpha*} + \tilde{\boldsymbol{q}}\tilde{\mu}_\delta - \boldsymbol{\omega} \times \boldsymbol{r}) \, . \end{aligned} \tag{109}$$

In the first equality we have made the assumption that the source has the apparent proper motion $\boldsymbol{\mu}_0$ when observed in the ICRS frame; the second uses the same proper motion vector derived from the AGIS data according to Eq. (97). If we now compute the coordinates of $\bar{\boldsymbol{u}}_{\rm B}(t)$ in C (using the first equality) and $\tilde{\mathsf{C}}$ (using the second equality), they must be related according to Eq. (93). Resolving the coordinate differences along α and δ we obtain the observation equations

$$\left.\begin{aligned} (t - t_{\rm P})\tilde{\boldsymbol{p}}'\boldsymbol{a} + \tilde{\boldsymbol{q}}'(\boldsymbol{\varepsilon} + (t - t_{\rm fr})\boldsymbol{\omega}) &= \Delta\alpha^* \\ (t - t_{\rm P})\tilde{\boldsymbol{q}}'\boldsymbol{a} - \tilde{\boldsymbol{p}}'(\boldsymbol{\varepsilon} + (t - t_{\rm fr})\boldsymbol{\omega}) &= \Delta\delta \end{aligned}\right\} \, , \tag{110}$$

where

$$\begin{bmatrix} \Delta\alpha^* \\ \Delta\delta \end{bmatrix} = \begin{bmatrix} \tilde{p}_{\tilde{X}} & \tilde{p}_{\tilde{Y}} & \tilde{p}_{\tilde{Z}} \\ \tilde{q}_{\tilde{X}} & \tilde{q}_{\tilde{Y}} & \tilde{q}_{\tilde{Z}} \end{bmatrix} \begin{bmatrix} \cos\tilde{\delta}\cos\tilde{\alpha} - \cos\hat{\delta}\cos\hat{\alpha} \\ \cos\tilde{\delta}\sin\tilde{\alpha} - \cos\hat{\delta}\sin\hat{\alpha} \\ \sin\tilde{\delta} - \sin\hat{\delta} \end{bmatrix} + (t - t_{\rm ep}) \begin{bmatrix} \tilde{\mu}_{\alpha*} \\ \tilde{\mu}_\delta \end{bmatrix} . \tag{111}$$

The observation equations in proper motion are of course the same as in Eq. (108).

Returning to the choice of comparison epoch t, it is clear that the result in terms of the estimated frame rotator parameters should in principle not depend on this choice. However, that will only be the case if the statistical correlations among the data are

[15] The expected acceleration due to, for example, the Andromeda galaxy or the Shapley Supercluster (Proust et al. 2006) are at most $\sim 10^{-3}$ of the galactocentric acceleration. On the other hand, nearby massive galactic objects and large-scale deviations of the galactic potential from axisymmetry could conceivably produce a larger deviation in the direction of the total acceleration.

[16] $\boldsymbol{a}'(\boldsymbol{b} \times \boldsymbol{c}) = \boldsymbol{b}'(\boldsymbol{c} \times \boldsymbol{a}) = \boldsymbol{c}'(\boldsymbol{a} \times \boldsymbol{b})$

taken into account in the least-squares estimation. Otherwise one should choose t to minimize these correlations. From Eq. (111) it is seen that the right-hand sides of Eq. (108) and (110) are uncorrelated if $t = t_{\rm ep}$, provided that the position and proper motion errors in AGIS are also uncorrelated (which is generally the case if $t_{\rm ep}$ is appropriately chosen, i.e., close to mid-mission). Consequently we suggest using $t = t_{\rm ep}$.

Observation equations for a source in $S_{\rm PM}$. Here it will be necessary to distinguish two cases depending on how the proper motions have been measured. If $\hat{\mu}_{\alpha*}$, $\hat{\mu}_\delta$ are the proper motion components of a source measured relative to background quasars, then the observation equations are simply

$$\left.\begin{aligned} \tilde{\boldsymbol{p}}'\boldsymbol{a} + \tilde{\boldsymbol{q}}'\boldsymbol{\omega} &= \tilde{\mu}_{\alpha*} - \hat{\mu}_{\alpha*} \\ \tilde{\boldsymbol{q}}'\boldsymbol{a} - \tilde{\boldsymbol{p}}'\boldsymbol{\omega} &= \tilde{\mu}_\delta - \hat{\mu}_\delta \end{aligned}\right\} . \tag{112}$$

As expected, the equations simplify to Eq. (108) in case the measured proper motion is zero. The observation equations obtained from the comparison of positions are the same as in Eq. (110), but with right-hand side

$$\begin{aligned} \begin{bmatrix} \Delta\alpha^* \\ \Delta\delta \end{bmatrix} = \begin{bmatrix} \tilde{p}_{\tilde{X}} & \tilde{p}_{\tilde{Y}} & \tilde{p}_{\tilde{Z}} \\ \tilde{q}_{\tilde{X}} & \tilde{q}_{\tilde{Y}} & \tilde{q}_{\tilde{Z}} \end{bmatrix} \begin{bmatrix} \cos\tilde{\delta}\cos\tilde{\alpha} - \cos\hat{\delta}\cos\hat{\alpha} \\ \cos\tilde{\delta}\sin\tilde{\alpha} - \cos\hat{\delta}\sin\hat{\alpha} \\ \sin\tilde{\delta} - \sin\hat{\delta} \end{bmatrix} \\ + (t - t_{\rm ep}) \begin{bmatrix} \tilde{\mu}_{\alpha*} \\ \tilde{\mu}_\delta \end{bmatrix} - (t - t_{\rm P}) \begin{bmatrix} \hat{\mu}_{\alpha*} \\ \hat{\mu}_\delta \end{bmatrix} . \end{aligned} \tag{113}$$

If, on the other hand, the proper motion is not measured relative to the local extragalactic background, but in a global non-rotating frame, then it already includes a contribution from $\boldsymbol{\mu}_0$ and the appropriate observation equations are obtained by deleting the terms depending on $\boldsymbol{a}$:

$$\left.\begin{aligned} \tilde{\boldsymbol{q}}'\boldsymbol{\omega} &= \tilde{\mu}_{\alpha*} - \hat{\mu}_{\alpha*} \\ -\tilde{\boldsymbol{p}}'\boldsymbol{\omega} &= \tilde{\mu}_\delta - \hat{\mu}_\delta \\ \tilde{\boldsymbol{q}}'(\boldsymbol{\varepsilon} + (t - t_{\rm fr})\boldsymbol{\omega}) &= \Delta\alpha^* \\ -\tilde{\boldsymbol{p}}'(\boldsymbol{\varepsilon} + (t - t_{\rm fr})\boldsymbol{\omega}) &= \Delta\delta \end{aligned}\right\} . \tag{114}$$

6.1.5. The null space vectors

In Sect. 4.4 we introduced the $n \times 6$ matrix $\boldsymbol{V}$ whose columns span the null space of the normal matrix $\boldsymbol{N}$. For completeness we give here the explicit expressions for one such set of null vectors. Any small change in the unknowns $\boldsymbol{x}$, by a linear combination of the columns in $\boldsymbol{V}$, will leave the calculated residuals unchanged. Applying the frame rotator for arbitrary $\boldsymbol{\varepsilon}$ and $\boldsymbol{\omega}$ obviously leaves the residuals unchanged, and we can therefore compute the columns of $\boldsymbol{V}$ as the partial derivatives of $\boldsymbol{x}$ with respect to the six frame rotator parameters. Since we are concerned with infinitesimal changes, the distinction between the AGIS and ICRS frames is no longer necessary. If $\boldsymbol{V}$ is partitioned similarly to $\boldsymbol{x}$ and $\boldsymbol{b}$ in Eq. (31), or $\boldsymbol{V} = [\boldsymbol{V}_s', \boldsymbol{V}_a', \boldsymbol{V}_c', \boldsymbol{V}_g']'$, we find by means of Eq. (114),

$$[\boldsymbol{V}_s]_i = \begin{bmatrix} q_X & q_Y & q_Z & \tau q_X & \tau q_Y & \tau q_Z \\ -p_X & -p_Y & -p_Z & -\tau p_X & -\tau p_Y & -\tau p_Z \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_X & q_Y & q_Z \\ 0 & 0 & 0 & -p_X & -p_Y & -p_Z \end{bmatrix} , \tag{115}$$

where $\tau = t_{\rm ep} - t_{\rm fr}$ and we have omitted the source index i on the matrix elements. The order of the astrometric parameters is $(\alpha*, \delta, \varpi, \mu_{\alpha*}, \mu_\delta)$. From Eqs. (92) and (105) we similarly obtain for the attitude parameters $\mathbf{a}_n = \{a_x, a_y, a_z, a_w\}$,

$$[\boldsymbol{V}_a]_n = \frac{1}{2} \begin{bmatrix} -a_w & -a_z & a_y & -\tau_n a_w & -\tau_n a_z & \tau_n a_y \\ a_z & -a_w & -a_x & \tau_n a_z & -\tau_n a_w & -\tau_n a_x \\ -a_y & a_x & -a_w & -\tau_n a_y & \tau_n a_x & -\tau_n a_w \\ a_x & a_y & a_z & \tau_n a_x & \tau_n a_y & \tau_n a_z \end{bmatrix} , \tag{116}$$

where $\tau_n = \bar{t}_n - t_{\rm fr}$. The calibration and global parameters are not affected by the frame rotator, so $\boldsymbol{V}_c = \boldsymbol{0}$ and $\boldsymbol{V}_g = \boldsymbol{0}$.

Let $\tilde{\boldsymbol{d}}$ be a vector of small changes to the unknowns $\boldsymbol{x}$, as for example the update computed in one of the AGIS iterations. In some situations it is desirable to remove from $\tilde{\boldsymbol{d}}$ its component in the null space, i.e., to project it on the row space of $\boldsymbol{N}$. This will for example ensure that the orientation and spin of the AGIS frame is not, on the average, changed by the update. In principle, we could achieve this by a process analogous to Eq. (87):

$$\boldsymbol{d} = \tilde{\boldsymbol{d}} - \boldsymbol{V}(\boldsymbol{V}'\boldsymbol{V})^{-1}\boldsymbol{V}'\tilde{\boldsymbol{d}} , \tag{117}$$

where $\boldsymbol{d}$ is the projection of the update on the row space of $\boldsymbol{N}$. This is equivalent to solving the unweighted least-squares problem $\boldsymbol{V}\boldsymbol{z} \simeq \tilde{\boldsymbol{d}}$, yielding the orientation and spin components as $\hat{\boldsymbol{z}} = (\boldsymbol{V}'\boldsymbol{V})^{-1}\boldsymbol{V}'\tilde{\boldsymbol{d}}$, followed by the subtraction of the null space component $\boldsymbol{V}\hat{\boldsymbol{z}}$. In practice, this can equivalently be achieved by means of the frame rotator, without the need to compute $\boldsymbol{V}$ explicitly.

6.1.6. Role of the frame rotator in AGIS

The frame rotator process consists of the three steps: (i) determine the frame rotator parameters according to Sect. 6.1.4; (ii) correct the astrometric parameters for all sources according to Sect. 6.1.2; (iii) correct the attitude parameters according to Sect. 6.1.3. In principle, this process only needs to be run after convergence of the AGIS iterations; nevertheless, there may be a case for running it after each AGIS iteration, preventing a progressive deviation from the ICRS in the course of the iterations.

In particular, for the simulation experiments described in Sect. 7 it was found necessary to run the frame rotator after each iteration, in order to be able to compare the results of each iteration with the 'true' astrometric parameters (used as input to the simulations). Without this correction, the differences between the 'estimated' and 'true' parameters for the source positions and proper motions would have been grossly distorted by frame errors originating from the starting values of the attitude parameters. In this case all the primary sources were treated as belonging to the subset $S_{\rm PM}$, for which Eq. (114) is appropriate.

6.2. *Selection of primary sources*

The astrometric core solution does not use all the sources observed by Gaia, but only a subset of them known as the primary sources. The selection of this subset is made iteratively, based on the results of earlier astrometric solutions and other processes such as double-star and variability analysis (not discussed in this paper). The main criterion for a primary source is that its proper direction, at all times when it is observed by Gaia, is adequately modelled by the standard astrometric model outlined in Sect. 3.2. Examples of sources that should be excluded based on this criterion are solar-system objects, short-period astrometric binaries with a significant size of the photocentre orbit, long-period astrometric binaries with a significant curvature of the photocentre orbit, certain unresolved binaries where one or both of the components are variable, and active galactic nuclei (AGNs) with significant variation of their photocentre positions. Variable stars, long-period resolved binaries, eclipsing and spectroscopic binaries need not be excluded a priori from the set of primary sources, although many of them are potentially problematic from an astrometric modelling viewpoint (e.g., a variable star might be part of an unresolved double or multiple star, resulting in a variable photocentre position). On the other hand, partially resolved double/multiple stars and other extended sources will be problematic even if their photocentres strictly adhere to the basic astrometric model, Eq. (3), because technically the determination of the photocentre becomes more difficult and less precise.

For the calibrations there are other requirements on the primary sources, in particular that there are enough of them at various magnitudes and colours, while their sky distribution is less important. Securing a sufficient number of primary sources for the calibrations will tend to include many more primary sources in some areas, such as the galactic plane, resulting in a very non-uniform distribution across the celestial sphere.

6.2.1. The number of sources needed for AGIS

The number of sources required for the Astrometric Global Iterative Solution is driven by the calibration needs of having representative numbers of sources of different magnitudes, and the attitude needs of having a sufficient number of sources in every knot interval.

Let us consider first the requirements for the geometric small-scale calibration of the CCDs (Sect. 3.4). The angular extent of a single CCD in the across-scan direction is about 0.1°. It scans the celestial sphere at a rate of 1° min^{-1} and therefore covers about 2000 deg^2 per week, if both fields of view are counted. Since there are 1966 pixel columns across the CCD, we have the convenient rule of thumb that each pixel column covers 1 deg^2 per week. Thus, if the average density of suitable primary sources is D deg^{-2} and we require a minimum of N observations of such sources per pixel column for its calibration, then the minimum time needed is N/D weeks. For example, with 10^8 primary sources we have $D = 2400$ deg^{-2}, and it is then reasonable that the small-scale calibration can be made, at a resolution of one or a few pixels, in a matter of weeks. However, for the gated observations of bright sources ($G \lesssim 12$ mag), the available numbers are much smaller and it will be necessary to sacrifice resolution in time, number of pixels, or both in order to have a sufficient number of observations per calibration cell. For example, the gate used for the brightest sources will perhaps mainly be used for $G \simeq 5.7$ to 8.8; the *total* number of such stars is about 130 000 (from the Tycho-2 Catalogue; Høg et al. 2000) or $D \simeq 3$ deg^{-2}, of which only a fraction will be suitable as primary sources. Thus, even over the whole five-year mission there will only be a few hundred observations per pixel column. From these considerations it is clear that as many as possible of the bright stars should be selected as primary sources.

Consider next the requirements for the attitude determination. The minimum reasonable knot interval is of the same order as the transit time over a CCD, or about 5 s. Since we require both along-scan and across-scan measurements, and the latter are normally only provided by the SM and some of the AF1 observations, we assume conservatively one observation in each coordinate per field-of-view transit. There are seven CCDs across the width of the field of view; the area scanned is therefore 1000 deg^2 per day in each field of view, or 0.06 deg^2 per 5 s interval. If we require, say, 100 transits per knot interval for a reliable attitude determination, then the minimum density of primary sources is $D = 1700$ deg^{-2}, or some 70 million sources in total. To achieve this density in the galactic pole regions requires that stars as faint as $G \simeq 19$ are included among the primary sources. Thus it is clear that the design aim of $\simeq 10^8$ primary sources is quite reasonable, and that these will have to include both very bright and very faint stars, as well as many quasars down to $G = 20$ for the extragalactic link.

Apart from these minimum requirements, it must be maintained that the quality of the solution will only improve, the more (good) primary sources are included. Stated in the negative sense, the solution quality cannot improve by removing good-quality primary sources. From this viewpoint one should aim to include as many primary sources as possible in the final solution.

On the other hand, one should not forget that it is possible to run AGIS with much fewer primary sources by disabling short-term small-scale calibrations and using longer knot intervals for the attitude. This will increase modelling errors, which however is acceptable for initial runs where the input data have not yet been properly calibrated.

6.2.2. Selection criteria

As outlined above a source has to pass several tests, derived from different processes, in order to qualify as a primary source. The most important test is derived from the AGIS solution itself, and is based on how well the standard astrometric model (Sect. 3.2) fits the data. However, if initially we want to limit the number of primary sources, a somewhat more sophisticated selection procedure is needed to guarantee the minimum requirements.

Each source carries an attribute representing the ‘relegation factor’ U, which is a floating-point number ranging from about 1 to infinity. $U \simeq 1$ implies the source is perfect for use in the astrometric solution, while successively larger values indicate less suitable sources. The relegation factor may incorporate the results of several different tests, and therefore provides a continuous variable for use in the selection process. The name derives from the need to ‘relegate’ a primary source into a secondary one when U exceeds a predefined value, which may happen for example in the course of the AGIS iterations, or from one solution to the next. On the other hand, a secondary source may be promoted to a primary if its U value decreases below the set threshold. This suggests that all potential primary sources should be processed through the source update (Sect. 5.1), after which its status as primary/secondary may be decided.

The excess source noise ϵ_i estimated during the source updating (Sect. 5.1.2) may be a good starting point for calculating the relegation factor, e.g.:

$$U_i = \sqrt{1 + [\epsilon_i/e(G_i)]^2}\,, \tag{118}$$

where $e(G)$ is a normalization factor depending on the magnitude. The choice of the function $e(G)$ determines the balance between absolute and relative contributions to the modelling error budget. With the choice $e(G) \simeq \sigma_l^{\rm AL}$ (the formal along-scan observational standard uncertainty for a source of magnitude G), U_i approximates the RMS normalized residual of the source. Selecting sources based on this U_i tends to discriminate against bright stars where modelling errors may dominate over photon-noise errors. On the other hand, choosing $e(G) = $ constant means that only sources with the smallest ϵ_i are accepted; this may remove too many of the faint sources, where ϵ_i is still small in comparison with $\sigma_l^{\rm AL}$. A reasonable compromise between these two extreme cases should be found. The value from Eq. (118) can later be combined with other factors indicating for example photometric variability, or some other potentially problematic property, so that in general the relegation factor can be determined with some formula using a combination of the source attributes.

Apart from the relegation factor, which indicates whether a source is at all suitable as a primary source, the general principle should be to maximize the total weight of the primary sources. The weight of a source i is defined as

$$W_i = \frac{1}{n_i}\sum_{l\in i}\frac{w_l}{\sigma_l^2+\epsilon_i^2}\,, \tag{119}$$

where the average is taken over the $n_i = \sum_{l\in i} 1$ accepted AL observations of the source. Note that we do not use the more obvious definition $W_i = \sum_{l\in i} W_l$ with W_l from Eq. (62), since we do not want to penalize a source by the excess attitude noise in some of its observations, nor favour a source because it is observed many times due to the scanning law.

Having defined the relegation factor and weight per source, the selection of primary sources can be made to maximize the total weight with due regard to sky uniformity (for attitude determination) and magnitude distribution (for instrument calibration). A possible procedure is the following.

We start by specifying the minimum density $D_{\rm min}$ (deg^{-2}) required for the attitude determination, the targeted total number $N_{\rm tot}$ of primary sources, with $N_{\rm tot} \ge (129600/\pi)D_{\rm min}$, and the maximum acceptable relegation factor $U_{\rm max}$. For the geometric calibration of gated observations we may also specify minimum numbers $\{N_g\}$ for several intervals in G. Then:

1. Using a coarse-grained tessellation of the celestial sphere (see below), select in each pixel the N_p sources with the largest W_i that satisfy $U_i \le U_{\rm max}$, where N_p is the minimum number per pixel that will ensure $D_{\rm min}$.
2. For each magnitude bin with a required minimum number N_g, count the actual number of primary sources already selected; if it is less than N_g add sources with $U_i \le U_{\rm max}$ based on W_i.
3. If the total number after Step 1 and 2 is less than $N_{\rm tot}$, add sources with $U_i \le U_{\rm max}$ based on W_i. If the total number exceeds $N_{\rm tot}$, reduce $U_{\rm max}$ and repeat the process.

If the required number cannot be reached in a particular pixel or magnitude bin, then it is necessary to increase $U_{\rm max}$ locally for that pixel or bin.

For the tessellation in Step 1 in principle any reasonable way to divide up the sphere into cells of approximately the same area could be used, but for statistical operations the HEALPix scheme (Górski et al. 2005) has some advantages (O'Mullane et al. 2001). The cell size is determined by the choice of HEALPix parameter *NSIDE* and is important as it predicts the level of homogeneity over the sphere. A cell area of about 1/3 of the field of view might be close to optimal, and is achieved with *NSIDE* = 128 yielding 196 608 cells.

For the first run with real data some selection must be made using the initial star catalogue. In this case the relegation factor may be set to 1 for all sources and the selection based entirely on their spatial distribution and magnitudes. This will reduce the input to the first run of the astrometric solution, after which the relegation factor will be updated as described above. The secondary-source update step runs on all sources not included in AGIS; hence this will set a relegation factor for all sources observed by Gaia.

6.3. Computation of standard uncertainties and correlations

It is mandatory that the catalogue of astrometric parameters resulting from the astrometric core solution includes complete and reliable information about the expected error statistics. The most important quantity is the estimated standard uncertainty of each astrometric parameter. However, the statistical correlation between the different astrometric parameters – both between the different parameters of the same source and between the parameters of different sources – is also important and should be quantified. Such correlations are produced both by attitude modelling errors (Sect. 5.2.5) and the statistical uncertainty due to the finite astrometric weight of the sources contributing to the attitude determination. More generally, we need a method to estimate the 5×5 covariance matrix $\mathrm{Cov}(\boldsymbol{s}_i, \boldsymbol{s}_j)$ of the astrometric parameters of any two sources i, j (including the case $i = j$). In principle, these are sub-matrices of the upper-left $n_s \times n_s$ part of $\boldsymbol{N}^\dagger$, the pseudo-inverse of the complete normal equations matrix in Eq. (30). (The pseudo-inverse should be used since the matrix is singular.) Although there are methods to compute selected elements in $\boldsymbol{N}^\dagger$ that may be feasible even for a system as large as this, it is utterly impossible to produce any significant fraction of the covariances by a direct computation. Instead, it will be necessary to rely on approximations and statistical estimates. A first approximation is obtained by ignoring the statistical uncertainty contributed by the errors of the attitude, calibration and global parameters; in this case we can ignore all parts of $\boldsymbol{N}$ in Eq. (31) except $\boldsymbol{N}_{ss}$ and find

$$\mathrm{Cov}(\boldsymbol{s}_i, \boldsymbol{s}_j) \equiv \left[(\boldsymbol{N}^\dagger)_{ss}\right]_{ij} \simeq \begin{cases} [\boldsymbol{N}_{ss}]_{ii}^{-1} & \text{if } i = j, \\ \boldsymbol{0} & \text{if } i \ne j, \end{cases} \tag{120}$$

where the inverse of $[\boldsymbol{N}_{ss}]_{ii} = \boldsymbol{A}_i'\boldsymbol{W}_i\boldsymbol{A}_i$ is regularly computed as part of the source updating using Eq. (57). However, in this approximation we clearly cannot estimate the covariance between sources ($i \ne j$), which is unacceptable; moreover, we underestimate the within-source covariance ($i = j$) because of the neglected attitude and calibration errors. Refining this estimate is a very important problem which however is addressed elsewhere (Holl et al. 2010, 2012).

A somewhat related problem is the need to be able to transform the astrometric results, without loss of information, to an arbitrary epoch different from the $t_{\rm ep}$ used in the astrometric solution. The standard model in Eq. (4) allows the astrometric parameters to be transformed in a completely reversible manner, based on the assumption of uniform space motion relative to the solar-system barycentre. In this process it is necessary to include the sixth astrometric parameter μ_{ri}, even if it is (partially) derived from a spectroscopic radial velocity. Similarly, the transformation of the covariance matrix must consider all six parameters. The relevant formulae are given in Vol. 1, Sect. 1.5.5 of The Hipparcos and Tycho Catalogues (ESA 1997) and are not repeated here.[17] Indeed, the normal equations for the six parameters contain the full information of the Gaia observations of a particular source with respect to the standard astrometric model, and for this reason it is desirable to compute and store these normals even if only a subset of them is used in the actual source update (cf. Sect. 5.1.3).

[17] The sixth parameter μ_r is denoted ζ in ESA (1997).

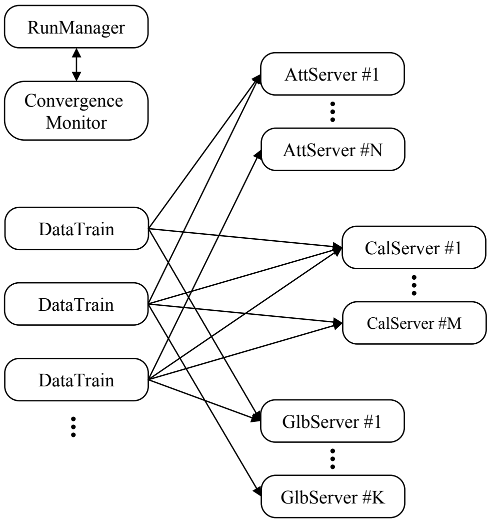


Fig. 8. Simplified architectural design diagram of AGIS. The rounded boxes are independent Java processes running in parallel on different nodes of a multi-CPU processing cluster. The arrows indicate main data exchanges between the various processes. Input/output-related data flow from/to the storage system is not shown, and likewise some important but conceptually irrelevant interactions between some elements (e.g., the *Server*s and the *RunManager*).

7. Software implementation and demonstration solutions

The practical realisation of the AGIS scheme as outlined in the preceding sections is contained in software that is being developed, since early 2006, jointly by teams at the European Space Astronomy Centre (ESAC) and Lund Observatory within the Gaia Data Processing and Analysis Consortium (DPAC). It is a central software module embedded into a complex, overall data processing system (O'Mullane et al. 2007), whose ultimate goal is the creation of the final Gaia catalogue with a targeted release date around 2021.

This section gives a concise overview of the architectural design of the AGIS software (Sect. 7.1) on the one hand, and on the other presents some selected results from a recently (June–November 2011) conducted large-scale astrometric solution (Sect. 7.2) using as input simulated data for more than 2 million sources. While this number is still a factor 50 smaller than the number of primary sources foreseen in the final AGIS runs (around 2018–2020), and the present simulations are much simplified especially with respect to the attitude modelling, we believe that this proof-of-concept run demonstrates the practical validity and correctness of the key theoretical concepts described in this paper. Future tests will involve simulated data sets that are both larger and more realistic, with a parallel further development of the algorithms and software in view of the added new complexities.

7.1. AGIS software overview

The term AGIS is subsequently often used to refer to the actual software implementation of the scheme. Like virtually all Gaia data processing software, AGIS is entirely written in the object-oriented Java programming language (O'Mullane et al. 2010). The implementation has been briefly outlined in Lammers et al. (2009) and more comprehensively in O'Mullane et al. (2011), to which the interested reader is referred for more details. In this section, some key classes are briefly described; following Java naming conventions their names (given in italics) are concatenated capitalized nouns, as in *RunManager*.

Owing to the number of sources and the associated large data volumes that have to be handled (see Sect. 1) it is clear that a well-performing system must be distributable on modern, multi-node, multi-core processing hardware environments and make optimal use of parallelism as far as permitted by the AGIS scheme. Another elementary consideration is that inherently slow disk input-output operations should be minimized and never allowed to be a bottleneck for any of the computing processes.

These basic requirements have led to the system schematically depicted in Fig. 8 with its main components and data flow. The central elements are *DataTrain*s, *Server*s, a *RunManager*, and a *ConvergenceMonitor*. When an AGIS run starts the *RunManager* splits the entire processing task of the first iteration into separate and independent jobs which are then taken and executed in parallel by *DataTrain*s that have been started on the different CPUs of the processing system. Each such job involves the processing of all observations for a group of sources (with typically 100–1000 sources per group), which is done by looping over the sources, one at a time. Before the loop is entered all the data that are needed for the processing (observations, as well as all source, attitude, and calibration data) have been loaded into memory and are passed on to the core algorithms. This is a key design aspect which, together with a suitable grouping of the data on the storage system, ensures that the system is never input/output limited and that it has a constantly high utilization of the CPUs (typically at the 90% level).

Each AGIS iteration starts with the updating of the respective sources (see Sect. 5.1). Computed provisional updates are then written to the storage system and, finally, the observations together with the updated source data are sent to attitude, calibration, and global servers via the CPU-interconnecting network. The servers themselves are distributed in different ways (see, e.g., Sect. 5.2.2 for attitude), but all are similar in that they accumulate normal equations by adding observation equations as outlined in Sects. 5.2.1, 5.3, and 5.4, respectively. When all *DataTrain* jobs are finished, the *RunManager* signals to the servers that all observations have been sent. This triggers that a Cholesky decomposition (Appendix C) is made of the accumulated normals matrices in every server, that the partial results on different servers are combined where necessary (e.g., the attitude segments are joined), and finally that the results are persisted in the storage system. That marks the end of an iteration. Subsequent iterations are then started in the same manner as the first one, viz., through the creation of a set of processing jobs.

Iteration k uses the computed updates from iteration $k-1$ and generates updated parameters for use in the following iteration $k+1$. This progress is monitored by the *ConvergenceMonitor* through the accumulation of a selected list of statistical quantities in the form of graphical plots (e.g., histograms of the updates of all the astrometric parameters) accessible in real time through a web interface. Naively, one may expect that the system can be considered converged if the updates become smaller than a pre-defined limit; however, finding an unambiguous and automatically verifiable convergence criterion has proven to be a surprisingly complex problem (see Sect. 4.4 in Bombrun et al. 2011). We believe now that human inspection is indispensable

Table 1. Characteristics of the simulated input data and demonstration solution.

Quantity	Value
Duration of science mission	5.0 yr
Number of sources	2 256 222
Number of along-scan (AL) observations	1.625×10^9
Number of across-scan (AC) observations	1.805×10^8
Standard uncertainty per AL / AC observation (representative):	
$G \leq 13$	92 μas / 520 μas
$G = 15$	230 μas / 1350 μas
$G = 17$	590 μas / 4000 μas
$G = 18$	960 μas / 7600 μas
$G = 19$	1600 μas / 16000 μas
$G = 20$	2900 μas / 38000 μas
Number of astrometric parameters	1.128×10^7
Number of attitude spline knots	6.575×10^5
Number of attitude parameters	2.630×10^6
Number of calibration parameters	7 812
Number of global parameters	1

to assess the convergence status of the system reliably and, ultimately, decide on the termination of the iterative loop.

An important feature of the *RunManager* is the ability to use different algebraic solution methods by selecting among different available iteration schemes. The description above explains the 'kernel' computation of provisional updates to the unknowns employing the Gauss-Seidel-type preconditioner approximation (Eq. 53 in Sect. 4.6) to the full normal matrix of the system (Sect. 4.5). How these provisional updates are actually combined at the end of an iteration to form the final updates to the unknowns depends on the chosen iteration scheme. AGIS can use all four schemes outlined in Sect. 4.7, viz., the simple iteration (SI), accelerated simple iteration (ASI), conjugate gradients (CG), and hybrid scheme (A/SI-CG). The previous description essentially refers to SI; in the other schemes there are a few extra steps which however are immaterial for understanding the software system.

AGIS is controlled through a list of a few hundred key–value parameters ('properties') which are configured before a run starts. Examples are: the numbers of servers and threads, the size of *DataTrain* jobs, the starting values for the unknowns, and the employed solution method. Also, which update blocks are active during a run is controlled via properties. Any combination that involves at least a source update is possible, e.g., SACG, SA, SCG.

The optimum number of data trains in a run is a complex trade-off between the available number of CPUs and memory, usable network bandwidth (more trains create more inter-CPU traffic), and the given maximum storage system throughput. By design of the system, the run time should scale inversely with the number of data trains (assuming there are enough CPUs), i.e., doubling the number of *DataTrain*s should halve the run time. In the tests done until now, the run times are very satisfactory; however, more work remains for achieving the desired optimal scaling behaviour. A first AGIS run using simulated data for a 5 yr mission with 50 million primary sources was successfully completed in June 2011. This being only a factor 2 less than the baseline 100 million sources envisaged in the final AGIS run towards the end of the mission, it marks an important milestone in the development of the operational system.

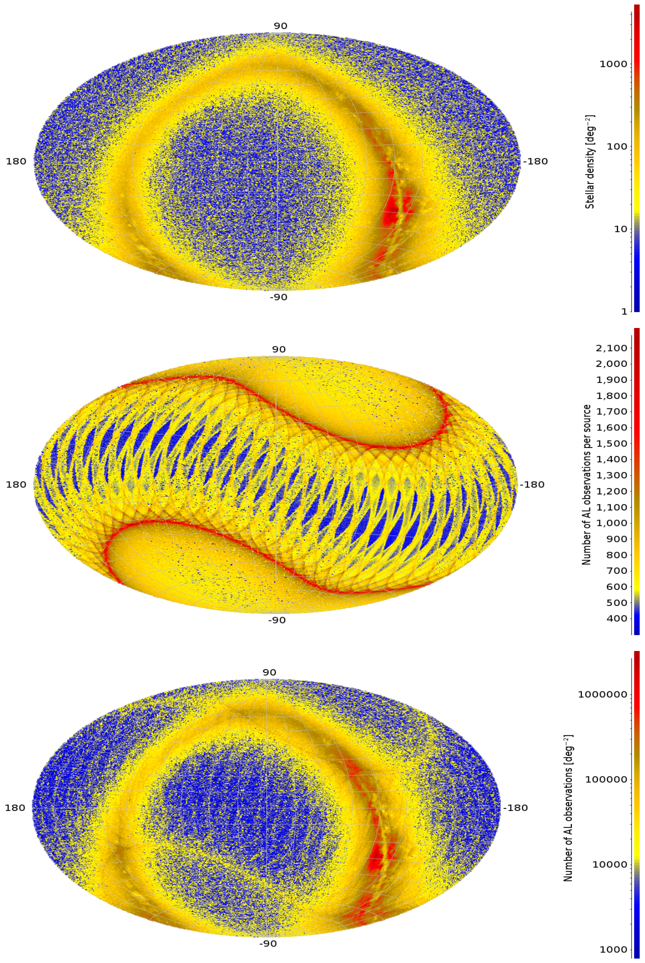


Fig. 9. All-sky projections of (from top to bottom) the total stellar density in the input data to the demonstration solution, the number of AL observations per source, and the resulting spatial density of AL observations. These and all subsequent sky maps use the Hammer–Aitoff equal-area projection in equatorial coordinates, with α running from $-180°$ to $+180°$ right to left. **Top:** The simulated sky contains some 2 million single stars covering the Gaia magnitude range $6 \leq G \leq 20$. The density ranges from less than 1 deg^{-2} around the galactic poles to a maximum of about 4800 deg^{-2} near the galactic centre in the bottom-right quadrant of the map. **Middle:** The number of along-scan observations per source reflects the scanning law of Gaia, which is roughly symmetric around the ecliptic plane and gives an overabundance of observations at ecliptic latitudes ±45°. **Bottom:** The combination of the source density and the scanning law gives the displayed density of along-scan observations.

7.2. Demonstration solution

7.2.1. Data simulation and model assumptions

Since the start of the development in early 2006, AGIS has been tested continuously using simulated datasets of varying complexity and size generated by the Gaia System Simulator (Masana et al. 2005) created by DPAC's dedicated coordination unit for Data Simulations (CU2; Mignard et al. 2008; Luri & Babusiaux 2011). In the following we present the results of a test solution using 5 years of simulated astrometric observations for about 2 million well-behaved (single) stars with a realistic distribution both in magnitude and coordinates, based on

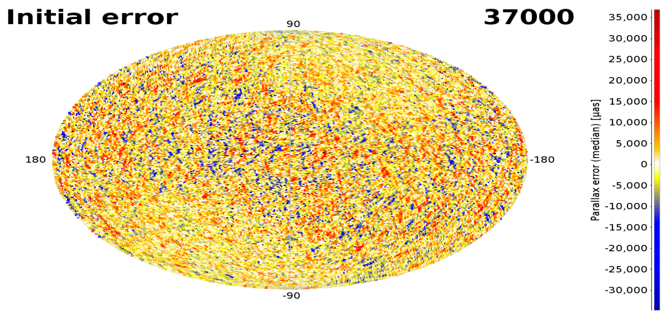


Fig. 10. Map of the parallax errors in iteration 1. The iterative astrometric solution starts off with spatially correlated errors in the astrometric parameters, generated as described in the text. These (initial) parallax errors have amplitudes of a few tens of mas. The number at the top-right corner of this (and subsequent) maps is the maximum value of the displayed range in μas.

the Besançon galaxy model (Robin et al. 2003). Figure 9 (top) shows the spatial source density of the data set in equatorial coordinates. Of particular interest for the AGIS run is the stark density contrast between ~ 1 and 5000 deg^{-2} mainly depending on galactic latitude, resulting in similarly high ratios in the total astrometric weight of the sources in Gaia's two fields of view. In Bombrun et al. (2011) it was shown (using simulations on a smaller scale) that a high weight contrast tends to reduce the convergence rate of the astrometric solution compared to a situation where the weights are more balanced; however, the solution always converges to the correct solution provided that enough CG iterations are carried out. We will show that this key result is confirmed in the demonstration solution.

The input data were generated using a fully-relativistic model of the observed (proper) directions $\boldsymbol{u}_i(t)$ in Eq. (7), including gravitational light deflection for PPN parameter $\gamma = 1$, assuming the Nominal Scanning Law (Sect. 3.3), the nominal geometrical instrument model (Sect. 3.4) and nominal performance of the instrument (in particular the centroiding accuracies $\sigma_l^{\rm AL}$, $\sigma_l^{\rm AC}$ as functions of G; see Table 1). However, in order to test the capability to recover a varying basic angle, a step-wise perturbation was introduced corresponding to the sinusoidal variation of $\Delta\Gamma_j$ in Eq. (19) with a period of 2.5 yr and an (unrealistically large) amplitude of 500 μas. In the astrometric solution, the large-scale AL calibration interval was set to 30 days, matching the step width of the perturbation signal. The solution used only one global parameter, viz., $g_0 = \gamma - 1$ as described in Sect. 5.4. Table 1 lists some statistics of the data and solution, while the middle and bottom maps in Fig. 9 show how the number of observations varies across the sky.

Not all elements of the numerical algorithms described in this paper have as yet been integrated into the running software system which otherwise implements the basic model described in Sect. 3. In particular, the estimation of source and attitude excess noise (Sects. 5.1.2 and 5.2.5) were not activated in the present solution. This was not a problem for the demonstration run, as the applied observation noise is well-behaved, purely Gaussian. Further development of the AGIS software will to a large extent focus on making the solution robust against all kinds of unexpected input data.

The demonstration solution was run with all four update blocks (S, A, C, G) enabled, using starting values for the attitude, calibration and global parameters that were erroneous on the mas level. (As explained in Sect. 4.5 and footnote 9, the results of the subsequent iterations are independent of the initial values for the source parameters, because the updating always starts with the source block.) These initial values were created by adding Gaussian, uncorrelated errors to the true attitude parameters, with a standard deviation of 50 mas, using the nominal calibration parameters (i.e., excluding the sinusoidal modulation of the basic angle), and a value of $g_0 \equiv \gamma - 1 = 0.1$ (Sect. 5.4). An attitude knot interval of 240 s was used in order to have a sufficient number of observations per degree of freedom even at the galactic poles. This interval is short enough that the attitude splines are able to represent the true attitude (i.e., the analytical nominal scanning law) with an RMS error of less than 9 μas. Although this is larger than the modelling errors aimed at in the real data analysis, it is sufficiently small in comparison with the typical attitude estimation errors (≥ 20 μas; see Fig. 11) to have a negligible impact on the overall astrometric accuracy of the present solution.

During the source update in the very first iteration, the initial errors in the attitude, calibration parameters and γ propagate to the sources, creating astrometric errors of a few tens of mas (Fig. 10) that are spatially correlated on a scale comparable to the attitude knot interval ($\sim 4°$). These errors are quite hard to remove in subsequent iterations, but may be representative of the situation encountered by AGIS when processing the real mission data based on a fairly uncertain initial attitude and instrument calibration.

With these starting conditions, 135 iterations were carried out using the conjugate gradients (CG) scheme. A re-initialisation of the CG scheme was made after the first 40 iterations to avoid the development of a much slower convergence phase observed in some previous runs; after this, no further re-initialisation was made. At iteration 135 the typical updates of, for example, the parallaxes and the along-scan attitude were at or below a level of 5×10^{-4} μas (Fig. 11). This is still slightly above the numerical noise floor set by the double-precision arithmetic ($\sim 10^{-16}$ rad or $\sim 10^{-5}$ μas). As discussed by Bombrun et al. (2011), truncating the iterations before the numerical noise floor has been reached implies the presence of spatially correlated 'truncation errors' having an amplitude of a few times the typical updates, or $\sim 10^{-3}$ μas in the present case. We now proceed with a more detailed analysis of the results.

7.2.2. Source results

In this section we focus on the results for the parallax (ϖ), which is arguably the most interesting parameter from an astrophysical viewpoint; moreover, as a scalar quantity independent of epoch and reference frame, its statistics can be summarized compactly and without ambiguity. However, the behaviour of the position and proper motion parameters is qualitatively similar, and some results are given in Table 2 and Fig. 14.

The top diagram in Fig. 11 shows the typical sizes of the errors and updates in parallax versus iteration number, as measured by the Robust Scatter Estimate[18] (RSE). The parallax errors settle relatively quickly (around iteration 25) at a level of 146 μas and remain stable till the end of the run with updates becoming successively smaller, reaching the level of 10^{-3} μas around iteration 120. The actual error distribution is symmetric but strongly non-Gaussian (in fact more like a Laplace distribution) due to

[18] The RSE is defined as 0.390152 times the difference between the 90th and 10th percentiles of the distribution of the variable. For a Gaussian distribution it equals the standard deviation. The RSE is used as a standardized, robust measure of dispersion in CU3.

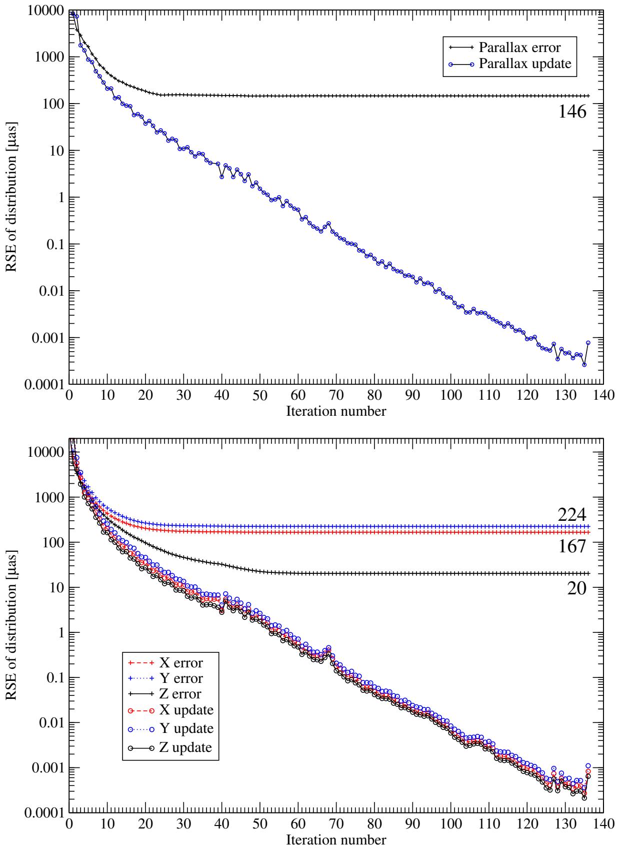


Fig. 11. Top: Evolution of the typical parallax error (crosses) and parallax update (circles) as functions of the iteration number. The typical error settles at around 146 μas. **Bottom:** Evolution of the typical attitude error (crosses) and update (circles) as functions of the iteration number, for the three principal SRS axes x (red), y (blue), and z (black). The errors settle at around 167, 224, and 20 μas, respectively. In both these plots the typical errors and updates are given by the Robust Scatter Estimate (RSE), similar to an RMS value (see footnote 18).

the variation of star numbers and observational standard uncertainties with magnitude (cf. Tables 1 and 2). The overall parallax error RSE of ∼ 150 μas is therefore representative for the median magnitude, $G \simeq 19$, in good agreement with current accuracy predictions (Lindegren 2010). The overall sizes of the errors and updates shown in Fig. 11 should however be complemented by more detailed statistics as functions of coordinate and magnitude.

Figure 12 shows the spatial distribution of the parallax errors at a few selected iterations. The left column shows the median error in each cell, while the right column shows the median absolute value of the error. These quantities serve as robust proxies for the mean and RMS values (the RSE is not used for the latter as many cells have too few sources for this measure), and therefore may suggest the levels of systematic and random errors as function of position. Figure 13 shows the corresponding maps for the parallax updates. After a few iterations, when the overall parallax errors are already below the 1 mas level, very significant systematic (i.e., spatially correlated) errors of a few mas remain, especially in the high-density areas of the galactic plane. These are damped in the subsequent iterations, but still remain at a level of several hundred μas in the galactic centre region around iteration 20, when the overall parallax errors start to settle at their final value according to Fig. 11. At iteration 135 the regional errors have virtually disappeared, and the error map shows a characteristic pattern with the solution being seemingly better around the galactic equator than in the polar regions. This is purely an effect of number statistics: in the galactic pole areas the cells contain rather few sources (often just a single source, in which case the displayed values are simply the individual parallax errors), while closer to the galactic plane the scatter from one cell to the next is reduced by the median-averaging over many sources.

The sequence of error maps for iterations 5, 20, and 135 corroborates a key result of Bombrun et al. (2011), viz., that by iterating long enough the system can cope with large spatial imbalances in the astrometric weights, provided that the minimum source density allows a good attitude determination at all points.

The median absolute values of the parallax errors shown in the right column of Fig. 12 quickly settle in a large-scale pattern that mainly reflects the expected variation of parallax accuracy with ecliptic latitude (see, e.g., Table 3 in Lindegren 2010). This, in turn, depends on the scanning law, i.e., on a combination of the number of observations per source (see Fig. 9, middle) and the geometric configuration of the scans – for example, the overdensity of observations at ±45° ecliptic latitude does little to improve the parallaxes, which are then mainly producing shifts in the AC direction, while Gaia is primarily sensitive to the AL displacement. Number statistics reduce the between-cell scatter of the median absolute values as well, which accounts for the smoother appearance along the galactic equator.

The update maps in Fig. 13 conform with expectations and the preceding discussion. Of particular interest from a diagnostic viewpoint is the observation that the amplitude and spatial distribution of the median updates in the non-converged solution give a fair indication of the (systematic) truncation errors. This is obviously useful for assessing the state of convergence, as the update maps can be constructed for the real mission data as well (whereas the error maps are of course unknown). For example, based on the median updates in iteration 20 (middle left map of Fig. 13) one might correctly conclude that truncation errors of a few hundred μas remain, especially in the galactic centre region, as shown in the middle left map of Fig. 12. By the same reasoning it appears that truncation errors at iteration 135 should be well below 1 μas.

In tests using other datasets with lower density contrasts we have seen a more rapid convergence. A similar slowdown in convergence for a non-uniform weight distribution was observed and discussed in Bombrun et al. (2011) based on small-scale simulations. A likely mechanism for this slowdown is related to the weight contrast problem discussed by van Leeuwen (2005) in connection with the new reduction of the Hipparcos data. Any of the fields of view scanning through a high-density area (or an area with many bright stars) creates a strong astrometric weight imbalance between the two viewing directions, as the other field usually points to an area of the sky with much lower source density (or fainter stars). Errors in the along-scan attitude create correlated errors in the parameters of all sources observed at that time, whether they are in the preceding or following field of view. If these source parameters are then used to correct the attitude, with little counterbalancing effect from the (relatively few) sources in the other field, the attitude may get only marginally improved, with the net effect of slowing down the damping and de-correlation of the errors in the high-density areas. Thus, more

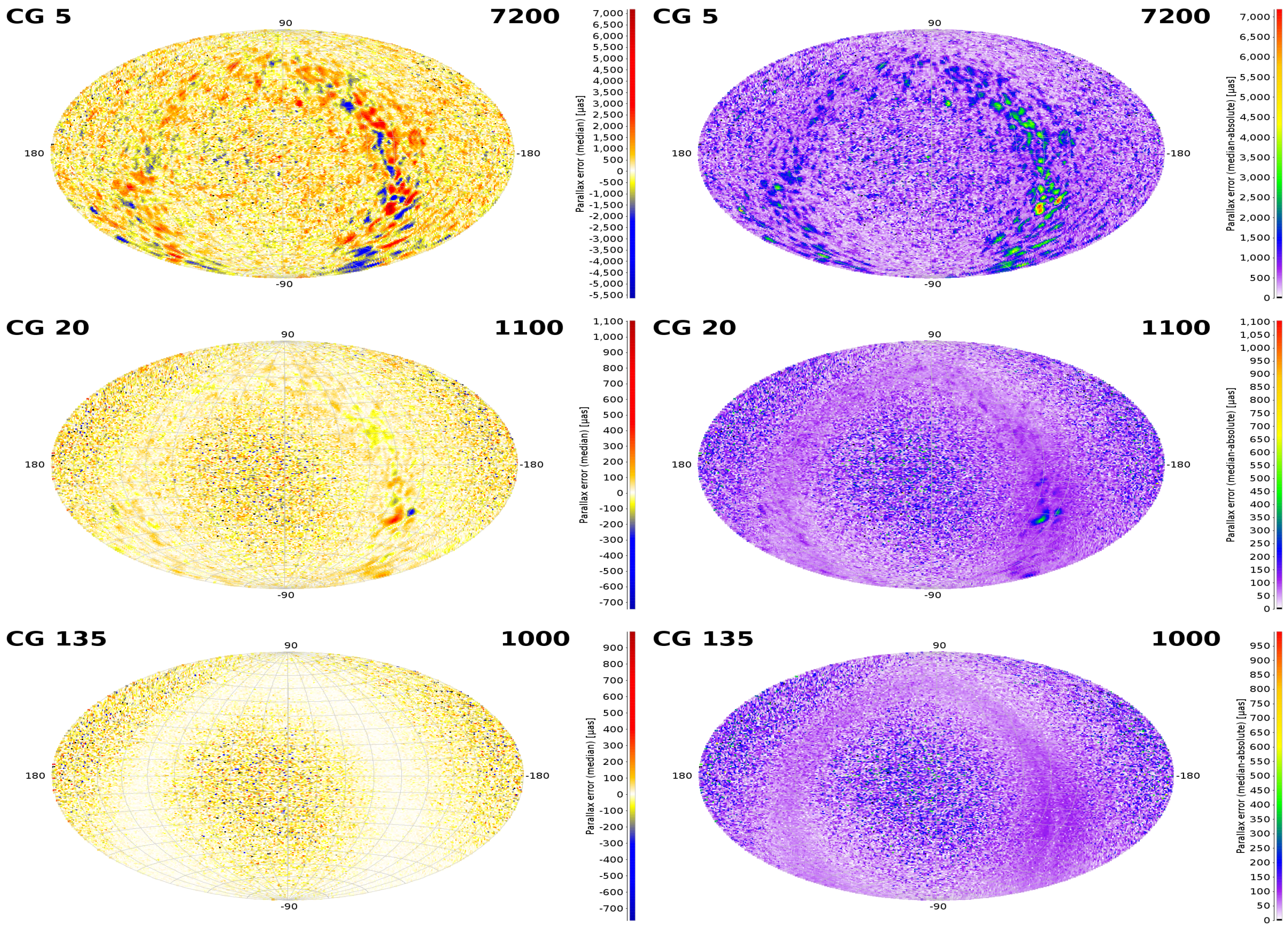

CG 5
7200
90
180
-180
-90
Parallax error (median) [µas]
Parallax error (median-absolute) [µas]
CG 20
1100
CG 135
1000

Fig. 12. Maps (in equatorial coordinates) of the parallax errors in the three selected iterations $k = 5, 20, 135$ (top to bottom). The left column shows the median of the parallax error $\varpi^{(k)} - \varpi^{\text{true}}$, while the right column shows the median of the absolute error $|\varpi^{(k)} - \varpi^{\text{true}}|$; each map cell (of about 0.84 deg^2) shows the colour-coded value computed for the sources located in that cell. Empty cells are shown in black. On every map plot the top left label indicates the iteration number and the top right label is the maximum value of the displayed range in μas. See text for further details.

iterations are needed compared to a more weight-balanced situation.[19]

Figure 14 shows the error and update maps of the proper motions at iteration 135. Since proper motions are vectors, the displayed quantities are the median lengths of the vectorial differences, which are non-negative by definition. The maps are in expected agreement with the corresponding parallax ones concerning visible patterns and structures. A prominent feature absent in the case of the parallaxes is the lighter bands of relatively smaller proper motion errors around ecliptic latitudes ±45° caused by the oversampling of these parallels by the scanning law (cf. Fig. 9). In contrast to the parallax case, these observations do contribute to the determination of the proper motion, especially for the component in ecliptic longitude.

All spatial maps in Figs. 12–14 show median values computed from distributions with stars of all magnitudes. A lot more information is contained in the magnitude-resolved versions of the maps, which are not presented here for brevity. Instead, Table 2 shows the RSEs (see footnote 18) of the errors and normalized errors (see below) of all the astrometric parameters in iteration 135, subdivided according to magnitude. The normalized error is defined as the error (in the case of right ascension, $\Delta\alpha* \equiv \Delta\alpha\cos\delta$) divided by the corresponding formal standard uncertainty obtained from the inverse of the source normal matrix, i.e., by using the approximation in Eq. (120). As discussed in Sect. 6.3 this approximation underestimates the true standard uncertainties of the astrometric parameters by neglecting the contribution from the attitude uncertainty, which may be particularly important for the bright stars where the photon noise is relatively less important.

The RSE of the errors in Table 2 are in reasonable agreement with recent mission accuracy assessments. For example, compared with Eqs. (5.2)–(5.5) in Lindegren (2010) the present values are a few per cent larger for $G < 15$, and up to some 20% smaller for the fainter stars. This suggests a more conservative photon-statistical error budget in Lindegren (2010), combined with a larger-than-nominal contribution from the attitude uncertainty in the present solution. The latter effect, further discussed in Sect. 7.2.3, is indeed to be expected in the present demonstra-

[19] In the new reduction of the Hipparcos data by van Leeuwen (2007) the weight ratio was artificially damped in order to improve the connectivity between the two fields of view in high-contrast situations. We have found that this is not required for Gaia provided that the solution is iterated to convergence; see Bombrun et al. (2011).

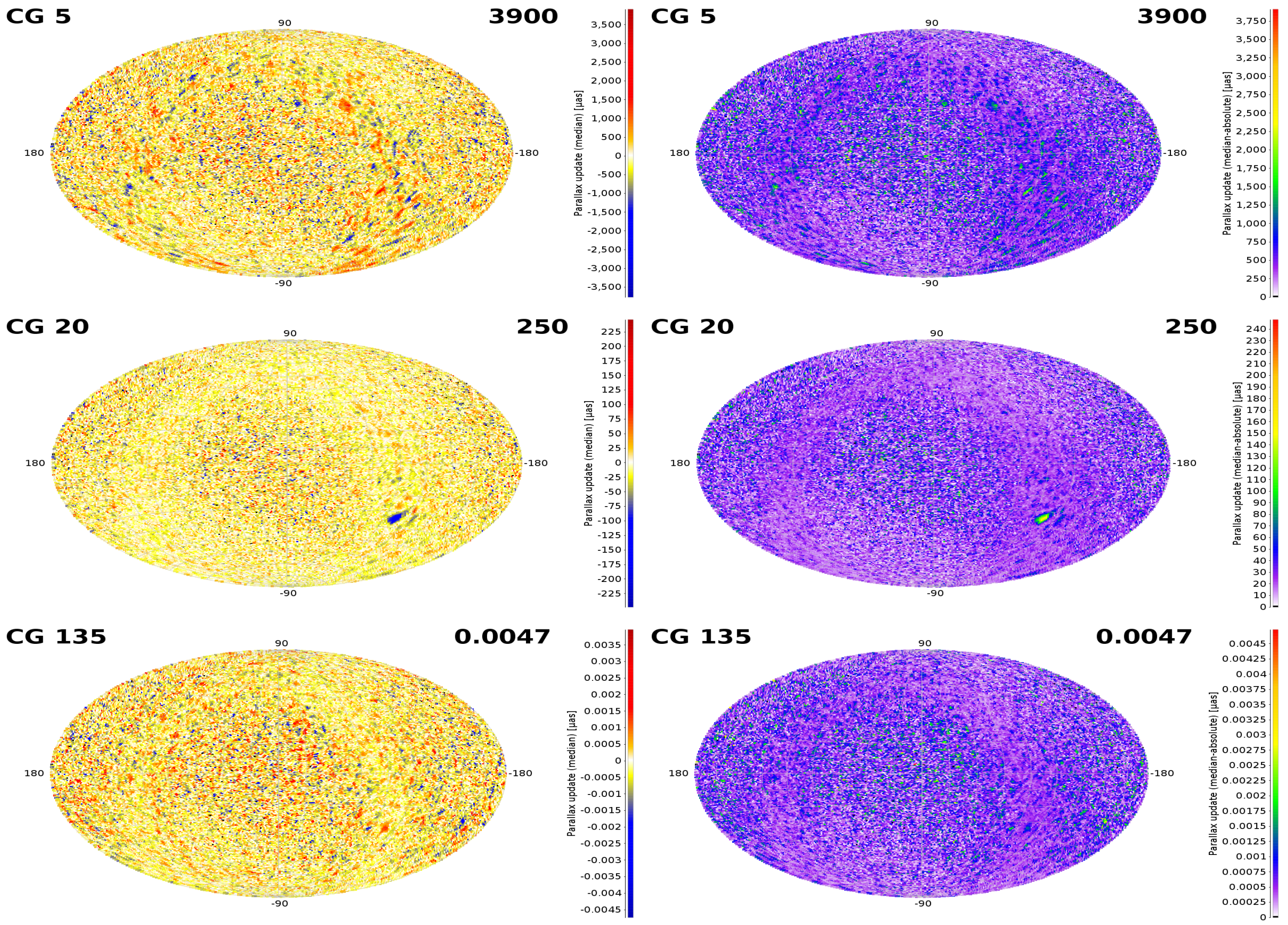


Fig. 13. Same as Fig. 12 but showing the updates in parallax, i.e., the median values of $\varpi^{(k)} - \varpi^{(k-1)}$ (left column) and $|\varpi^{(k)} - \varpi^{(k-1)}|$ (right column).

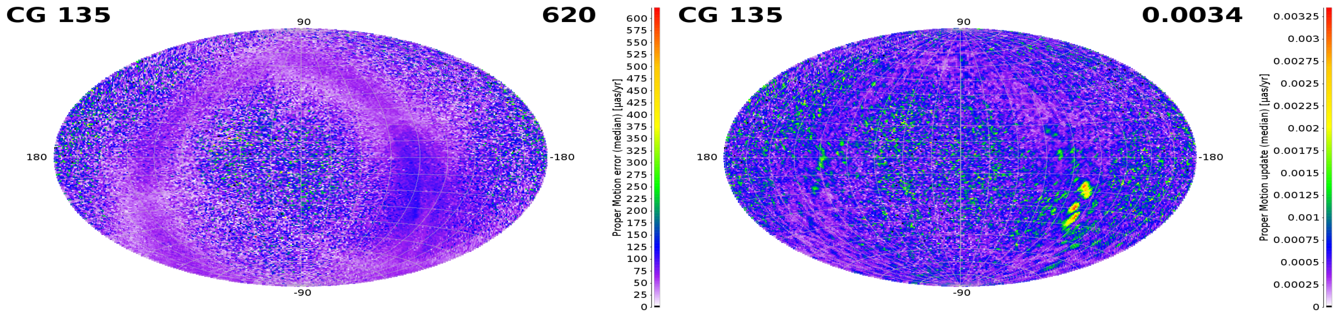


Fig. 14. Maps of the error (left) and update (right) in proper motion for iteration 135. Each cell shows the colour-coded median error/update in units of μas yr^{-1}, where the individual error/update is computed in terms of the equatorial components as $(\Delta\mu_{\alpha*}^2 + \Delta\mu_\delta^2)^{1/2}$.

tion solution using far fewer primary stars than planned for the real mission.

A more stringent test of the quality of the solution is obtained by considering the RSE of the normalized errors (i.e., after division by the formal standard uncertainties as described above), which is here denoted ϱ. Ideally we should have $\varrho \simeq 1$ for any parameter and any magnitude. As shown in Table 2, this is very nearly the case in all parameters for $G > 15$, meaning that the actual errors are roughly consistent with the standard uncertainties computed from Eq. (120). For the brighter stars ϱ becomes progressively larger, supporting the interpretation that the attitude uncertainty has a significant impact on the accuracy of the bright stars in this solution. For example, the value $\varrho = 1.369$ obtained for the parallaxes of stars brighter than $G = 13$ suggests a quadratic attitude contribution to the parallax errors for these stars of $[7.5^2 - (7.5/1.369)^2]^{1/2} = 5.1$ μas, while a similar computation for the next three magnitude bins gives 5.7, 5.1 and 5.4 μas (with a rapidly increasing uncertainty). Thus it appears that the

Table 2. RSE of the errors in the astrometric parameters and of the normalized errors (i.e., after division by the formal standard deviations) for different magnitude ranges. The asterisk in $\alpha*$ indicates that the errors are true arcs on the sky; cf. footnote 11. See text for further explanations.

Magnitude range	No. stars	RSE of error [μas and μas yr^{-1}]					ϱ = RSE of normalized error [unitless]				
		$\alpha*$	δ	ϖ	$\mu_{\alpha*}$	μ_δ	$\alpha*$	δ	ϖ	$\mu_{\alpha*}$	μ_δ
$G < 13$	18 253	6.6	5.7	7.5	4.5	4.0	1.507	1.470	1.369	1.443	1.425
$13 \le G < 15$	70 355	12.4	10.6	14.9	8.7	7.5	1.112	1.100	1.082	1.098	1.100
$15 \le G < 16$	88 116	20.2	17.3	24.9	14.3	12.3	1.024	1.024	1.022	1.032	1.024
$16 \le G < 17$	151 639	30.8	26.7	38.4	21.8	19.0	1.010	1.010	1.010	1.014	1.008
$17 \le G < 18$	272 424	49.4	42.8	61.8	34.8	30.4	1.004	1.006	1.003	1.002	1.003
$18 \le G < 19$	489 253	83.3	70.7	104.1	58.9	50.8	1.001	1.001	1.002	1.003	1.002
$19 \le G$	1 166 182	167.9	140.0	207.6	118.5	100.2	1.001	1.000	1.000	1.001	1.000
all G	2 256 222	116.8	98.7	145.6	82.4	70.6	1.009	1.009	1.007	1.009	1.008

values $\varrho > 1$ found for the parallaxes of the brighter stars can be accounted for by assuming a constant contribution, by about 5–6 μas RMS, to the parallax errors from attitude and/or calibration errors. For the other astrometric parameters we similarly find a constant RMS contribution to the positional errors of about 4–5 μas, and to the proper motion errors of about 3–4 μas yr^{-1}. It will be shown below that these numbers are consistent with the actual attitude errors found in the solution.

It should be noted that the reference epoch for the astrometric parameters was set to exactly half-way into the mission, i.e., t_e = 2014.5 for the simulated mission interval 2012.0–2017.0. This is optimal in the sense that the positional uncertainties are minimized for approximately this epoch, and that the errors in position and proper motion are nearly uncorrelated. A reference epoch half-way through the mission was also assumed for the accuracy assessment in Lindegren (2010), with which the present results have been compared.

In this solution, the median value of the parallax errors of all the 2.2 million sources was not significantly different from zero (the actual value was +0.004 μas). This shows that, in the absence of systematic observational errors, the astrometric solution is able to determine the *absolute* parallaxes of the sources, as could be expected from the basic principles of the mission. It is especially worth noting that this was achieved while simultaneously determining the PPN γ parameter, known to be strongly correlated with the parallaxes (cf. Sect. 5.4).

7.2.3. Attitude results

The bottom diagram in Fig. 11 shows the RSE attitude errors and updates as a functions of the iteration number, where the small differences of the attitude quaternions have been transformed into small rotations along the SRS axes (x, y, z) according to Sect. A.6 and expressed in μas. The error component around the z axis corresponds to the AL attitude error, while the x and y components are linear combinations of the AC attitude errors in the PFoV and FFoV.[20] The z (AL) errors settle at an overall level of 20 μas around iteration 60, while the updates continue to decrease in a similar manner as for the parallaxes (Fig. 11, top). The RSE values of the x and y errors converge to 167 μas and 224 μas, i.e., an order of magnitude larger than in z, reflecting the larger observational errors in the AC direction (Table 1) and the smaller number of AC observations. The ratio of the errors about y and x, $224/167 \simeq 1.34$ is in perfect agreement with the value expected from the geometry of the observations (Fig. 2), viz., $\tan(\Gamma_\mathrm{c}/2) \simeq 1.34$ for a basic angle of $\Gamma_\mathrm{c} = 106.5°$.

The converged AL attitude error of 20 μas is completely consistent with the previously inferred constant RMS contribution, by 5–6 μas, to the parallax errors (see Sect. 7.2.2), as can be seen from the following considerations. For most stars, the propagation of random observational errors from individual AL observations to the parallaxes (say) is largely governed by geometrical factors and the total number of observations per star, and can be statistically described by a 'coefficient of improvement' which can be estimated to $207.6/2300 \simeq 0.09$ using the RSE of the parallax errors for the faintest bin in Table 2 combined with the typical AL observational error at $G \simeq 19.6$ (cf. Table 1). This factor assumes that the individual observational errors (at each CCD) are uncorrelated, which is a very good approximation for the photon-statistical centroiding errors, but not for the attitude errors, which have a correlation length determined by the knot interval of the attitude spline. In the demonstration solution, the knot interval was 240 s, which is much longer than the time it takes an image to cross the nine CCDs in the astrometric field ($\simeq$ 40 s). Therefore it is a much better approximation to assume a *constant* attitude error for the whole field crossing, corresponding to nine AL observations. As a result, the coefficient of improvement relevant for the attitude error should be a factor three larger, or $\simeq 0.27$. The AL attitude uncertainty of 20 μas therefore corresponds to $20 \times 0.27 \simeq 5.4$ μas in the parallax, in very good agreement with the empirical result of 5–6 μas. For the other astrometric parameters a corresponding calculation yields a contribution of about 4 μas in position and 3 μas yr^{-1} in proper motion. Although these numbers are somewhat smaller than the empirical estimates in Sect. 7.2.2 (possibly indicating an additional contribution from the calibration errors), the overall agreement is striking.

The attitude errors obtained in the solution ultimately come from the observational errors of the individual observations, through the process of fitting the attitude spline functions to these observations. If more observations (of the same quality) are added, the attitude errors are expected to diminish inversely with the square root of the number of observations (as long as the modelling errors are not a limiting factor). The present AL attitude error of 20 μas is roughly what can be expected from the density and magnitudes of primary sources in the demonstration solution, as can be seen from the following simple calculation. The AL attitude has essentially one degree of freedom per knot

[20] The attitude errors discussed here must not be confused with the excess attitude noise ϵ_a introduced in Sect. 3.6. The latter represents modelling errors, which are practically absent in the demonstration solution.

interval (240 s). The average number of AL observations per degree of freedom is therefore about 2500 (Table 1). From the magnitude distribution in Table 2 and the AL observational uncertainties in Table 1 one can estimate that the average AL observation carries a statistical weight ($\sigma_{\rm AL}^{-2}$) corresponding to an AL standard uncertainty of $\sigma_{\rm AL} \simeq 650$ μas (taking into account the weight reduction by ϱ^{-2} for the bright stars). The mean flow of observations therefore should allow the AL attitude to be pinned down with an uncertainty of about $650 \times 2500^{-1/2} \simeq 13$ μas. However, this rough calculation assumes uniform distribution of the stars across the sky, with the same mean density (55 deg^{-2}) as in the demonstration run. Considering that large parts of the sky have a much lower density (typically 5–10 deg^{-2}; see Fig. 9, top), which implies a less precise attitude at the corresponding times, and that we have also neglected the attitude modelling errors, which here amount to at most 9 μas RMS (Sect. 7.2.1), it is not unreasonable that the overall AL attitude uncertainty is about 50% larger than according to the above calculation.

The demonstration run uses only 2% of the stars envisaged for the final AGIS solution, and the majority of them are faint, whereas the real primary stars are preferentially selected among the brighter stars when possible (cf. the discussion in Sect. 6.2.2). For a model distribution of 10^8 primary sources similar to the one described by Hobbs et al. (2010), the AL observational uncertainty corresponding to the average statistical weight is more like 200 μas, rather than the 650 μas in the present data. On the other hand, the attitude knot interval will also be much shorter than the 240 s used in the present run, perhaps even as short as 5 s, which is about the shortest knot interval that can reasonably be used in view of the normal CCD integration time of 4.42 s. Combining these numbers we estimate that the final AGIS run on the real Gaia data might obtain an AL attitude uncertainty, due to the photon noise, of about (20 μas) × (200/650) × $[0.02 \times (240/15)]^{1/2} \simeq 6$ μas. To this should be added the attitude modelling error (i.e., how well the spline can represent the effective attitude), which is difficult to estimate without more reliable information about attitude irregularities (Appendix D.4). Generally speaking, the optimum knot interval will roughly balance the estimation and modelling uncertainties, so that the total uncertainty is less than twice the estimation uncertainty. Assuming a total AL attitude error of 12 μas RMS, this would give less than 4 μas RMS to be added quadratically to the parallax uncertainties. Thus, the attitude contribution to the final astrometric parameters appears to be relatively small even for the bright stars. However, this does not take into account the additional complications caused by the gated observations (Appendix D.3) and residual calibration errors due to, for example, CTI effects (Appendix D.2).

7.2.4. Calibration results

The calibration model used for the run merely contained one effect ($N_{\rm AL} = 1$ in Eq. 20), viz., the large-scale calibration $\Delta\eta$ in Eq. (15) accounting for geometric distortions of the CCDs and optical effects which are indistinguishable from geometric irregularities of the focal plane. On this account, we expect the simulated basic angle signal (see beginning of Sect. 7.2) to manifest itself through a corresponding time-dependence of the AL large-scale calibration parameters $\Delta\eta_{0fn0j}$. The asterisks in Fig. 15 show, for f = PFoV and for each time interval j, the calibration parameter values of the demonstration solution averaged over the 62 CCDs (n). The solid line depicts the step-sinusoidal input basic-angle signal applicable for PFoV. As anticipated, the estimated calibration parameters are in very good agreement with the input signal: the RMS value of the differences is 0.20 μas, corresponding to an RMS error of 0.40 μas for the basic angle offsets $\Delta\Gamma_j$ per calibration time interval (cf. Eq. 19). This is reasonably consistent with the expected precision of the large-scale calibration based on the total weight of the observations, as shown by the following calculation. The mean number of AL observations per calibration time interval and field of view is 1.33×10^7 (cf. Table 1). Assuming, as we did in Sect. 7.2.3, that an AL observation of average weight corresponds to a standard uncertainty of $\sigma_{\rm AL} \simeq 650$ μas, the expected precision of the basic angle determination is $2^{1/2} \times (650~\mu{\rm as}) \times [1.33 \times 10^7]^{-1/2} \simeq 0.25$ μas. The observed scatter, 0.40 μas, is larger by roughly the same factor as found for the AL attitude errors.

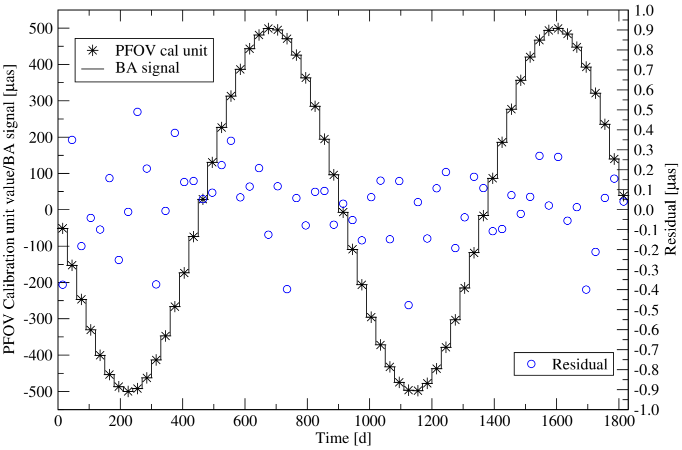


Fig. 15. Variation of the AL large-scale calibration parameters (averaged over all CCDs) in the preceding field of view (PFoV), as a function of the time since the beginning of the mission. The step-sinusoidal curve is the expected variation due to the simulated basic-angle variation having a period of 2.5 yr and an amplitude of 500 μas, but constant within each 30 day interval. The asterisks show the results of the solution (one value per 30 day interval), and the circles show the differences on the magnified scale to the right. Thanks to the constraint in Eq. (16), the mean calibration parameters in the following field of view (FFoV) exactly mirror the displayed ones, and are therefore not shown.

The good agreement between the input calibration signal and the recovered parameters demonstrates the correct functioning of the generic calibration model (see Sect. 3.4) in this simple case. We expect that many more validation runs will be needed to confirm this result in more complex circumstances, i.e., with more calibration effects of different functional compositions and dependencies. A further important aspect is the practical study of possible hidden correlations and degeneracies of calibration with source, attitude, and global parameters which may not be fully obvious at the mathematical level.

7.2.5. Global results

Starting the iterations from a PPN γ value of 1.1, the purpose of running with the Global block was to see how such a grossly wrong initial value would affect the (overall) convergence rate and to what level the correct value could be recovered (recall that the input data were simulated using $\gamma = 1$).

Figure 16 presents the evolution of $g_0 = \gamma - 1$ during the run. In iteration 135 the parameter had settled on the value $g_0 =$

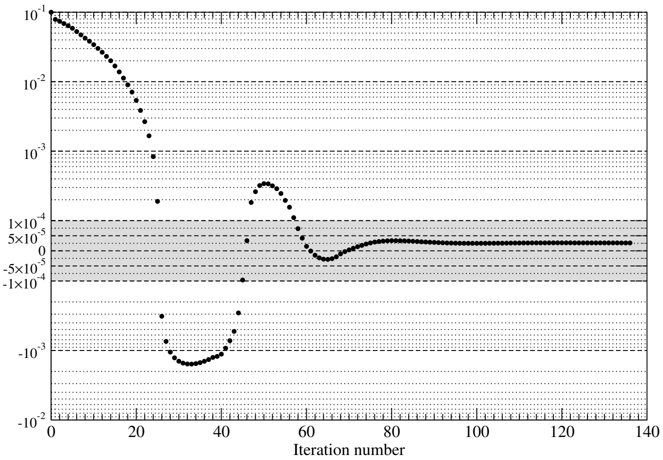


Fig. 16. Evolution of the estimated global parameter $g_0 = \gamma - 1$ (where γ is the PPN parameter) as a function of the iteration number. g_0 settles at a level of 2.6×10^{-5} from a starting value of $g_0 = 0.1$. The formal uncertainty of g_0 in this solution is 2.15×10^{-5}. Note that a linear scale is used for $|g_0| < 10^{-4}$ (the grey area), while a logarithmic scale is used outside of this interval.

2.62×10^{-5}, with a formal standard uncertainty[21] of 2.15×10^{-5}. At +1.2 standard deviations, this value of g_0 is not significantly different from 0. The run shows that the system is capable of recovering the 'correct' value with an error compatible with the statistical uncertainty, which in this case is set by the photon noise of the individual observations.

The total astrometric weight of the AL observations in the demonstration run is $\simeq 4000\ \mu\text{as}^{-2}$. According to Hobbs et al. (2010) this should yield an RMS uncertainty in γ of about 3×10^{-5}, in reasonable agreement with the formal standard uncertainty given above and the parameter value obtained in the solution.

7.3. Processing times

The demonstration solution was run on an IBM cluster at ESAC using 14 nodes (out of 32 available), each node having two processors[22] with four cores each; thus in total 112 CPUs were engaged. This configuration of 14 nodes is estimated to have a total floating point performance of 0.65 Tflop s^{-1} (0.65×10^{12} floating point operations per second). One iteration took about 1 hr (with high CPU occupancy, typically 90%), so the total run time for 135 iterations was nearly 6 days, corresponding to about 3×10^{17} floating point operations (flop).

Scaled up to the projected 10^8 primary sources of the final AGIS run this would amount to 1.5×10^{19} flop. Using a more conservative estimate of 5×10^{19} flop to account for additional features not included in the demonstration run, this will require some 60 days on a typically targeted 10 Tflop s^{-1} machine. On the other hand, there could also be a significant saving in computing time due to the fact that the final solution will start off from a previous solution, already very close to the final one; a smaller number of iterations might therefore suffice. In summary, the estimated processing time is clearly within the feasible range.

8. Conclusions

A fundamental part of the scientific data processing for the Gaia mission is the astrometric core solution, which will be run during and after the mission (ca. 2013–2020) with successively larger datasets and eventually encompassing at least some 100 million primary sources. This solution is central to the performance of the mission as a whole, since it not only provides the astrometric results for these primary sources, but also the reference frame (in the form of the instrument attitude as a function of time) and the geometric calibration of the instrument, for use by a large number of other processes in the overall scientific reduction of the Gaia data.

In order to accomplish the astrometric core solution, a software system known as AGIS (Astrometric Global Iterative Solution) is being built within Coordination Unit 3 (CU3) of the Gaia Data Processing and Analysis Consortium (DPAC). As detailed in this paper, the necessary mathematical models and numerical algorithms are well understood, and have been developed with sufficient rigour to allow the potential accuracy of Gaia to be fully exploited. Most critical parts of this system have been implemented, and numerous test runs have demonstrated the theoretical validity of the global iterative approach, as well as its practical feasibility in terms of data management and computations. While a number of additional features will have to be included in the software before it can be considered ready for the flight data, and many more complications will undoubtedly be discovered during the actual analysis of these data, all the fundamental parts of AGIS are already in place.

In 2011, with roughly two years left until the launch of the satellite and time to mature the concepts and software presented here into a robust operational system, we have no reason to doubt that AGIS will be able to compute an accurate astrometric solution, consistent with the ambitious goals of the Gaia mission.

Acknowledgements. The constant work of the Gaia Data Processing and Analysis Consortium (DPAC) has played an important part in this work. We are particularly indebted to CU2 for the production of independently simulated Gaia-like data for use in the system. The data simulations have been done in the supercomputer Mare Nostrum at Barcelona Supercomputing Center – Centro Nacional de Supercomputación (The Spanish National Supercomputing Center). Research and development in Sweden is kindly supported by the Swedish National Space Board (SNSB).

The successful development of concepts, algorithms and software incorporated in AGIS over a number of years, as well as the drafting of this manuscript, has benefited from the interaction with numerous colleagues, of which we especially wish to mention Sergei Klioner, Alex Bombrun, Alexey Butkevich, Jos de Bruijne, Berry Holl, Floor van Leeuwen, and François Mignard.

We thank the referee, Dr. F. van Leeuwen, for numerous constructive comments which helped to improve the original version of the manuscript.

Our special thanks go to Gaia's former Project Scientist Michael Perryman, whose vision, leadership, and enthusiasm in the early years of the project laid the foundations for the excellent progress that is today seen throughout DPAC and with AGIS in particular.

References

Axelsson, O. 1996, Iterative Solution Methods (Cambridge University Press)
Bastian, U. 1995, in ESA Special Publication, Vol. 379, Future Possibilities for astrometry in Space, ed. M. A. C. Perryman & F. van Leeuwen, 99
Bastian, U. & Biermann, M. 2005, A&A, 438, 745
Björck, Å. 1996, Numerical Methods for Least Squares Problems (Society for Industrial and Applied Mathematics)

[21] The formal uncertainty of g_0 was calculated as $N_{gg}^{-1/2}$ (Eq. 31) times the factor $(1 - \rho)^{-1/2} = \cot(\xi/2) = 2.414$, where $\xi = 45°$ is the constant angle between Gaia's spin axis (z) and the direction to the Sun; as explained by Hobbs et al. (2010), this factor takes into account the statistical correlation (ρ) between PPN γ and the parallaxes. The correction factor would not have been required, had the solution included the pseudo-parameter g_1 (Sect. 5.4).

[22] Intel Nehalem EP Xeon (quad-core, 2.93 GHz), with 32 GB RAM

Bombrun, A., Lindegren, L., Hobbs, D., et al. 2011, A&A, submitted

Bombrun, A., Lindegren, L., Holl, B., & Jordan, S. 2010, A&A, 516, A77

Chambers, J., James, D., Lambert, D., & Vander Wiel, S. 2006, Statistical Science, 21, 463

de Boor, C. 2001, A Practical Guide to Splines, Rev. ed., Applied Mathematical Sciences, Vol. 27 (Springer)

de Bruijne, J., Siddiqui, H., Lammers, U., et al. 2010, in IAU Symposium, Vol. 261, IAU Symposium, ed. S. A. Klioner, P. K. Seidelmann, & M. H. Soffel, 331–333

Dongarra, J., Bunch, J., Moler, C., & Stewart, G. 1979, LINPACK Users' Guide (Philadelphia, PA, USA: Society for Industrial and Applied Mathematics), 367

Dravins, D., Lindegren, L., & Madsen, S. 1999, A&A, 348, 1040

ESA. 1997, The Hipparcos and Tycho Catalogues, ESA SP-1200

Feissel, M. & Mignard, F. 1998, A&A, 331, L33

Gilbert, A. C., Kotidis, Y., Muthukrishnan, S., & Strass, M. J. 2002, in Proc. 28th International Conference on Very Large Data Bases, Hong Kong, China, 454

Golub, G. H. & van Loan, C. F. 1996, Matrix computations, 3rd ed. (The Johns Hopkins University Press, Baltimore)

Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, ApJ, 622, 759

Greenbaum, A. 1997, Iterative Methods for Solving Linear Systems (Society for Industrial and Applied Mathematics)

Greenwald, M. & Khanna, S. 2001, in Proc. ACM SIGMOD International Conference on Management of Data, Santa Barbara, CA, 58

Gunn, J. E., Carr, M., Rockosi, C., et al. 1998, AJ, 116, 3040

Gwinn, C. R., Eubanks, T. M., Pyne, T., Birkinshaw, M., & Matsakis, D. N. 1997, ApJ, 485, 87

Hamilton, W. R. 1843, in Proceedings of the Royal Irish Academy, Vol. 2, 424

Hansen, P. C. 1998, Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion, Monographs on Mathematical Modeling and Computation 4 (SIAM)

Higham, N. 1990, in Reliable Numerical Computation, ed. M. G. Cox & S. J. Hammarling (Oxford University Press), 161

Higham, N. J. 2002, Accuracy and Stability of Numerical Algorithms, 2nd edn. (Philadelphia, PA, USA: Society for Industrial and Applied Mathematics), 680

Hobbs, D., Holl, B., Lindegren, L., et al. 2010, in IAU Symposium No. 261, ed. S. A. Klioner, P. K. Seidelmann, & M. H. Soffel, 315

Høg, E. 2008, in IAU Symposium No. 248, ed. W. J. Jin, I. Platais, & M. A. C. Perryman, 300

Høg, E., Fabricius, C., Makarov, V. V., et al. 2000, A&A, 355, L27

Holl, B., Hobbs, D., & Lindegren, L. 2010, in IAU Symposium No. 261, ed. S. A. Klioner, P. K. Seidelmann, & M. H. Soffel, 320

Holl, B., Lindegren, L., & Hobbs, D. 2012, in preparation for A&A

Huber, P. 1981, Robust Statistics (Wiley)

Janesick, J. R. 2001, Scientific charge-coupled devices (SPIE Optical Engineering Press)

Jordi, C., Gebran, M., Carrasco, J. M., et al. 2010, A&A, 523, A48+

Kane, T. R., Likins, P. W., & Levinson, D. A. 1983, Spacecraft dynamics (McGraw Hill Book Company)

Klioner, S. A. 2003, AJ, 125, 1580

Klioner, S. A. 2004, Phys. Rev. D, 69, 124001

Klumpp, A. R. 1976, J. Spacecraft, 13, 754

Kopeikin, S. M. & Makarov, V. V. 2006, AJ, 131, 1471

Laborie, A., Davancens, R., Pouny, P., et al. 2007, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 6690, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series

Lammers, U., Lindegren, L., O'Mullane, W., & Hobbs, D. 2009, in ASPC Series, Vol. 411, Astronomical Data Analysis Software and Systems XVIII, ed. D. A. Bohlender, D. Durand, & P. Dowler, 55

Lawson, C. L. & Hanson, R. J. 1974, Solving Least Squares Problems (Prentice-Hall, New Jersey)

Lindegren, L. 2005, in ESA Special Publication, Vol. 576, The Three-Dimensional Universe with Gaia, ed. C. Turon, K. S. O'Flaherty, & M. A. C. Perryman, 29

Lindegren, L. 2010, in IAU Symposium No. 261, ed. S. A. Klioner, P. K. Seidelmann, & M. H. Soffel, 296

Lindegren, L., Babusiaux, C., Bailer-Jones, C., et al. 2008, in IAU Symposium No. 248, ed. W. J. Jin, I. Platais, & M. A. C. Perryman, 217

Lindegren, L. & Bastian, U. 2011, in EAS Publications Series, Vol. 45, 109–114

Lindegren, L. & Dravins, D. 2003, A&A, 401, 1185

Lindegren, L. & Kovalevsky, J. 1995, A&AS, 304, 189

Lindegren, L. & Perryman, M. A. C. 1996, A&A, 116, 579

Luri, X. & Babusiaux, C. 2011, in EAS Publications Series, Vol. 45, EAS Publications Series, 25–30

Masana, E., Luri, X., Anglada-Escudé, G., & Llimona, P. 2005, in ESA Special Publication, Vol. 576, The Three-Dimensional Universe with Gaia, ed. C. Turon, K. S. O'Flaherty, & M. A. C. Perryman, 457–+

Mignard, F., Bailer-Jones, C., Bastian, U., et al. 2008, in IAU Symposium No. 248, ed. W. J. Jin, I. Platais, & M. A. C. Perryman, 224

Murray, C. A. 1983, Vectorial astrometry (Bristol: Adam Hilger)

O'Mullane, W., Banday, A. J., Górski, K. M., Kunszt, P., & Szalay, A. S. 2001, in Mining the Sky, ed. A. J. Banday, S. Zaroubi, & M. Bartelmann, 638

O'Mullane, W., Lammers, U., Bailer-Jones, C., et al. 2007, in Astronomical Society of the Pacific Conference Series, Vol. 376, Astronomical Data Analysis Software and Systems XVI, ed. R. A. Shaw, F. Hill, & D. J. Bell, 99

O'Mullane, W., Lammers, U., Hernández, J., et al. 2010, in Astronomical Society of the Pacific Conference Series, Vol. 434, Astronomical Data Analysis Software and Systems XIX, ed. Y. Mizumoto, K.-I. Morita, & M. Ohishi, 135

O'Mullane, W., Luri, X., Parsons, P., et al. 2011, Experimental Astronomy, 1, 10.1007/s10686-011-9241-6

Perryman, M. A. C., de Boer, K. S., Gilmore, G., et al. 2001, A&A, 369, 339

Press, W., Teukolsky, S., Vetterling, W., & Flannery, B. 2007, Numerical Recipes: The Art of Scientific Computing, 3rd edn. (Cambridge University Press)

Prod'homme, T., Brown, A. G. A., Lindegren, L., Short, A. D. T., & Brown, S. W. 2011, MNRAS, 414, 2215

Prod'Homme, T., Weiler, M., Brown, S. W., Short, A. D. T., & Brown, A. G. A. 2010, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 7742, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series

Proust, D., Quintana, H., Carrasco, E. R., et al. 2006, A&A, 447, 133

Robin, A. C., Reylé, C., Derrière, S., & Picaud, S. 2003, Astronomy and Astrophysics, 409, 523

Schnabel, R. B. & Eskow, E. 1999, SIAM Journal on Optimization, 9, 1135

Seabroke, G. M., Holland, A. D., Burt, D., & Robbins, M. S. 2009, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 7439, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series

Short, A., Prod'homme, T., Weiler, M., Brown, S., & Brown, A. 2010, in Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, Vol. 7742, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series

Soffel, M., Klioner, S. A., Petit, G., et al. 2003, AJ, 126, 2687

Stewart, G. 1998, Matrix Algorithms: Basic decompositions (Society for Industrial and Applied Mathematics)

Turon, C., O'Flaherty, K. S., & Perryman, M. A. C., eds. 2005, ESA Special Publication, Vol. 576, The Three-Dimensional Universe with Gaia

van der Vorst, H. 2003, Iterative Krylov Methods for Large Linear Systems (Cambridge University Press)

van Leeuwen, F. 2005, A&A, 439, 805

van Leeuwen, F. 2007, Hipparcos, the New Reduction of the Raw Data, Astrophysics and Space Science Laboratory. Vol. 350 (Springer)

Wertz, J. R. 1978, Spacecraft Attitude Determination and Control (Kluwer Academic Publishers)

Appendix A: Quaternions

Quaternions form a non-commutative algebra in $\mathbb{R}^4$. Invented by W. R. Hamilton in 1843 as an extension of the complex numbers (Hamilton 1843), their most common usage today is for representing spatial rotations in a particularly compact and convenient way, with applications for example in computer graphics and spacecraft attitude control. Quaternions can be introduced and understood in many different ways, with a correspondingly confusing multitude of notations and conventions. Here we give just a brief introduction to the subject, stating only the minimum results needed for our applications. For more details, see for example Wertz (1978) and Kane et al. (1983).

A.1. Quaternion algebra

A quaternion is a quadruple of real numbers for which the following algebraic operations are defined. For any quaternions $\mathbf{a} = \{a_x,\, a_y,\, a_z,\, a_w\}$ and $\mathbf{b} = \{b_x,\, b_y,\, b_z,\, b_w\}$ we have addition

$$\mathbf{a} + \mathbf{b} = \left\{a_x + b_x,\ a_y + b_y,\ a_z + b_z,\ a_w + b_w\right\}, \tag{A.1}$$

multiplication

$$\begin{aligned}\mathbf{ab} = \Big\{\ & a_x b_w + a_y b_z - a_z b_y + a_w b_x, \\ & -a_x b_z + a_y b_w + a_z b_x + a_w b_y, \\ & a_x b_y - a_y b_x + a_z b_w + a_w b_z, \\ & -a_x b_x - a_y b_y - a_z b_z + a_w b_w \Big\}\,,\end{aligned} \tag{A.2}$$

and scalar multiplication

$$s\mathbf{a} = \left\{ sa_x,\ sa_y,\ sa_z,\ sa_w \right\} \tag{A.3}$$

for scalar s. Subtraction is analogous to addition, as derived from $\mathbf{a} - \mathbf{b} = \mathbf{a} + (-1)\mathbf{b}$. The conjugate of $\mathbf{a}$ is

$$\mathbf{a}^* = \left\{ -a_x,\ -a_y,\ -a_z,\ a_w \right\}, \tag{A.4}$$

the norm (length) is

$$\|\mathbf{a}\| = (a_x^2 + a_y^2 + a_z^2 + a_w^2)^{1/2}\,, \tag{A.5}$$

and the inverse (provided $\|\mathbf{a}\| > 0$) is

$$\mathbf{a}^{-1} = \|\mathbf{a}\|^{-2}\,\mathbf{a}^*\,. \tag{A.6}$$

We have

$$(\mathbf{ab})^* = \mathbf{b}^*\mathbf{a}^*\,, \qquad (\mathbf{ab})^{-1} = \mathbf{b}^{-1}\mathbf{a}^{-1}\,. \tag{A.7}$$

Any non-zero quaternion can be normalized to unit length. In analogy with the notation for vector normalization, we use angular brackets for this operation:

$$\langle \mathbf{a} \rangle = \|\mathbf{a}\|^{-1}\mathbf{a}\,. \tag{A.8}$$

The triplet of real numbers (a_x, a_y, a_z) can be regarded as the coordinates of the vector $\boldsymbol{a}$ in some reference system $\mathsf{S} = [\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$ (where $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ is a right-handed set of orthogonal unit vectors). Thus, we can write $\mathbf{a} = \{\mathsf{S}'\boldsymbol{a},\, a_w\}$, where $\mathsf{S}'\boldsymbol{a} = [a_x\ a_y\ a_z]'$ constitutes the so-called vector part of the quaternion, and a_w the scalar part. Both scalars and vectors (in $\mathbb{R}^3$) can thus be seen as special cases of quaternions, namely, when the vector or the scalar part is zero. This allows us to write for example $\|\mathbf{a}\|^2 = \mathbf{aa}^* = \mathbf{a}^*\mathbf{a}$. In terms of the usual vector-scalar operations the quaternion multiplication can also be written as

$$\mathbf{ab} = \left\{ \mathsf{S}'\left(\boldsymbol{a}\times\boldsymbol{b} + \boldsymbol{a}b_w + \boldsymbol{b}a_w\right),\ a_w b_w - \boldsymbol{a}'\boldsymbol{b} \right\}. \tag{A.9}$$

Note that the vector part of a quaternion only makes sense when expressed in some coordinate system (S in this example); a physical vector cannot be part of a quaternion.

A.2. Spatial rotations

Unit quaternions (of unit length) are convenient for representing orientations and spins of objects in three-dimensional space. This is compacter, numerically more stable and requires fewer arithmetic operations than the use of rotation matrices. Compared with the use of Euler angles, much fewer or no trigonometric functions need to be evaluated, and singularities are avoided.

According to Euler's rotation theorem any change in the orientation of an object can be described as a rotation by a certain angle around some fixed axis. Let this axis be represented by the unit vector $\boldsymbol{u}$ and the rotation angle by ϕ, reckoned in the positive sense around the vector. In the given reference system $\mathsf{S} = [\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$, the rotation is then represented by the unit quaternion

$$\mathbf{q} = \left\{ \mathsf{S}'\boldsymbol{u}\sin\frac{\phi}{2},\ \cos\frac{\phi}{2} \right\}. \tag{A.10}$$

The useful property here is that a sequence of rotations is represented by the product of the corresponding unit quaternions (see Sect. A.3).

From Eq. (A.6) it follows that the inverse of a unit quaternion equals its conjugate, so

$$\mathbf{q}^{-1} = \mathbf{q}^* = \left\{ -\mathsf{S}'\boldsymbol{u}\sin\frac{\phi}{2},\ \cos\frac{\phi}{2} \right\}, \tag{A.11}$$

which represents a rotation by $-\phi$ around $\boldsymbol{u}$.

A rotation by the angle 2π around any axis is represented by the unit quaternion $\{0, 0, 0, -1\}$. Since this operation is physically equivalent to no rotation at all, it implies a sign ambiguity in the quaternion representation of any given rotation (modulo 2π). This is potentially a problem only when the quaternion is used to represent a continuously changing orientation as a function of time, as is the case for the Gaia attitude (Sect. 3.3). It is then necessary to ensure that no sign flips occur, e.g., when converting from some other representation of the orientations.

Equation (A.10) suggests an alternative, non-redundant, way of representing spatial rotation, namely by means of the components of the vector $\boldsymbol{\phi} = \boldsymbol{u}\phi$. For any continuous rotation the three components could be continuous functions of time. This formalism is mainly useful for small rotations ($\|\boldsymbol{\phi}\| \ll 1$); when applied for example to a spinning satellite the length of the vector may grow indefinitely, causing unacceptable numerical errors in finite arithmetic. For the arbitrary vector $\boldsymbol{\phi}$ we nevertheless introduce the special notation $\mathbf{Q}(\mathsf{S}'\boldsymbol{\phi})$ for the unit quaternion, in the reference system S, representing a spatial rotation by the angle $\phi = \|\boldsymbol{\phi}\|$ about an axis parallel to $\boldsymbol{\phi}$:

$$\mathbf{Q}(\mathsf{S}'\boldsymbol{\phi}) = \begin{cases} \left\{ \mathsf{S}'\langle\boldsymbol{\phi}\rangle\sin\dfrac{\phi}{2},\ \cos\dfrac{\phi}{2} \right\} & \text{if } \phi > 0, \\ \{0, 0, 0, 1\} & \text{if } \phi = 0. \end{cases} \tag{A.12}$$

This notation is here only used when discussing the small rotational offset between the ICRS and AGIS frames in Sect. 6.1.1.

A.3. Vector and frame rotations

Vector rotation and frame rotation are not standard notions in vector or quaternion calculus, but we have found them helpful in order to clarify the relations between vectors and their representations in different reference systems.

Vector rotation. Let $\{\mathsf{S}'\boldsymbol{r}_0,\, 0\}$ be the quaternion representation in the reference system S of the arbitrary vector $\boldsymbol{r}_0$. Rotating the vector an angle ϕ around unit vector $\boldsymbol{u}$ results in a new vector $\boldsymbol{r}_1$, whose coordinates in S can be calculated by two successive quaternion multiplications,

$$\left\{ \mathsf{S}'\boldsymbol{r}_1,\ 0 \right\} = \mathbf{q}\left\{ \mathsf{S}'\boldsymbol{r}_0,\ 0 \right\}\mathbf{q}^{-1}\,, \tag{A.13}$$

where $\mathbf{q}$ is given by Eq. (A.10) and $\mathbf{q}^{-1} = \mathbf{q}^*$. We call the operation in Eq. (A.13) for the *vector rotation* of $\boldsymbol{r}_0$ by the quaternion $\mathbf{q}$. This calculation requires the use of a particular reference system (S in this example) in which to express the vectors and the quaternion; the resulting physical vector $\boldsymbol{r}_1$ is however independent of the reference system.

Applying n successive vector rotations, specified by the quaternions $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$, gives the vector $\boldsymbol{r}_n$ in

$$\left\{\mathsf{S}'\boldsymbol{r}_n\,, 0\right\} = \mathbf{q}_n \cdots \mathbf{q}_2\mathbf{q}_1 \left\{\mathsf{S}'\boldsymbol{r}_0\,, 0\right\} \mathbf{q}_1^{-1}\mathbf{q}_2^{-1} \cdots \mathbf{q}_n^{-1}\,. \qquad \text{(A.14)}$$

This is equivalent to a single vector rotation by $\mathbf{q} = \mathbf{q}_n \cdots \mathbf{q}_2\mathbf{q}_1$. All the quaternions are here expressed in the fixed reference system S, and the order of multiplication is from right to left.

Frame rotation. A more common application in astrometry is where the reference system itself is rotated, say from $\mathsf{S}_0 = [\boldsymbol{x}_0\ \boldsymbol{y}_0\ \boldsymbol{z}_0]$ to $\mathsf{S}_1 = [\boldsymbol{x}_1\ \boldsymbol{y}_1\ \boldsymbol{z}_1]$, by the quaternion $\mathbf{q}$. Given the coordinates $\mathsf{S}_0'\boldsymbol{r}$ of the arbitrary vector $\boldsymbol{r}$ in reference system S_0, we want to compute the coordinates $\mathsf{S}_1'\boldsymbol{r}$ of the same physical vector in the rotated reference system. By applying a vector rotation to each of the basis vectors we find

$$\left\{\mathsf{S}_1'\boldsymbol{r},\ 0\right\} = \mathbf{q}^{-1}\left\{\mathsf{S}_0'\boldsymbol{r},\ 0\right\}\mathbf{q}\,, \qquad \text{(A.15)}$$

which we refer to as the *frame rotation* of $\boldsymbol{r}$ by the quaternion $\mathbf{q}$.

It is important to note that the numerical representation of the quaternion $\mathbf{q}$, representing a frame rotation from S_0 to S_1, is the same in the two frames. This follows because the components of the vector $\boldsymbol{u}$ are invariant under a frame rotation about the vector itself, i.e., $\mathsf{S}_1'\boldsymbol{u} = \mathsf{S}_0'\boldsymbol{u}$. In Eq. (A.10) the vector part of $\mathbf{q}$ can therefore be expressed in either of the two reference systems, i.e., we may take $\mathsf{S} = \mathsf{S}_0$ or $\mathsf{S} = \mathsf{S}_1$. The scalar part $\cos(\phi/2)$ is of course independent of the reference system.

Successive frame rotations can therefore be accomplished by referring each rotation to the current set of axes, which is usually precisely what is needed. Let $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$ be a sequence of frame rotations, from S_0 to S_1, and then from S_1 to S_2, etc.; here $\mathbf{q}_1$ is expressed in S_0 (or S_1), $\mathbf{q}_2$ in S_1 (or S_2), and so on. The corresponding transformation of the coordinates of the vector $\boldsymbol{r}$ is given by

$$\left\{\mathsf{S}_n'\boldsymbol{r},\ 0\right\} = \mathbf{q}_n^{-1} \cdots \mathbf{q}_2^{-1}\mathbf{q}_1^{-1}\left\{\mathsf{S}_0'\boldsymbol{r},\ 0\right\}\mathbf{q}_1\mathbf{q}_2 \cdots \mathbf{q}_n\,, \qquad \text{(A.16)}$$

equivalent to a single frame rotation by $\mathbf{q} = \mathbf{q}_1\mathbf{q}_2 \cdots \mathbf{q}_n$. The quaternions are here expressed in the concurrent reference system, and the order of multiplication is from left to right.

A.4. Angular velocity

Let $\mathbf{q}$ be a unit quaternion, expressed in the non-rotating reference system C, and let us assume that $\mathbf{q}$ is a differentiable function of time. The angular velocity $\boldsymbol{\Omega}$ associated with the time-dependent vector rotation by $\mathbf{q}$ is the same for all vectors, and given by

$$\left\{\mathsf{C}'\boldsymbol{\Omega},\ 0\right\} = 2\dot{\mathbf{q}}\mathbf{q}^{-1}\,, \qquad \text{(A.17)}$$

where the dot signifies the time derivative. This result can be derived by taking the time derivative of the vector rotation formula for arbitrary vector $\boldsymbol{r}$, using $\dot{\boldsymbol{r}} = \boldsymbol{\Omega}\times\boldsymbol{r}$ and Eq. (A.9).

Let S be the reference system obtained after rotation by $\mathbf{q}$. The coordinates of the angular velocity vector in the new system are found by performing a frame rotation of Eq. (A.17) by $\mathbf{q}$; thus

$$\left\{\mathsf{S}'\boldsymbol{\Omega},\ 0\right\} = \mathbf{q}^{-1}\left\{\mathsf{C}'\boldsymbol{\Omega},\ 0\right\}\mathbf{q} = 2\mathbf{q}^{-1}\dot{\mathbf{q}}\,. \qquad \text{(A.18)}$$

These expressions show, for example, how the instantaneous angular velocity of Gaia can be calculated either in the celestial reference system (CoMRS), using Eq. (A.17), or in the instrument system (SRS), using Eq. (A.18), from a knowledge of the attitude $\mathbf{q}$ and its time derivative $\dot{\mathbf{q}}$.

Algorithm A.1 For given $\boldsymbol{A} = [A_{xx}\ A_{xy}\ A_{xz};\ A_{yx}\ \cdots]$, this algorithm returns a unit quaternion $\mathbf{q} = \{q_x,\ q_y,\ q_z,\ q_w\}$ such that Eq. (A.19) is satisfied.

```
$s \leftarrow A_{xx} + A_{yy} + A_{zz}$
for $i = x,\ y,\ z$ do
    $|q_i| \leftarrow [(1 - s)/4 + A_{ii}/2]^{1/2}$
end for
$q_w \leftarrow [(1 + s)/4]^{1/2}$
if $|q_x| \geq \max(|q_y|, |q_z|)$ then
    $i \leftarrow x,\ \ j \leftarrow y,\ \ k \leftarrow z$
else if $|q_y| \geq \max(|q_x|, |q_z|)$ then
    $i \leftarrow y,\ \ j \leftarrow z,\ \ k \leftarrow x$
else
    $i \leftarrow z,\ \ j \leftarrow x,\ \ k \leftarrow y$
end if
$q_i \leftarrow |q_i|\,\mathrm{sign}[A_{jk} - A_{kj}]$
$q_j \leftarrow |q_j|\,\mathrm{sign}[q_i(A_{ij} + A_{ji})]$
$q_k \leftarrow |q_k|\,\mathrm{sign}[q_i(A_{ik} + A_{ki})]$
```

A.5. The attitude matrix

For completeness, we give here the full relations between the attitude matrix defined in Sect. 3.3 and the quaternion representation of the attitude. Let $\mathsf{C} = [\boldsymbol{X}\ \boldsymbol{Y}\ \boldsymbol{Z}]$ be the celestial reference system (CoMRS; Sect. 3.1) and $\mathsf{S} = [\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$ the instrument system (SRS). The attitude matrix $\boldsymbol{A}$ is defined by Eq. (8). The attitude quaternion $\mathbf{q} = \{q_x,\ q_y,\ q_z,\ q_w\}$ gives the rotation from the CoMRS to the SRS, i.e., $\{\mathsf{C}'\boldsymbol{x}, 0\} = \mathbf{q}\,\{\mathsf{C}'\boldsymbol{X}, 0\}\,\mathbf{q}^{-1}$, etc. Then

$$\boldsymbol{A} = \begin{bmatrix} 1 - 2(q_y^2 + q_z^2) & 2(q_xq_y + q_zq_w) & 2(q_xq_z - q_yq_w) \\ 2(q_xq_y - q_zq_w) & 1 - 2(q_x^2 + q_z^2) & 2(q_yq_z + q_xq_w) \\ 2(q_xq_z + q_yq_w) & 2(q_yq_z - q_xq_w) & 1 - 2(q_x^2 + q_y^2) \end{bmatrix}. \qquad \text{(A.19)}$$

The conversion from $\boldsymbol{A}$ to $\mathbf{q}$ is less straightforward if we want to avoid numerical problems for certain attitudes. A stable algorithm was given by Klumpp (1976). In our notation, using pseudo-code, it is given by Algorithm A.1. Note that $\{q_x,\ q_y,\ q_z,\ q_w\}$ and $\{-q_x,\ -q_y,\ -q_z,\ -q_w\}$ correspond to the same $\boldsymbol{A}$, while the algorithm always returns a quaternion with $q_w \geq 0$; a reversal of the signs may therefore sometimes be required to ensure the temporal continuity of the quaternion components.

A.6. Differential rotation

Up until now, the quaternion formulae given in this Appendix hold rigorously for arbitrary rotations. We now derive a useful result, which however is only valid to first order in the (small) rotation angles. It can be used, for example, to compute the attitude error angles about the SRS axes, as was done in Sect. 7.2.3.

Let $\mathbf{q}_0$ and $\mathbf{q}_1$ be given unit quaternions, representing the two nearly co-aligned reference systems $\mathsf{S}_0 = [\boldsymbol{x}_0\ \boldsymbol{y}_0\ \boldsymbol{z}_0]$ and $\mathsf{S}_1 = [\boldsymbol{x}_1\ \boldsymbol{y}_1\ \boldsymbol{z}_1]$ with respect to some third (common) reference system such as the ICRS. It is required to express the difference between S_1 and S_0 by means of three small angles ϕ_x, ϕ_y, ϕ_z representing rotations about the axes in either system. More precisely, let $\boldsymbol{\phi} = [\phi_x\ \phi_y\ \phi_z]'$ be the spatial rotation that brings S_0 into coincidence with S_1. Assuming that $\|\boldsymbol{\phi}\| \ll 1$ and neglecting terms of order $\|\boldsymbol{\phi}\|^2$, we have $\mathsf{S}_1 \simeq \mathsf{S}_0 + \boldsymbol{\phi}\times\mathsf{S}_0$, or

$$\mathsf{S}_1'\mathsf{S}_0 \simeq \boldsymbol{I} + (\boldsymbol{\phi}\times\mathsf{S}_0)'\,\mathsf{S}_0 = \begin{bmatrix} 1 & \phi_z & -\phi_y \\ -\phi_z & 1 & \phi_x \\ \phi_y & -\phi_x & 1 \end{bmatrix}. \qquad \text{(A.20)}$$

According to Eq. (8) this matrix describes the orientation of S_1 with respect to S_0. If $\mathbf{d}$ is the quaternion representing a frame rotation from S_0 to S_1, we have (Sect. A.3) $\mathbf{q}_1 = \mathbf{q}_0\mathbf{d}$ and hence

$$\mathbf{d} \equiv \{d_x,\, d_y,\, d_z,\, d_w\} = \mathbf{q}_0^{-1}\mathbf{q}_1\,. \qquad \text{(A.21)}$$

Given that $\|\boldsymbol{\phi}\|$ is small, d_x, d_y and d_z are also small quantities, while $|d_w| \simeq 1$. Due to the sign ambiguity of the quaternion representation (Sect. A.2), d_w could however have either sign. Comparing Eqs. (A.19) and (A.20) we find, to first order in the small quantities,

$$\phi_x \simeq 2d_xd_w\,, \quad \phi_y \simeq 2d_yd_w\,, \quad \phi_z \simeq 2d_zd_w\,, \qquad \text{(A.22)}$$

where the factor d_w (being close to ±1) guarantees that the angles obtain their correct signs. Equations (A.21)–(A.22) provide the required transformation from ($\mathbf{q}_0$, $\mathbf{q}_1$) to (ϕ_x, ϕ_y, ϕ_z).

Appendix B: Splines

A spline is a piecewise polynomial function $S(t)$ defined on some interval $[t_{\text{beg}}, t_{\text{end}}]$. That is, if the interval is divided into $K > 0$ sub-intervals by means of the instants t_k (called knots), such that $t_{\text{beg}} = t_0 < t_1 < \cdots < t_K = t_{\text{end}}$, then in the sub-interval $t_k \le t < t_{k+1}$ the spline function $S(t)$ equals some polynomial $P_k(t)$, $k = 0 \ldots K-1$. The splines of interest here consist of polynomials of some fixed order M (or degree $M-1$); typically $M = 4$, corresponding to cubic splines. Moreover, the splines are usually maximally smooth, i.e., $S^{(m)}(t) \equiv \mathrm{d}^mS/\mathrm{d}t^m$ is continuous for $t_{\text{beg}} < t < t_{\text{end}}$ and $m = 0 \ldots M-2$.

The K polynomials of order M require KM coefficients for their specification. For a maximally smooth spline, there are $M-1$ continuity conditions for each interior knot, namely $S^{(m)}(t_k-) = S^{(m)}(t_k+)$ for $m = 0 \ldots M-2$ and $k = 1 \ldots K-1$; thus a total of $(K-1)(M-1)$ constraints. The spline consequently has $KM - (K-1)(M-1) = K + M - 1$ degrees of freedom.[23] In the context of least-squares fitting, this equals the number of unknowns (parameters to fit), which we denote by N. In the following we take N and M to be the characteristic numbers of the spline, rather than K and M. For a maximally smooth spline we have $K = N - M + 1$.

B.1. B-splines

There are many different ways in which a spline could be represented (parametrized). The approach taken here is to consider $S(t)$ as a linear combination of some basis functions $B_n(t)$,

$$S(t) = \sum_{n=0}^{N-1} c_n\, B_n(t)\,, \qquad \text{(B.1)}$$

where c_n are coefficients to be determined. Choosing the basis functions to have minimal support (see below) for the given order and smoothness leads to the so-called B-splines (de Boor 2001).

The B-splines are uniquely defined by the adopted knot sequence. In the following we shall use τ_n (rather than t_n) to denote the knots associated with the B-splines. The reason is that the knot sequence for the B-splines has a slightly more general interpretation than just the simple division of $[t_{\text{beg}}, t_{\text{end}}]$ considered above. Moreover, when fitting the spline to given data points, this allows us to use t_k (for example) to denote the time associated with the kth data point, without ambiguity.

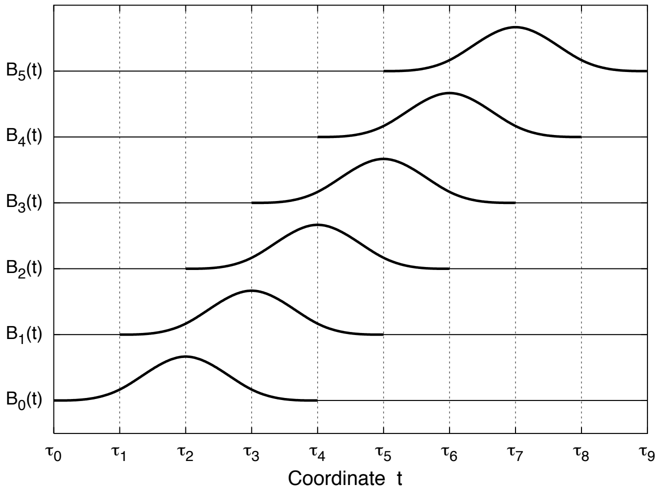


Fig. B.1. The first six B-splines of order $M = 4$ (cubic) defined on the regular knot sequence $\tau_0, \tau_1, \ldots$. For better visibility, each B-spline is vertically displaced by one unit, and the non-zero parts are drawn in thick lines.

The knot sequence must be non-decreasing, $\tau_0 \le \tau_1 \le \tau_2 \le \cdots$, and at least two knots must be different. Very often we use a regular knot sequence in which $\tau_{n+1} = \tau_n + \Delta\tau$ for some $\Delta\tau > 0$ (the knot interval). Figure B.1 shows the first six cubic B-splines defined on a regular knot sequence. Note that the non-zero part of each B-spline, shown in heavy line, stretches over M consecutive knot intervals. We use the convention that the B-splines are labelled with the same index as the left-most knot of its non-zero part; therefore, the support of $B_n(t)$ is (τ_n, τ_{n+M}).

It is not possible to construct a maximally smooth spline function of order M that is non-zero over less than M consecutive knot intervals. This is what we mean by the statement that B-splines have minimal support: they could not be shortened without loosing some smoothness. This is an important property for the least-squares fitting, as shown by the following consideration. Least-squares fitting of (B.1) to a given set of data points (t_k, z_k) (where $t_k \in [t_{\text{beg}}, t_{\text{end}}]$ for each k) will be done by forming normal equations. The normal equations matrix $\boldsymbol{N}$ is a symmetric, positive definite matrix of dimension[24] $N \times N$. Since N may be very large, it is desirable that the matrix is sparse, i.e., that most elements are zero. It is easy to see that element N_{ij} will be non-zero as soon as $B_i(t_k)B_j(t_k) \ne 0$ for some data point k. To make the matrix maximally sparse, we should therefore choose basis functions with minimal support. B-splines have minimal support and are therefore ideal for least-squares fitting using sparse matrix algebra.

Since the support of a B-spline of order M extends over at most M consecutive knot intervals, we have $N_{ij} = 0$ for $|i -$

[23] When the spline is used for interpolation, it is typically chosen to go exactly through the $K+1$ values $S(t_k)$ for $k = 0 \ldots K$. This leaves $(K + M - 1) - (K + 1) = M - 2$ degrees of freedom. Thus, for a cubic spline ($M = 4$), two more conditions must be imposed for the spline to become unique. The most common choice is to make the second derivative vanish at t_{beg} and t_{end}. This defines the 'natural' interpolating spline. By contrast, when splines are least-squares fitted to data, as in the attitude determination problem, there are typically many data points per sub-interval, and no need for special endpoint conditions.

[24] In the attitude determination problem, each of the four components of the quaternion is represented by a spline, so the number of parameters is actually $4N$ and the normal matrix is of dimension $4N \times 4N$. Alternatively, we may take the elements of $\boldsymbol{N}$ to be sub-matrices of dimension 4×4. The indices i and j used below refer to these sub-matrices.

Algorithm B.1 For given spline order M, knot sequence $\{\tau_n\}$, time t, and left index ℓ (such that $\tau_\ell \leq t < \tau_{\ell+1}$), this algorithm returns $\{b_0 \dots b_{M-1}\}$ such that $b_0 = B_{\ell-M+1}(t), \dots, b_{M-1} = B_\ell(t)$.

```
$b_0 \leftarrow 1$
for $i = 0$ to $M - 2$ do
    $R_i \leftarrow \tau_{\ell+i+1} - t$
    $L_i \leftarrow t - \tau_{\ell-i}$
    $s \leftarrow 0$
    for $j = 0$ to $i$ do
        $u \leftarrow b_j/(R_j + L_{i-j})$
        $b_j \leftarrow s + R_j \times u$
        $s \leftarrow L_{i-j} \times u$
    end for
    $b_{i+1} \leftarrow s$
end for
```

$j| > M - 1$. This shows that $\boldsymbol{N}$ is a symmetric banded matrix with (half-) bandwidth equal to $M - 1$. Cholesky decomposition preserves the band structure of the matrix and is therefore ideal for solving the normals; it is also numerically very efficient.

B.2. Calculation of B-splines

At any given point t there are at most M non-zero B-splines, namely $B_{\ell-M+1}(t), B_{\ell-M+2}, \dots B_\ell(t)$, where ℓ is the 'left index' of t, such that $\tau_\ell \leq t < \tau_{\ell+1}$. They can be computed simultaneously in a numerically stable way by means of de Boor's algorithm (de Boor 2001), which is given as pseudo-code in Algorithm B.1. If needed, the derivatives of the B-splines with respect to t can be computed simultaneously with little additional effort.

By inspection of the algorithm it is found that the knots used for computing the B-spline values in the interval $[\tau_\ell, \tau_{\ell+1})$ are $\tau_{\ell-M+2}$ through $\tau_{\ell+M-1}$. For example, with reference to Fig. B.1 (with $M = 4$), the B-splines between τ_3 and τ_4 (i.e., for left index $\ell = 3$) depend on $\{\tau_1, \tau_2, \dots, \tau_6\}$, but not on τ_0 or τ_7, even though $B_0(t)$ in general depends on τ_0 and $B_3(t)$ depends on τ_7.

B.3. Use of multiple knots

Algorithm B.1 works also in the case when several consecutive knots are placed at the same t coordinate. Having a knot of multiplicity m (i.e., m knots at the same t) removes the continuity for derivatives of order $M - m$ and higher. The normal situation is that the knots have multiplicity 1, which means that the spline is continuous at the knots up to and including its $(M-2)$th derivative, but discontinuous in its $(M - 1)$th derivative. By inserting multiple knots at some specific instant, one allows the spline to become less smooth at this point. For example, in a cubic spline ($M = 4$) a triple knot ($m = 3$) allows the first derivative to become discontinuous at that point, while leaving the spline function itself continuous. In the Gaia attitude processing, multiple knots will be used for modelling the effects of micrometeoroid impacts, which cause (almost) instantaneous changes in the angular velocity, corresponding to discontinuities in the first derivative of the attitude spline.

Multiple knots may also be used at the endpoints of the spline interval $[t_\text{beg}, t_\text{end}]$. At any point in this interval, there must be exactly M non-zero B-splines in order that a linear combination of them should to be able to represent an arbitrary spline of order M. Again, with reference to Fig. B.1 (for $M = 4$), we see that this is the case to the right of τ_3 (or τ_{M-1} in general). Thus we should put $\tau_{M-1} = t_\text{beg}$. The first $M - 1$ knots can in principle be placed anywhere, as long as $\tau_0 \leq \tau_1 \leq \dots \leq \tau_{M-1}$: any such arrangement will produce M non-zero B-splines in the next sub-interval (τ_{M-1}, τ_M), and although the B-splines are different depending on the arrangement of the knots, they are always linearly independent and therefore can be used as a basis for the spline. In particular, it is possible to put the first M knots at the same point, i.e., $\tau_0 = \tau_1 = \dots = \tau_{M-1} = t_\text{beg}$.

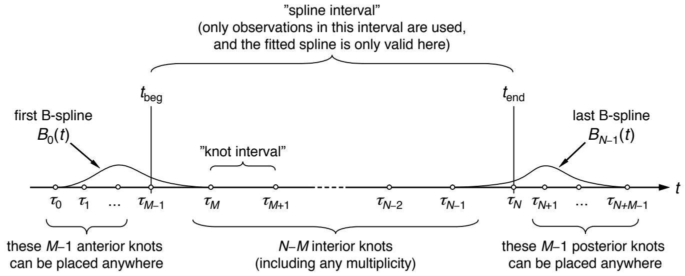


Fig. B.2. Illustration of the knot placement for a spline of order M (e.g., $M = 4$ for a cubic spline) with N degrees of freedom. $[t_\text{beg}, t_\text{end}]$ is the spline interval over which the spline is fitted to given data points. The end knots τ_{M-1} and τ_N are at the endpoints of the spline interval. The $N-M$ interior knots are chosen to give the spline the desired flexibility, including multiple knots where required. The placement of the anterior ($\tau \leq t_\text{beg}$) and posterior knots ($\tau \geq t_\text{end}$) is in principle arbitrary: it does not change the fitted spline in $[t_\text{beg}, t_\text{end}]$, but may affect the condition number of the least-squares fit. The parameters of the fitted spline are the coefficients $c_0, \dots c_{N-1}$ of the B-splines $B_0(t)$ through $B_{N-1}(t)$.

Corresponding considerations apply to the right limit of the spline interval: with N degrees of freedom, the support of the last B-spline $B_{N-1}(t)$ extends from τ_{N-1} to τ_{N+M-1}. The spline interval must end at $\tau_N = t_\text{end}$, and the remaining $M-1$ knots can be placed anywhere provided that $\tau_N \leq \tau_{N+1} \leq \dots \leq \tau_{N+M-1}$. In particular, it is possible to have an M-fold knot at the endpoint of the spline interval, i.e., $t_\text{end} = \tau_N = \tau_{N+1} = \dots = \tau_{N+M-1}$. Figure B.2 summarizes the placement of knots in relation to a given spline interval for given order M and degrees of freedom N (number of spline coefficients).

Although the placement of the first and last $M-1$ knots is arbitrary, and does not change the resulting spline between t_beg and t_end, their placement does affect the numerical stability of the resulting least-squares system. Butkevich & Klioner (unpublished technical note) has pointed out that collapsing the anterior and posterior knots into the end knots, so that $\tau_0 = \tau_1 = \dots = \tau_{M-1} = t_\text{beg}$ and $t_\text{end} = \tau_N = \tau_{N+1} = \dots = \tau_{N+M-1}$) results in a system with much smaller condition number than a regular sequence extending beyond the endpoints. For a cubic spline ($M = 4$) the spline interval then begins and ends with 4-fold knots as illustrated in Fig. B.3. However, the use of data segmentation, as described in Sect. 5.2.2, may not permit this device.

Appendix C: A robust Cholesky algorithm for positive semidefinite matrices without pivoting

C.1. The use of normal equations

The least-squares problems considered in this paper are solved by the method of normal equations (here denoted $\boldsymbol{Nx} = \boldsymbol{b}$), using the Cholesky algorithm to decompose the symmetric normal matrix $\boldsymbol{N}$. This method is known to be computationally efficient

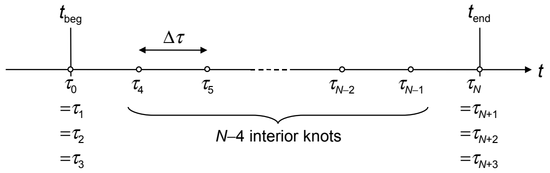


Fig. B.3. Definition of a regular knot sequence for fitting a cubic spline (order $M = 4$) in the interval $[t_{\rm beg}, t_{\rm end}]$. The spline interval is divided into $N - 3$ knot intervals of equal length $\Delta\tau = (t_{\rm end} - t_{\rm beg})/(N - 3)$.

and accurate for well-conditioned problems (e.g., Stewart 1998), and is moreover well adapted to in-place manipulation of sparse matrices such as the band matrix obtained when fitting B-splines (Fig. 5)

The method of normal equations for the general least-squares problem is usually discouraged in the literature, due to the much superior stability and accuracy of alternative methods operating directly on the observation equations $\boldsymbol{Ax} \simeq \boldsymbol{h}$ (where $\boldsymbol{N} = \boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{b} = \boldsymbol{A}'\boldsymbol{h}$), e.g., using Householder orthogonal transformations (e.g., Björck 1996). However, when working with very large problems that are inherently well-posed, thanks to a good design of experiment and reduction model, our experience is that the method of normal equations is nearly always adequate in terms of accuracy, and then has the edge over other methods in terms of speed, storage and simplicity of the code. Moreover, in these problems, iterative improvement of the solution is usually required for other reasons (non-linearity, elimination of outliers), which partly compensates for the loss of precision when forming the normal equations.

C.2. The Cholesky algorithm

The standard Cholesky algorithm (e.g., Björck 1996; Golub & van Loan 1996) requires that $\boldsymbol{N}$ is positive definite, which is always the case for a well-conditioned least-squares problem. Given $\boldsymbol{N}$ and $\boldsymbol{b}$, the solution of the system $\boldsymbol{Nx} = \boldsymbol{b}$ proceeds in three steps:

C1. Use the Cholesky algorithm to find the unique upper-diagonal matrix $\boldsymbol{U}$, with positive diagonal entries, such that $\boldsymbol{N} = \boldsymbol{U}'\boldsymbol{U}$.
C2. Solve the lower-triangular system $\boldsymbol{U}'\boldsymbol{y} = \boldsymbol{b}$.
C3. Solve the upper-triangular system $\boldsymbol{Ux} = \boldsymbol{y}$.

The matrix $\boldsymbol{U}$ (or its transpose) is known as the Cholesky factor or square root of $\boldsymbol{N}$. As all the computations can be made in-place, we have symbolically

$$[\,\boldsymbol{N}\,|\,\boldsymbol{b}\,] \xrightarrow{\rm C1} [\,\boldsymbol{U}\,|\,\boldsymbol{b}\,] \xrightarrow{\rm C2} [\,\boldsymbol{U}\,|\,\boldsymbol{y}\,] \xrightarrow{\rm C3} [\,\boldsymbol{U}\,|\,\boldsymbol{x}\,]\,. \qquad \text{(C.1)}$$

The extension to an arbitrary number of right-hand sides (to the right of the vertical line) is trivial. For example, the inverse $\boldsymbol{N}^{-1}$ can be obtained by inserting the identity matrix for $\boldsymbol{b}$. We also note that C1 and C2 are mathematically equivalent to pre-multiplying the matrix and right-hand side, respectively, by $(\boldsymbol{U}')^{-1}$. Performing C2 on the unit vector $\boldsymbol{e}_i = [0, 0, \dots, 1, \dots, 0]'$ (with 1 in position i) thus produces $\tilde{\boldsymbol{e}}_i = (\boldsymbol{U}')^{-1}\boldsymbol{e}_i$ such that $\tilde{\boldsymbol{e}}_i'\tilde{\boldsymbol{e}}_j = \boldsymbol{e}_i'\boldsymbol{U}^{-1}(\boldsymbol{U}')^{-1}\boldsymbol{e}_j = \boldsymbol{e}_i'\boldsymbol{N}^{-1}\boldsymbol{e}_j = (\boldsymbol{N}^{-1})_{ij}$. Selected elements or sub-matrices of $\boldsymbol{N}^{-1}$ can thus be obtained without computing the full matrix.

Algorithm C.1 Returns upper-triangular $\boldsymbol{U}$ such that $\boldsymbol{U}'\boldsymbol{U} = \boldsymbol{N}$, where $\boldsymbol{N} \in \mathbb{R}^{n\times n}$ is symmetric and positive semidefinite. Also computes an estimate d of the rank defect of $\boldsymbol{N}$.

```
 1: $\boldsymbol{U} \leftarrow \boldsymbol{N}$,  $d \leftarrow 0$
 2: for $j = 0$ to $n - 1$ do
 3:     for $i = 0$ to $j$ do
 4:         $s \leftarrow \sum_{k=0}^{i-1} U_{ki}U_{kj}$
 5:         if $i < j$ then
 6:             if $U_{ii} > 0$ then
 7:                 $U_{ij} \leftarrow (U_{ij} - s)/U_{ii}$
 8:             else
 9:                 $U_{ij} \leftarrow 0$
10:             end if
11:         else
12:             if $U_{jj} - s > 0$ then
13:                 $U_{jj} \leftarrow \sqrt{U_{jj} - s}$
14:             else
15:                 $U_{jj} \leftarrow 0$,  $d \leftarrow d + 1$
16:             end if
17:         end if
18:     end for
19:     for $i = j + 1$ to $n - 1$ do
20:         $U_{ij} \leftarrow 0$
21:     end for
22: end for
```

Algorithm C.2 Returns $\boldsymbol{y}$ such that $\boldsymbol{U}'\boldsymbol{y} = \boldsymbol{b}$, where $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ is upper-triangular and $\boldsymbol{b} \in \mathbb{R}^n$.

```
y ← b
for $i = 0$ to $n - 1$ do
    if $U_{ii} > 0$ then
        $y_i \leftarrow (y_i - \sum_{k=0}^{i-1} U_{ki}y_k)/U_{ii}$
    else
        $y_i \leftarrow 0$
    end if
end for
```

The above three steps are accomplished by Algorithms C.1–C.3 for arbitrary symmetric positive definite $\boldsymbol{N}$. (Actually, these algorithms include the non-standard modification discussed below in order to handle semidefinite matrices gracefully.) A few remarks should be made concerning its practical implementation. First, the matrices $\boldsymbol{U}$ and $\boldsymbol{N}$ in C1, $\boldsymbol{y}$ and $\boldsymbol{b}$ in C2, and $\boldsymbol{x}$ and $\boldsymbol{y}$ in C3 can share the same memory if it is acceptable that $\boldsymbol{U}$ overwrites $\boldsymbol{N}$, etc (in-place calculation). Second, since $\boldsymbol{N}$ is symmetric, only the upper-diagonal part of it (N_{ij} for $i \le j$) is used in C1, and similarly for $\boldsymbol{U}$. For large systems one can therefore save roughly half the memory by storing only the upper-diagonal parts of $\boldsymbol{N}$ and $\boldsymbol{U}$, e.g., as one-dimensional arrays. The code in lines 19–21 of Algorithm C.1 is then irrelevant. Third, if $\boldsymbol{N}$ is a 'skyline matrix' with envelope E_j (i.e., $N_{ij} = 0$ for $i < E_j$) then $\boldsymbol{U}$ has the same envelope: the Cholesky decomposition gives no fill-in above the envelope. This allows to store and decompose certain sparse matrices very efficiently, such as the band matrix in Fig. 5.

C.3. Application to semidefinite systems

In several of our applications the normal matrix is however not positive definite, either from a lack of observations (e.g., data gaps in the attitude spline representation) or by design (e.g., for the calibration parameters). Application of the standard Cholesky algorithm in such cases results in an exception which may be non-trivial to handle (for example, by changing

Algorithm C.3 Returns $\boldsymbol{x}$ such that $\boldsymbol{Ux} = \boldsymbol{y}$, where $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ is upper-triangular and $\boldsymbol{y} \in \mathbb{R}^n$.

```
x ← y
for i = n − 1 to 0 do
    if U_ii > 0 then
        $x_i \leftarrow (x_i - \sum_{k=i+1}^{n-1} U_{ik}x_k)/U_{ii}$
    else
        $x_i \leftarrow 0$
    end if
end for
```

the knot sequence of the attitude spline) and which may leave the partially solved system in an undefined state. However, that the Cholesky algorithm fails does not mean that there is no solution to the normal equations: on the contrary, there is an infinitude of solutions and the problem is rather which one to pick. Indeed, in many situations we could in principle accept *any* valid solution; for example, when the null space of the problem is known a priori, and we are prepared to handle the associated non-uniqueness of the solution (cf. Sect. 6.1). For these and several other reasons it is advantageous if the computation can be continued in some sensible way, while of course noting the detected rank deficiency.

A number of methods are available to handle rank-deficient or ill-conditioned least-squares problems. Singular Value Decomposition (SVD; e.g., Björck 1996; Golub & van Loan 1996; Press et al. 2007) is the method often recommended; it provides the unique 'pseudo-solution' with the smallest Euclidean norm. However, SVD is computationally expensive and the pseudo-solution is not necessarily better than any other valid solution to the singular least-squares problem.

By construction, the normal matrix $\boldsymbol{N} = \boldsymbol{A}'\boldsymbol{A}$ is positive semidefinite: $\boldsymbol{x}'\boldsymbol{N}\boldsymbol{x} = \|\boldsymbol{A}\boldsymbol{x}\|^2 \geq 0$ for any $\boldsymbol{x}$ (cf. footnote 7). A modification of the standard Cholesky algorithm allows the decomposition in C1 to be made also for such matrices, although $\boldsymbol{U}$ is no longer unique; similarly C2 and C3 can be modified to produce a valid (if non-unique) solution to the normal equations $\boldsymbol{N}\boldsymbol{x} = \boldsymbol{b}$. For example, the LINPACK routine xCHDC (Dongarra et al. 1979) implements a robust version of the Cholesky algorithm, using complete pivoting (i.e., a simultaneous permutation of the rows and columns of $\boldsymbol{N}$) to generate the unique square root with non-zero elements only in the first r rows, if $r < n$ is the rank of the matrix (for a detailed analysis, see Higham 2002). Other modifications of the Cholesky algorithm (e.g., Schnabel & Eskow 1999) also uses pivoting.

Permuting the rows and/or columns of the matrix is highly undesirable in the present applications. For example, when applied to a band matrix (such as Fig. 5) it likely destroys the band structure, and in general prevents the envelope-based storing of the sparse matrices $\boldsymbol{N}$ and $\boldsymbol{U}$ outlined above. While pivoting is never needed for the Cholesky factorization of a positive definite matrix, it is thought to be an essential ingredient in modified algorithms aimed at more general symmetric matrices. A simple modification of the Cholesky algorithm, which makes it applicable to the semidefinite case without pivoting was described by Lawson & Hanson (1974, Eq. 19.5); see also Golub & van Loan (1996, Eq. 4.2.11), who however warn that "it may be preferable to incorporate pivoting". Nevertheless we have implemented the corresponding modifications in Algorithms C.1–C.3 without pivoting, and find that they work quite well in our applications. Algorithm C.1 includes a rough estimation of the rank defect $d = n - r$.

The numerical accuracy of the decomposition in Algorithm C.1 was tested in MATLAB for a range of rank-deficient random positive semidefinite matrices $\boldsymbol{N}$ (patterned after Higham 1990) by computing the quantity $\rho_N = \|\boldsymbol{N} - \boldsymbol{U}'\boldsymbol{U}\|_\text{F}/(u\|\boldsymbol{N}\|_\text{F})$ as a measure of the relative error in units of the floating point precision. Here $\|\boldsymbol{N}\|_\text{F} = [\text{trace}(\boldsymbol{N}'\boldsymbol{N})]^{1/2}$ is the Frobenius norm of $\boldsymbol{N}$, and $u = 2^{-52}$ is the unit roundoff (Golub & van Loan 1996) of the double precision floating point arithmetic used. We find that the present algorithm, without pivoting, performs almost as well as LINPACK's xCHDC with complete pivoting, as judged from the statistics of our ρ_N compared with the corresponding quantity ρ_k reported by Higham. However, C.1 is much less useful as a rank-revealing algorithm – the estimated rank defect is often much too small.

Similarly, in order to check the numerical validity of the solution to the rank-deficient normal equations computed by C.1–C.3, we made the following experiments, using the same matrices as above. For random vectors $\boldsymbol{x}$ we first computed $\boldsymbol{b} = \boldsymbol{N}\boldsymbol{x}$, then used C.1–C.3 to recover a solution $\tilde{\boldsymbol{x}}$ (usually very different from $\boldsymbol{x}$). Finally we computed $\rho_b = \|\boldsymbol{b} - \boldsymbol{N}\tilde{\boldsymbol{x}}\|_\text{F}/(u\|\boldsymbol{b}\|_\text{F})$. We find that our algorithm performs almost as well as MATLAB's basic solution `xtilde = N\b`, and much better than the minimum norm solution `xtilde = pinv(N)*b` (with default tolerance). For example, the 99th percentile of ρ_b was $\sim 10^4$ for C.1–C.3, $\sim 10^3$ for MATLAB's backslash (\) operator, and $\sim 10^{11}$ (!) when using `pinv`.

We conclude from these limited experiments that the present version of the Cholesky algorithm is a useful, simple and efficient substitute for much more sophisticated algorithms applicable in the semidefinite case. Since it does not use pivoting, it preserves the envelope of the matrix and is therefore especially suited for banded matrices and envelope-based sparse matrix methods. It provides an estimate of the rank of the matrix, which however is rather unreliable. In the positive definite case the algorithm is equivalent to the standard Cholesky method.

Appendix D: Complexities beyond the basic modelling

Section 3 describes a set of baseline models for the sources, attitude, and geometrical instrument, which are believed to be realistic enough to serve as an acceptable first-order approximation of the actual data for primary sources. Due to the many complexities of the real satellite and its operation, as well as the physical environment in space, there are however many additional effects that may affect the astrometric results at the μas level, and which have to be considered in the final modelling. In this Appendix we discuss some of the known effects that will be addressed in future versions of the astrometric solution.

D.1. Chromaticity

Although the Gaia telescopes are all-reflective, with no refractive elements in the optical paths to the astrometric field, they are nevertheless not completely achromatic. In the presence of odd wavefront errors, such as coma, the centroids of the optical images do in fact depend on the wavelength, and hence on the spectral energy distributions of the observed sources.[25] For the typical wavefront errors expected in the astrometric field of

[25] 'Centroid' should here not be understood as the centre of gravity of the optical image; rather, it is a non-trivial function of the light distribution, similar to the estimation of the image location κ in Sect. 3.5.

Gaia (about 50 nm RMS), the AL centroid shift from an early-type star to a late spectral type could amount to several mas. This systematic effect, known as chromaticity, can therefore be many times larger than the photon-statistical uncertainty of the estimated image location (cf. Table 1). It is thus essential to have a very good calibration of the chromaticity, for which it is necessary to know the spectral energy distribution of every observed source. This is obtained from the photometric observations in the BP and RP fields (see Fig. 3).

Chromaticity is eliminated in the CCD signal analysis by using a Line Spread Function (LSF), $L(x)$ in Eq. (23), which is correspondingly shifted depending on the (known) spectrum of the source. The resulting AL pixel coordinate κ is therefore in principle achromatic, and the effect need not be further considered in the astrometric solution. However, as mentioned in Sect. 3.4, it is envisaged to have diagnostic colour-dependent terms in the geometric instrument model of the astrometric solution. These calibration parameters should obtain negligible values in the solution if the chromaticity has been correctly accounted for in the calibration of $L(x)$. Conversely, non-zero values can be used to improve the LSF calibration in the next processing cycle.

D.2. Charge Transfer Inefficiency of the CCDs

The CCD signal model in Eq. (23) assumes a perfectly linear detector, which is not exactly the case for the real detectors and especially not in the presence of radiation damage on the CCDs. Traps in the silicon substrate, produced by particle radiation in the space environment, will capture charges during the TDI operation of the CCDs, and release them with delays ranging from a fraction of the TDI period to minutes. The charge capture and release processes introduce a number of phenomena, collectively referred to as Charge Transfer Inefficiency (CTI = 1 − CTE, where CTE is the Charge Transfer Efficiency; Janesick 2001). CTI will affect all kinds of observations in Gaia (astrometric, photometric, spectroscopic). The most important phenomena for the astrometric observations are the apparent charge loss (because part of the charges are released outside the observed window) and centroid shift (because some charges are released with a delay of one or a few TDI periods). These and more general effects of the radiation damage are the subject of extensive theoretical and experimental studies within the Gaia community (e.g., Seabroke et al. 2009; Prod'Homme et al. 2010; Prod'homme et al. 2011). The adopted method to handle these effects in the Gaia data processing is by means of forward modelling using a so-called Charge Distortion Model (CDM; Short et al. 2010). In the context of the CCD signal model of Sect. 3.5, the CDM may be represented by the (non-linear) operator D:

$$\lambda_k \equiv \mathrm{E}\left(N_k\right) = D\left[\{\lambda^0_{k'}\}_{k'\le k}\,\middle|\,\Psi\right] , \tag{D.1}$$

where λ^0_k is the signal model at sample k in the absence of radiation-damage effects, e.g., according to Eq. (23). The variable Ψ represents the state of the CCD, e.g., in terms of how much radiation damage it has suffered. Note that the expected value of sample k depends on the CCD illumination history up to and including sample k, which is expressed by the CDM taking as input the (undamaged) value not only for sample k but also for the preceding samples ($k' \le k$).[26] One of the methods employed for mitigating CTI effects in Gaia is through a periodic (e.g., once per second) electronic injection of charges in a few consecutive TDI lines. As the lines of charges travel across the CCD, most of the harmful traps are temporarily filled, thus reducing the CTI of subsequent charge transfers (Laborie et al. 2007). The method has the additional benefit of periodically resetting the illumination history of the pixels, so that in Eq. (D.1) only the samples since the previous charge injection need to be considered (Short et al. 2010).

The CDM depends on a moderate number of parameters that will be estimated in parallel with the LSF (or PSF) calibration prior to the astrometric solution (the 'Instrument response parameters' in Fig. 1). *In principle*, the subsequently estimated image location κ should then not only be achromatic, as discussed in Appendix D.1, but also free of CTI effects, so that the astrometric solution can use a purely geometric instrument model, as required by the primary source model. Although this is obviously a highly idealised condition, it it nevertheless what the final data analysis must aim to achieve.

For the simple image of a primary source, the centroid shift due to the CTI depends mainly on the magnitude of the source, the time since the previous charge injection, and the accumulated radiation dose experienced by the CCD (Prod'homme et al. 2011). It is expected that imperfections in the CDM calibration will likewise show up in systematic shifts depending primarily on these (known) quantities, and can be represented by a set of diagnostic calibration functions in the generic calibration model of Sect. 3.4. Non-negligible values of the diagnostic parameters in the astrometric solution indicate that the CDM is not doing its job properly, and they can then be used to improve the model. The reader is referred to the cited papers for quantitative information on the expected level of CTI effects in Gaia data, the effectiveness of different mitigation strategies, and the associated performance degradations.

D.3. Effects of the finite CCD integration time

Up until now we have regarded the astrometric observations of Gaia as instantaneous measurements of the crossings of the source images over the fiducial 'observation line' (Fig. 4) at the centre of the CCD (or of the gated portion of the CCD) in the AL direction. In reality, due to the finite integration time (T) of the CCD observations, any measurement clocked into the CCD readout register at time t actually depends on the average attitude and source position over the preceding integration time interval, $[t-T, t]$. The time delay is, to first order, taken into account by associating the measurement with the observation time $t - T/2$ (Sect. 3.5). Following Bastian & Biermann (2005) we should, more precisely, assume that the observed location κ of the image centre in the pixel stream is a weighted mean of the *instantaneous* location $\kappa_*(t)$ of the optical image relative to the charge image during the preceding integration interval:

$$\kappa = \frac{\int_0^T e(\tau)\kappa_*(t-\tau)\,\mathrm{d}\tau}{\int_0^T e(\tau)\,\mathrm{d}\tau} , \tag{D.2}$$

where $e(\tau)$ is the (nominally flat) 'exposure function' for look-back time τ, i.e., the rate at which electrons are produced, and

For the sake of illustration, the centroid could for example be the point obtained by fitting a Gaussian PSF to the image.

[26] A further complication is that most AF observations are binned in the AC direction before read-out, while the CTI effects operate independently on each pixel column. Thus the CDM should ideally be applied to the two-dimensional charge image, and the distorted charge image then binned for comparison with the one-dimensional data. This requires the use of a PSF replacing the LSF in Eq. (23).

transported to the read-out register, for constant illumination. The instantaneous location is given by

$$\kappa_*(t) = s\left[\eta_*(t) - \eta_{fng}\right] + k(t)\,, \tag{D.3}$$

where s is the local scale factor (pixels per radian), $\eta_*(t)$ the instantaneous AL field angle of the optical image centre, and η_{fgn} the AL coordinate of the fiducial observation line for the appropriate AC coordinate, field of view, etc. The function $k(t)$ is the inverse of t_k, relating the time coordinate to the TDI period index k. For the present discussion $k(t)$ is regarded as a continuous function, ignoring the step-wise transportation of the charge image in TDI mode.[27]

Recalling that $\eta_*(t)$ is decreasing with time (cf. Fig. 3), while $k(t)$ is increasing, it is seen that $\kappa_*(t)$ remains approximately constant throughout the integration. Let us denote by t_c the exact time when the centre of the optical image crosses the fiducial observation line, so that $\eta_*(t_c) = \eta_{fng}$, and let $\kappa_c = k(t_c)$ be the corresponding pixel coordinate. If the speed of the optical image exactly matches the speed of the charge image, or $s\dot{\eta}_* + \dot{k} = 0$, it is seen that $\kappa_*(t)$ is indeed constant and equal to κ_c. Let us now consider what happens if there is a mismatch between the speeds. This could be caused by (i) a deviation in the local scale value s due to optical distortion; (ii) a non-nominal local scan rate; and (iii) that the object itself has significant motion (e.g., an asteroid). Assuming that $\dot{k}$ is constant and adopting a Taylor series expansion for the AL field angle over the exposure time, we have

$$k(t) = \kappa_c + (t - t_c)\dot{k}\,, \tag{D.4}$$

$$\eta_*(t) = \eta_{fng} + (t - t_c)\dot{\eta}_* + \frac{1}{2}(t - t_c)^2\ddot{\eta}_* + \cdots\,. \tag{D.5}$$

Inserting in Eq. (D.3), and assuming that s is constant across the section of the CCD in question, we obtain by means of Eq. (D.2)

$$\kappa = \kappa_c + (s\dot{\eta}_* + \dot{k})e_1 + \frac{1}{2}s\ddot{\eta}_* e_2 + \cdots\,, \tag{D.6}$$

where

$$e_m = \frac{\int_0^T e(\tau)(T/2 - \tau)^m\,\mathrm{d}\tau}{\int_0^T e(\tau)\,\mathrm{d}\tau}\,, \quad m = 1, 2, \ldots \tag{D.7}$$

are the normalized moments of $e(\tau)$ relative to the exposure midtime at $\tau = T/2$. For a constant and matching image speed we recover $\kappa = \kappa_c$ as expected. In the general case of imperfect speed matching and non-constant scan rate there is a difference which should be taken into account in the astrometric solution. If we assume that s is known, the speed mismatch $s\dot{\eta}_* + \dot{k}$ can be computed for every observation, and the factor e_1 can then be estimated as an instrument calibration parameter using the generic calibration model in Sect. 3.4. e_1 will depend (at least) on the CCD and gate used; but due to the accumulating radiation damage it is likely to evolve with time and could possibly have a magnitude-dependent component as well. In the next (quadratic) term we may know $\ddot{\eta}_*$ (from the attitude) and $e_2 \simeq T^2/12$ (for constant exposure function) to sufficient accuracy that it might be applied as a correction to the observed κ. However, since most observations are ungated (giving maximum T), it may be better to adopt the uncorrected κ for this maximum T as defining the *effective* attitude,[28] and only correct the gated observations for the difference in e_2; hence they, too, will refer to the effective attitude. A complication is that $\ddot{\eta}_*$ in Eq. (D.6) should be computed from (unknown) *physical* attitude, but to first order it can be obtained from the effective attitude. Attitude irregularities on time scales shorter than T add further complications (see Appendix D.4), but it is unlikely that higher-order terms in Eq. (D.6) can profitably be accounted for.

D.4. Attitude irregularities

The basic attitude model described in Sect. 3.3 uses a spline representation which is (normally) continuous in the angles specifying the instantaneous orientation of the instrument, as well as in the first $M - 2$ derivatives of the angles, where M is the order of the spline (typically cubic splines are used, for which $M = 4$). The actual (physical) attitude is much more complex and in particular there may be discontinuities and irregularities on time scales that are too short to be adequately represented by a spline with the knot separations considered in the basic model. Low-frequency perturbations ($\lesssim 0.01$ Hz) are of no concern here, as they can be perfectly represented by splines. The most important contributors to perturbations at higher frequencies are thruster noise in the micro-propulsion system used for the attitude control; the discrete and partially stochastic nature of the control system (for example that the commanded thrusts are updated once per second); micrometeoroid impacts on the satellite; and various dynamical effects such as fuel sloshing and structural vibrations.

The high-frequency perturbations due to the thruster noise and control system generate angular jitter of the physical attitude that has a significant amplitude relative to the astrometric accuracies ultimately aimed at, but still small in comparison with the AL pixel size ($\simeq 59$ mas). Thanks to the TDI integration this high-frequency jitter is largely removed from the effective attitude by the averaging over the exposure time T. As a result, the shortest knot interval needed to accurately represent the effective attitude is also of the order of the exposure time, or about 5 s. The optimum knot interval may be longer, depending on the number and magnitudes of the primary sources available for the attitude determination, and on the actual level of perturbations.

The expected frequency of micrometeoroid impacts of various sizes can be predicted from the known velocity and mass spectrum of interplanetary particles. Each impact produces a quasi-instantaneous change in the angular velocity of the satellite, while the attitude angles are continuous across the impact time. It is expected that a few hundred impacts will occur every year producing a change in the AL angular velocity exceeding 1 mas s^{-1} (which should be easily detectable), with the frequency roughly inversely proportional to the minimum change considered. Discontinuities in the *physical* attitude rate can be represented in the spline model by inserting multiple knots at the estimated times of impact, t_i (Sect. 5.2.6). However, the *effective* attitude will instead see a linear change in the attitude rates over an interval equal to the exposure time T, centred on t_i, which requires that multiple knots are inserted both at $t_i - T/2$ and $t_i + T/2$. (This treatment becomes more problematic in connection with

[27] The corresponding expression in Bastian & Biermann (2005) is their Eq. (6), in which $k(t)$ is the (integer) index of the last TDI clock stroke before time t. Thus their $\kappa_*(t)$ oscillates with an amplitude of ±0.5 unit for every TDI period. The continuous approximation adopted in our Eq. (D.3) is acceptable since we are always considering integrals covering a whole number of oscillations.

[28] The effective attitude is then the physical attitude convolved with the (average) exposure function for maximum T. It corresponds closely to the 'astrometric attitude' introduced by Bastian & Biermann (2005).

gated observations, when T is non-nominal.) Impacts will be detected by inspecting the observation residuals in connection with the attitude updating (Sect. 5.2.5).

The detailed re-examination of the Hipparcos attitude by van Leeuwen (2005, 2007) revealed numerous discontinuities of the along-scan attitude angle (scan phase) of several tens of mas. A large fraction of them could be linked to the beginning or end of eclipses experienced by Hipparcos in its highly elliptic orbit around the Earth. A likely cause is thermal re-adjustment of the solar-panel hinges, following a sudden change in temperature. As there is no change in the net angular momentum, but only a re-distribution of inertia, the attitude rates are the same before and after a discontinuity. For Gaia it is estimated that such 'clanks' will be negligible, but the attitude processing should nevertheless be capable of identifying instances, should they occur, and to take appropriate measures. Due to the finite CCD integration time, the apparent effects of a clank will be two discontinuities in the attitude rates, equal but of opposite sign, and separated by the integration time T. Again, this can be handled by suitable modification of the knot sequence of the attitude spline. Like the micrometeoroid impacts, clanks will be detected during the attitude updating by means of the characteristic patterns that they generate in the observation residuals versus time.

Appendix E: Tables of acronyms and variables

Table E.1 is a list of acronyms used in the paper. Table E.2 lists the most important variables, with a short description and a reference to where it is introduced or explained.

Table E.1. List of acronyms

Acronym	Description
AC	ACross scan direction (Fig. 3)
ACF	ACross scan coordinate in the Following FoV
ACP	ACross scan coordinate in the Preceding FoV
AF	Astrometric Field
AGIS	Astrometric Global Iterative Solution
AGN	Active Galactic Nucleus
AL	ALong scan direction (Fig. 3)
ASI	Accelerated Simple Iteration
BAM	Basic-Angle Monitor (Fig. 3)
BCRS	Barycentric Celestial Reference System
BP	Blue Photometer (Fig. 3)
CCD	Charge-Coupled Device
CDM	Charge Distortion Model
CFS	Calibration Faint Star
CG	Conjugate Gradient
CPU	Central Processing Unit
CoMRS	Centre of Mass Reference System
CTE	Charge Transfer Efficiency (of a CCD)
CTI	Charge Transfer Inefficiency (of a CCD)
CU2	DPAC Coordination Unit 2, 'Data Simulations'
CU3	DPAC Coordination Unit 3, 'Core Processing'
DPAC	Data Processing and Analysis Consortium
EADS	European Aeronautic Defence and Space company
ESA	European Space Agency
ESAC	European Space Astronomy Centre
FFoV	Following Field of View (Fig. 2)
FPA	Focal Plane Assembly (Fig. 3)
FoV	Field of View
GCRS	Geocentric Celestial Reference System
HEALPix	Hierarchical Equal-Area iso-Latitude Pixelisation
IAU	International Astronomical Union
ICRS	International Celestial Reference System
LSF	Line Spread Function
NSL	Nominal Scanning Law
PFoV	Preceding Field of View (Fig. 2)
PPN	Parametrised Post-Newtonian (relativity formalism)
PSF	Point Spread Function
RMS	Root-Mean-Square
RP	Red Photometer
RSE	Robust Scatter Estimate (footnote 18)
RVS	Radial Velocity Spectrometer (Fig. 3)
SI	Simple Iteration
SM	Sky Mapper
SRS	Scanning Reference System (Fig. 2)
SVD	Singular Value Decomposition
TB	TeraByte
TCB	Barycentric Coordinate Time
TDI	Time-Delayed Integration (CCD operation mode)
VLBI	Very Long Baseline Interferometry
WFS	WaveFront Sensor (Fig. 3)
XML	eXtensible Markup Language

Table E.2. List of mathematical variables

Var.	Description	Ref.
A_u	the astronomical unit	Sect. 3.2
$\boldsymbol{A}$	attitude matrix	Eq. (8)
$\boldsymbol{a}$	galactocentric acceleration vector	Sect. 6.1.4
$\boldsymbol{a}$	attitude parameters	Sect. 3.3
$\mathbf{a}_n$	quaternion coefficient for $B_n(t)$	Eq. (10)
$B_n(t)$	B-spline function	Sect. B.1
$\boldsymbol{B}$	iteration matrix	Sect. 4.5
$\boldsymbol{b}$	right-hand side of normal equations	Eq. (30)
$\boldsymbol{b}_G(t)$	barycentric coordinate of Gaia at time t	Eq. (4)
C	celestial reference system	Sect. 3.1
$\boldsymbol{C}$	constraint matrix for the calibration parameters	Eq. (22)
$\boldsymbol{c}$	calibration parameters	Sect. 3.4
D	non-linear Charge Distortion Model	Eq. (D.1)
$\boldsymbol{d}$	update vector	Sect. 4.5
E	expectation operator	Eq. (23)
$e(t)$	CCD exposure function	Appendix D.3
$\boldsymbol{e}$	error vector	Sect. 4.5
f	field index (±1 for preceding/following)	Eq. (13)
$\boldsymbol{f}$	detector coordinates	Eq. (1)
$\boldsymbol{f}_\mathrm{P}$, $\boldsymbol{f}_\mathrm{F}$	preceding, following viewing direction	Fig. 2
G	Gaia broadband magnitude	Sect. 3.4
g	gate index	Eq. (13)
$\boldsymbol{g}$	global parameters	Sect. 3.4
$\boldsymbol{h}$	auxiliary data, e.g., ephemerides	Eq. (7)
$\boldsymbol{I}$	the identity matrix	Sect. 4.5
i	subscript denoting some source	Sect. 3.2
j	'short' calibration time interval index	Eq. (15)
K	no. of TDI periods for CCD integration	Sect. 3.5
$\boldsymbol{K}$	preconditioner matrix	Sect. 4.5
k	TDI index in CCD pixel stream	Sect. 3.5
k	'long' calibration time interval index	Eq. (15)
k	iteration index	Eq. (40)
L	Line Spread Function	Eq. (23)
L^*_r	shifted Legendre polynomial of degree r	Eq. (15)
l	subscript denoting some observation	Sect. 3.6
ℓ	left index in a knot sequence	Sect. B.2
M	spline order – $M = 4$ for cubic spline	App. B
N	number of degrees of freedom for a spline	App. B
$\boldsymbol{N}$	normal equations matrix	Eq. (30)
n	attitude parameter index	Eq. (10)
n	CCD index	Eq. (14)
n	dimension of the normal matrix $\boldsymbol{N}$	Eq. (35)
$\boldsymbol{n}$	nuisance parameters	Eq. (1)
$\boldsymbol{p}_i$	normal-triad component	Eq. (5)
Q	objective function to be minimized	Eq. (24)
$\mathbf{q}$	attitude quaternion	Eq. (10)
$\boldsymbol{q}_i$	normal-triad component	Eq. (5)
r	degree of the small-scale calibration polynomial	Eq. (15)
$\boldsymbol{r}_i$	normal-triad component	Eq. (5)
R_l	residual of observation l	Sect. 3.6
S	Scanning Reference System	Sect. 3.1
s	local scale factor	Eq. (D.3)
$\boldsymbol{s}$	astrometric ('source') parameters	Sect. 3.2
$\boldsymbol{s}_i$	the astrometric parameters for source i	Sect. 3.2
T	light integration (exposure) time on CCD	Sect. 3.4, D.3
t_k	time for sample k	Sect. 3.5
t_l	time for observation l	Sect. 3.5
t_ep	reference epoch for astrometric parameters	Sect. 3.2
t_fr	reference epoch for frame rotator	Sect. 6.1
t_P	reference epoch for non-rotating ICRS sources	Sect. 6.1.4

Table E.2. List of mathematical variables (continued)

Var.	Description	Ref.
$t_{\rm PM}$	reference epoch for moving ICRS sources	Sect. 6.1.4
U	relegation factor for primary-star selection	Sect. 6.2.2
$\boldsymbol{u}_i$	proper direction to source i	Sect. 3.2
$\bar{\boldsymbol{u}}_i$	geometric direction to source i	Sect. 3.2
$\boldsymbol{V}$	nullspace of the normal matrix	Sect. 4.5
$\boldsymbol{W}_i$	weight matrix for source i	Eq. (57)
w_l	downweighting factor for observation l	Sect. 3.6, 5.1.2
W_l	statistical weight of observation l	Sect. 3.6
$\boldsymbol{x}$	differential parameter vector	Sect. 4.3
$[\boldsymbol{X}\ \boldsymbol{Y}\ \boldsymbol{Z}]$	celestial reference system (ICRS or CoMRS)	Sect. 3.1
$[\boldsymbol{x}\ \boldsymbol{y}\ \boldsymbol{z}]$	Scanning Reference System (SRS)	Fig. 2
α	flux (image parameter)	Eq. (23)
α_i	right ascension of source i at $t_{\rm ep}$	Sect. 3.2
β	background level (image parameter)	Eq. (23)
$\Gamma_{\rm c}$	(conventional) basic angle	Eq. (13)
γ	PPN curvature parameter	Eq. (88)
δ_i	declination of source i at $t_{\rm ep}$	Sect. 3.2
ϵ_a	excess attitude noise	Eq. (28)
ϵ_i	excess source noise (for source i)	Eq. (28)
ϵ_l	excess noise in observation l	Eq. (28)
$\boldsymbol{\varepsilon}$	orientation correction (frame rotator)	Sect. 6.1
ζ	across-scan (AC) field angle	Eq. (12)
η	along-scan (AL) field angle	Eq. (13)
κ	along-scan (AL) pixel coordinate (image location)	Eq. (23)
λ	regularization parameter for the attitude update	Eq. (81)
λ_k	intensity for CCD sample values	Eq. (23)
μ	across-scan (AC) pixel coordinate (image location)	Sect. 3.4
$\mu_{\alpha*i}$	proper motion in α ($\times\cos\delta_i$) for source i	Sect. 3.2
$\mu_{\delta i}$	proper motion in δ for source i	Sect. 3.2
μ_{ri}	radial proper motion for source i	Sect. 3.2
$\boldsymbol{\mu}_0$	proper motion due to galactocentric acceleration	Sect. 6.1.4
ν	number of degrees of freedom	Sect. 5.1.2
ϖ_i	parallax for source i	Sect. 3.2
ϱ	normalized RSE error of astrometric parameters	Sect. 7.2.2
σ_l	formal standard uncertainty for observation l	Sect. 3.6
τ_n	attitude spline knot n	Sect. B.1
φ	along-scan instrument angle	Eq. (12)
$\boldsymbol{\omega}$	spin correction (frame rotator)	Sect. 6.1

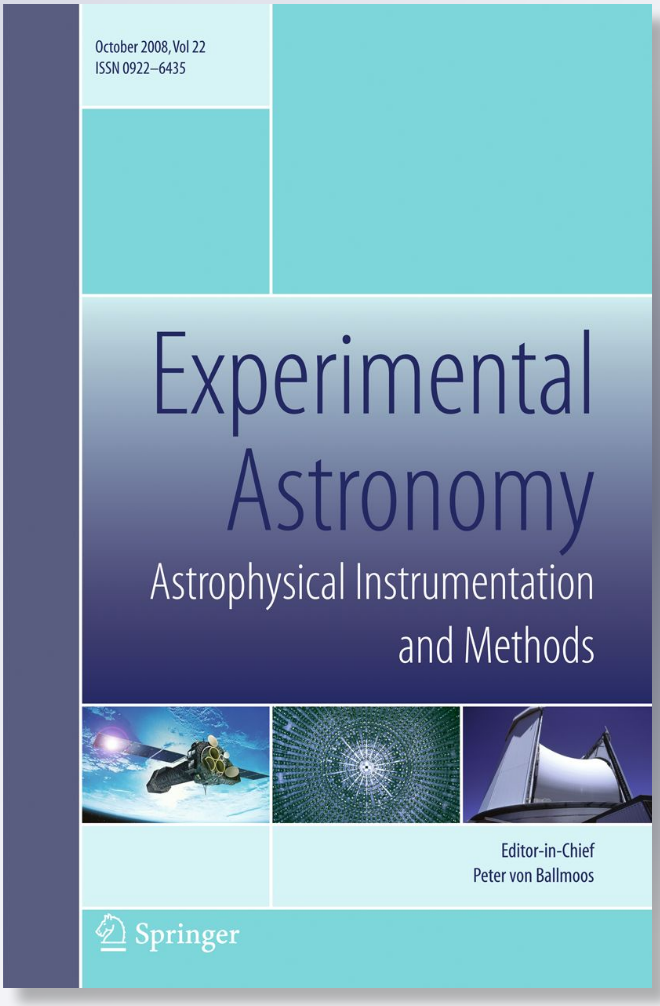
Using Java for distributed computing in the Gaia satellite data processing

William O'Mullane, Xavier Luri, Paul Parsons, Uwe Lammers, John Hoar & Jose Hernandez

Experimental Astronomy
Astrophysical Instrumentation
and Methods

ISSN 0922-6435

Exp Astron
DOI 10.1007/
s10686-011-9241-6



October 2008, Vol 22
ISSN 0922–6435

Experimental Astronomy

Astrophysical Instrumentation and Methods

Editor-in-Chief
Peter von Ballmoos

Springer

Springer

Exp Astron
DOI 10.1007/s10686-011-9241-6

SHORT COMMUNICATION

Using Java for distributed computing in the Gaia satellite data processing

William O'Mullane · Xavier Luri · Paul Parsons · Uwe Lammers · John Hoar · Jose Hernandez

Received: 27 October 2010 / Accepted: 4 July 2011

Abstract In recent years Java has matured to a stable easy-to-use language with the flexibility of an interpreter (for reflection etc.) but the performance and type checking of a compiled language. When we started using Java for astronomical applications around 1999 they were the first of their kind in astronomy. Now a great deal of astronomy software is written in Java as are many business applications. We discuss the current environment and trends concerning the language and present an actual example of scientific use of Java for high-performance distributed computing: ESA's mission Gaia. The Gaia scanning satellite will perform a galactic census of about 1,000 million objects in our galaxy. The Gaia community has chosen to write its processing software in Java. We explore the manifold reasons for choosing Java for this large science collaboration. Gaia processing is numerically complex but highly distributable, some parts being embarrassingly parallel. We describe the Gaia processing architecture and its realisation in Java. We delve into the astrometric solution which is the most advanced and most complex part of the processing. The Gaia simulator is also written in Java and is the most mature code in the system. This has been successfully running since about 2005 on the

W. O'Mullane (✉) · U. Lammers · J. Hoar · J. Hernandez
Space Robotic Exploration, European Space Astronomy Centre, Madrid, Spain
e-mail: womullan@sciops.esa.int

X. Luri
ICC-UB/IEEC, Departament d'Astronomia i Meteorologia,
Universitat of Barcelona, Barcelona, Spain

P. Parsons
The Server Labs, Madrid, Spain

Published online: 29 July 2011

supercomputer “Marenostrum” in Barcelona. We relate experiences of using Java on a large shared machine. Finally we discuss Java, including some of its problems, for scientific computing.

Keywords Distributed computing · Java · Astrometry · Cloud computing · Mathematics

1 Introduction

1.1 The Gaia mission

The Gaia satellite is destined for the Lagrange point L2 early in 2013 after launch from French Guiana aboard a Soyuz Fregat Rocket. Gaia is the European Space Agency’s (ESA) sixth cornerstone mission. Its goal to make a phase space map of our galaxy. Spinning around its own axis in a Lissajous orbit around L2 for five years Gaia will continually scan the sky observing more than one thousand million (10^9) celestial sources, on average eighty times each. The scientific goals of the experiment are manifold and covered in detail in [12], the data will help key research in the composition and formation of our galaxy.

Gaia contains two astrometric telescopes at a fixed angle of 106.5° as well as a radial velocity spectrograph and two photometers. The astrometric design allows true parallaxes (distances to stars) to be obtained [7] after careful data processing. A least-squares fitting scheme named the Astrometric Global Iterative Solution (AGIS) (Lindegren et al. 2011, in preparation) is currently foreseen to perform the astrometric data reduction. Equally involved processing is required for photometry and spectroscopy.

Gaia processing software is being written by the Gaia Data Processing and Analysis Consortium (DPAC). DPAC is a pan-European federation of institutes comprising of over four hundred astronomers, physicists and programmers. The consortium is led by eleven members of the community who form the DPAC Executive (DPACE). DPAC not only has the responsibility to write the processing software but also to run and maintain it until the final Gaia catalogue is produced around the year 2020. The software will be run in six data processing centres—each responsible for a different facet of the overall processing. An overview of DPAC and the processing is provided in [11].

1.2 Computing

Simulations of Gaia data have been in production at the University of Barcelona (UB) since about 1998. Some of these simulations require considerable computing power and are discussed in Section 3. Initial AGIS experiments have been conducted using simulation data from UB in the past few years. The AGIS efforts are discussed in Section 2. By nature the data processing must be distributed. If one considers that there are $\approx 10^{12}$ (low resolution cutout) images downlinked from Gaia, at one millisecond per image that is over 30 years of processor time. Massive distribution is the only possibility to deal with this data. At this point the Gaia community has quite some experience with large software and computing efforts, which is not unusual in the science community. What some consider quite unusual is that all Gaia software is written in the Java language. We discuss the choice of Java in Section 4.

2 AGIS–a complex computing problem

The Astrometric Global Iterative Solution (AGIS) provides the rigid framework for all of the Gaia measurements. Gaia is spinning freely and making observations which relate only to other observations made by Gaia. It is an absolute instrument–there is no list of input stars such as there was for Hipparcos [2], the predecessor of Gaia. What this means is that the Gaia data must be reduced in a self-consistent manner such that all individual observations of celestial sources, the model of each source's position and motion, Gaia's own attitude, orbit and velocity must be in harmony. Later the entire system will be aligned with the International Celestial Reference System (ICRS).

The complete mathematical formulation for AGIS is presented in Lindegren et al. (2011, in preparation) while the computational framework is described in O'Mullane et al. (2011, submitted). Here we recap these aspects briefly in Section 2.1 before discussing results (Section 2.2) and performance/manpower trade offs (Section 2.3).

2.1 AGIS overview

AGIS is a block-iterative solution for the Gaia astrometry. It consists of at least four blocks which may be calculated independently. Each block is however dependent on the result of the other blocks. We term one step through all blocks an outer iteration, although the term outer is frequently dropped. This is simplistically formulated in the following equations:

$$S = A + G + C \tag{1}$$

$$A = S + G + C \tag{2}$$

$$G = S + A + C \tag{3}$$

$$C = S + A + G \tag{4}$$

which are discussed further below.

The vast majority of Gaia sources (1) may be described by a six parameter astrometric model. These parameters and their derivation are fairly standard in modern astrometry [8] and describe the position and motion of the source in three dimensions. The parameters are:

α	azimuthal angle of object a.k.a. right ascension
δ	angular distance from the celestial equator (north or south) a.k.a. declination
π	annual parallax, the apparent shift in angular position when viewed from opposite sides of the sun
$\mu_{\alpha*}$	$\mu_{\alpha*} = \mu_\alpha cos(\delta)$ proper motion in α direction
μ_δ	proper motion in δ direction
μ_r	radial velocity, motion in the line of sight direction

From the ≈80 observations of each source made by Gaia a least-squares fitting of the observations to the model may be made. The fit must include several intricacies namely:

- the orientation of Gaia in space (or attitude, (2))
- the path of light through the instrument (or calibration, (4))
- global parameters such as relativistic numbers (Section 3)
- light bending according to general relativity (see [6])

Current best estimates for attitude, calibration and global parameters are used for any given fit of the source. This fit must be performed on all sources but to make the problem tractable a subset of sources may be treated first. Around 10^8 sources are needed to make the global reference frame but it could be as many as $5 \cdot 10^8$. Once this solution has converged the source calculation (using the final attitude etc.) may be performed on all remaining sources. This is an important efficiency improvement. A direct solution for a large number of sources has been shown to be intractable [1].

The application of the source equation (known as source update) requires the gathering of all observation of the source and the current attitude, calibration and global parameters. The application of the relativistic model requires precise ephemerides of the planets and of the satellite. Our current approach is to group observations together on disk for easy loading, since there is no cross talk between the source equations i.e. all sources may be processed in parallel if we wish. The other data required is sufficiently small (order of a few hundred MB) that it may be loaded once in memory and used to process multiple sources, typically batches of several thousand sources per job on a processor.

Attitude and calibration on the other hand require all observations in given time periods. We do not need to hold on to the observations but may add their contributions to partial equations. Hence as each source is updated the updated parameters and the observations are used to update the partial solutions for the attitude and calibration. The system has been run with up to 100 computing nodes and 1,400 Java threads. We constantly encounter bottlenecks usually in the calibration and attitude processes for the obvious reason that they need to see all sources. The processing of several thousand sources in a job makes the communication more efficient—we send fewer large messages rather than millions of small ones.

2.2 Results

AGIS has been running at ESAC since early 2006 with increasing numbers of sources and observations. This is discussed briefly in O'Mullane et al. (2011, submitted) from which we reproduce Table 1 below. Since the entire system is designed for AGIS and we have few interconnects between processes we did not find it necessary to use any HPC library nor GRID software. Effectively jobs are taken from a *Whiteboard* by numerous *DataTrains* which load the appropriate data and process it writing the results back to a database. The *Whiteboard* may simply be considered as a table in a database where jobs are written an updated. *DataTrain* is a term coined within DPAC for a process which runs through a set of data passing it to a series of algorithms. It provides an abstraction layer between algorithm and data access.

The machine used for the Gaia tests so far has been purchased in installments by ESA. The initial machine for the first tests in 2005 consisted of twelve nodes each with 6 GB of RAM and two processors (3.6 GHz Xeon EM64T). An EMC storage area network (SAN) with 5 TB of disk is attached to the nodes using fibre optic cable and the nodes are also connected via Gigabit Ethernet cards in a local area network. AGIS has also been tested, and shown to run as fast, using a cheap Rack Server Network Attached Storage device of 6 TB. This cluster was upgraded to 18 nodes in 2006 and a further four nodes but with quad core processors were added in 2007. The quad core processors functioned very well for us providing the performance of four processors and in Table 1 the number of processors counts each core as a processor.

This is not very special machine, it consists of standard Dell power edge blades with standard Intel Xeon processors running Red Hat Linux. The blades are housed in a standard Dell rack. There is no special HPC software used, a network of normal Linux machines could work in the same manner.

Table 1 AGIS run times decrease as more processors are added

Date	Observations	Procs	Time (h)	Normalised (obs/h)
2005	1.6×10^7 src	12	3	0.9×10^6
2006	8×10^7	36	5	0.5×10^6
2007	8×10^7	24	3	1.3×10^6
2008	8×10^7	31	1	3.2×10^6
2009	2.6×10^8	50	1.8	2.8×10^6
2010	4×10^9	68	9.5	6.2×10^6

Note that the data volume increased from 2005 to 2006 from 18 months to 5 years, the processor power also increased but the run time went up. This was dramatically improved in 2007. In 2010 we have implemented a new math frame work around Conjugate gradient—this provides a better solution at some cost. The normalised column shows throughput per processor in the system (total observations/processors/hours) e.g. an indication of the real performance

As such this machine perhaps does not merit the term HPC but we will move toward 10 Teraflops in the coming years.

Various different types of astronomical test have been performed to show AGIS produces the correct solution for Gaia. For this paper we are mainly interested in performance however, which is best presented by Table 1. We see that AGIS has been slow in 2007 but finally it was made much faster in 2008 with a prospect of more improvement to come. It should be noted that more functionality has also been introduced each year—a more complete source model, a more complex calibration scheme etc. but the efficiency has been increased. Some speedups are due to profiling while others are due to mathematical methods. Between the Lund group and the ESAC group finally the problem is being tackled from both math and computing side with competence. Early work on AGIS was not so fruitful as it treated the problem as something to just be *run* on a computer rather than a problem for which a system needed to be designed to make it efficient. In that system math and computing techniques have been questioned and varied to arrive at the efficient implementation we have today.

2.3 Performance vs manpower

We know we could write some parts of AGIS to run perhaps two or more times faster than Java. We could get a little more performance by utilising special machines and rewriting our code again for example toward a cell processor hybrid such as the Roadrunner. Speed of code is indeed an issue for Gaia in general but manpower is perhaps a bigger issue—we have a great deal of processing software to write. The estimated manpower in man years for the Gaia processing is around 2,000. Granted this includes operation but development alone pre launch is estimated at 1,000 man years. In the case of the astrometric solution we have good records. The initial working AGIS in ESAC was done in about 180 man days so just over half a man year. It was clear to us that this was only possible in Java—in C or C++ this would have taken far longer. This claim is of course difficult to quantify, as a programming collective we have decades of experience with C and C++ projects and it is our opinion that coding in Java is cleaner and faster. Even in Java the problem is difficult. The previous solution which AGIS replaced was worked on for four years by various people; to be modest lets say only eight man years went into it, but it was more like 12. To date on AGIS we have spent about 15 man years and we estimate to finish AGIS in Java we need another ten man years with probably more after launch. There is also post launch maintenance to consider—we feel Java maintenance will be lower cost than say C, and we have eight years of operations to consider—we have 27 man years in the plan for this.

Of course it is difficult to calculate but we would need far more manpower to manage code in C and more still if we were to customise for a particular computer system (especially as it would become obsolete during our operations). Again just for AGIS we are talking of over 50 man years of effort which in today's monetary terms is around 5 million euro. Even 20% more effort to code in another language would be a considerable sum. We think we can buy the hardware we need to run AGIS for around a million euro. It has been observed on other space programs than manpower typically ends up costing more than machines and apparently we are no different. Our intention is also to buy the most common and cheap processors—these are the the chips Java is usually running fastest on as well because of their ubiquity. Hence a special more super computer oriented machine would probably cost more. Finally there is power consumption if we could make AGIS four times faster we could save 25% on power. Energy costs may well go up but probably not faster than manpower costs since the saving on energy requires spending on manpower to customise code for our "hard to code" problem these will cancel each other out.

3 Gaia simulations—use of shared computing resource

The development of the *Gaia simulator* has been an integral part of the Gaia data processing. This software tool is designed to provide realistic simulations of the Gaia observations, closely mimicking the format and content of the data that Gaia will send to ground. The main purpose of this simulated data has been the feeding of the data processing chain in order to test and validate it, although it has also been used for other purposes, for instance to evaluate some satellite design options and to prepare the mission scientific case.

As mentioned above, the simulator has been running at the University of Barcelona since 1998. Its initial development was closely tied to the first attempts to develop a viable *Global Iterative Solution* but now it has become a mature tool able to simulate a wide variety of celestial objects, physical and instrumental effects and data formats for the multiple data processing modules developed by the Gaia DPAC.

In this section we will first review our experience with the development in Java, the advantages and drawbacks we found, and then discuss a specific (and extreme) example of Java versus C performance found during this development.

3.1 Java, a new kid on the block

When the development of the simulator was started the first choice to be made was the programming language to use. The team undertaking this task was at the time (1998) mainly (and almost exclusively) composed of scientists, with few software engineering expertise. The obvious choice, based on the programming experience of the team, would have been FORTRAN with perhaps C as a second, but somewhat frowned upon, choice. Furthermore, astronomical and numerical libraries were available in FORTRAN, and in most cases also in C, but not in Java.

However, Java was finally chosen for two main reasons:

1. The ongoing *Global Iterative Solution* development contract was going to be implemented in Java by requirement of ESA. Given that the two projects were closely tied at the time using a common language was natural.
2. The advice of professional software engineers from ESA and some members of the team with software engineering background pointed towards the use of an Object Oriented language, strongly advising against FORTRAN for a large collaborative project. This left the choice between C++ and Java.

At the end Java was chosen, but not without frequent second thoughts in the coming years. The widening scientific community that was becoming involved in the project was reticent for some time to adopt the new language, specially given the accumulated experience in FORTRAN and C and the lack of libraries in Java.

However, eleven years later the landscape has much changed, and now Java has been even more widely (almost completely) adopted as a viable programming language for scientific programming in the Gaia simulator community. Furthermore, the project has become large and complex, involving development teams distributed around Europe. The management of the project has at this stage adopted many professional software development tools (UML, Hudson, PMD, Checkstyle, etc.) that would not have been available had a language like FORTRAN been chosen, making the coding much more robust and reliable. Also, like the team developing AGIS and based on previous experiences, they feel that the development has been quicker and more seamless than it would have been using C++. The motives for this feeling are varied, but one of the big reasons is that thanks to the garbage collection implemented in the JVMs memory leaks are less of an issue. Since the simulator is quite intensive in memory usage, not having to worry about this problem is seen as a great advantage.

In short, the initial reticences on the use of Java for scientific programming in Gaia have vanished, to the point that one of the managers of the simulator development has changed the teaching of programming for first year physics students from FORTRAN to Java.

3.2 The blessing of portability

One of the advantages of Java not mentioned in the previous section is its portability. As said, the simulator has been running for eleven years and during this time it has been ported to several machines, and in the process Java has shown that its portability is real and practical.

Due to its intensive computation needs the Gaia simulator has mostly been run at supercomputing centers, using significant amounts of computation time It was initially run at the *Centre de Supercomputació de Catalunya* (CESCA) were several medium-sized clusters were used: an IBM SP2, a Compaq HPC320, an HP RX2600 and an SGI ALTIX 3700. In all cases the migration from one cluster to the next was seamless, the only serious complication being the adaptation of the execution to the different queue systems in each machine. Later on, in 2005, the simulator was moved to the *Barcelona Supercomputing Center* (BSC) where the *Mare Nostrum* supercomputer had been just installed. Again, the migration was quite simple, with the major complication being how to adapt the execution to the new distributed environment and queue system. Today the simulator is still running there, having cruised through a *Mare Nostrum* upgrade in 2006 without needing any significant change and having consumed a some millions of CPU hours and generated many terabytes of simulated data.

In these years the portability of Java has therefore been a major advantage, saving time and resources that otherwise would have been devoted to adapting the code to the new environment. Furthermore, the portability of Java has also allowed the running of the simulator on other environments like local clusters for testing and development and a cluster at CNES for tailored small simulations through a web page manager.

3.3 Performance in scientific computation

In the initial years of the project the performance in numerical computing of Java was discussed many times. Not having previous experience on its use for scientific computation there was some worry that the language could prove to be too slow at some point of the development. However, some initial tests with a set of some often-used numerical algorithms showed that Java was not much slower that C in solving these problems, thanks to the *Just In Time* (JIT) compilation in the Java virtual machines.

In the next years the JIT virtual machines steadily improved, and the first working code was developed in the project, showing in practice that the implementation in Java was not producing any of the feared bottlenecks. Nowadays, Java has become part of the landscape in scientific programming in Gaia, and although worries about performance are still occasionally an element of resistance, given the accumulated experience and clear savings in development time performance is no longer an issue.

Furthermore, the development model in DPAC is based on six-month development cycles; at the end of each cycle an improved version is produced and tested, leading to further requirements, corrections and improvements for the next cycle. In this framework the optimisation of the system is not as important as the maintainability and flexibility of the code to allow such a quick production. The robustness and clarity of Java helps a lot in such a process, and as mentioned before has probably saved a significant amount of money in programming resources.

A real example encountered during the simulator will illustrate this point. A key piece of the Gaia astrometry is the calculation of the so-called “relativistic corrections”, the changes on the apparent position of the objects in the sky deriving from relativistic effects: aberration, gravitational light bending, etc. This is a complex calculation taking into account the ephemeris of the major solar system bodies and requiring (for a μ as accuracy) to reach the limit of the numerical precision of variables of the "double" type (64 bit floating point).

An initial (legacy) implementation of these calculations was available from S. Klioner in the form of C code and was used in the simulator until 2008 through JNI calls. At that time, in order to avoid the inconveniences of mixing two languages in the simulator, the same author developed for DPAC a new implementation coded in Java. Both implementations were thoroughly compared and results matched at sub-μ as level. However, during the testing it was found that the computation times differed substantially between the two versions, the Java version being between four and ten times **faster** than the C version! Possible external sources of this difference (like overheads of the JNI calls) were ruled out and it was concluded that the difference was actually intrinsic to the code.

It is possible, even likely, that an optimisation of the C code would make it much more efficient, to at least the level of the Java code. However, this example clearly shows how the same developer did a quicker and better job in Java (a language that, unlike C, he was unfamiliar with). On the other hand, the difference possibly comes also from the excellent refactoring afforded by the new JITs virtual machines that now automatically makes many of the performance fixes which previously had to be manually implemented—in C of course all optimisations must be done by the coder.

Finally, the optimization work on the Java code has continued. The team is now exploring the increasing possibilities of the options for aggressive optimization and garbage collection tailoring available in some Java virtual machines (especially the IBM one, used in the MareNostrum supercomputer) that will possibly lead to further improvements in the performance of the simulator.

4 Java, Gaia and commercial uses

When [10] was presented at the Astronomical Data Analysis and Software Systems (ADASS) conference in 1999 most people in that community did not know what Java was and this was the only paper mentioning Java. Two years later about half of the astronomy related projects at ADASS involved Java. Some project must take the first step, in this case Gaia and Planck were experimenting with Java. These experiments possibly made it easier for the Integral SOC and the entire Herschel Science Ground Segment to be written in Java [14]. Gaia and Java go back many years, the initial prototype for the Global Iterative Solution was already in Java in 1999 [9].

Java in the late 90s was not the same as it is today but already then it was seen to have potential for science development. For a project like Gaia we were faced with a life span of over 20 years for our software and an entrenched Fortran community. It was clear already back then, as discussed earlier, that Gaia processing software would not be written in Fortran but needed to be in a more modern, preferably OO, language, C++ being the obvious choice.

The transition for some scientists from Fortran to C++ was seen as difficult or impossible. Java simply worked more easily and was chosen and eventually accepted by all.

4.1 Negative aspects

We do not claim Java is the silver bullet [3] and we have several problems which we live with especially in numerical coding. As already mentioned especially when we started, there were no good numerical libraries. The situation improved from about 2000 onwards when Java started to become really usable for scientific computing in terms of robustness and performance. A number of promising numerical library developments took place (e.g. JAMA—a Java Matrix Package) but until around 2005 most of them had stopped, leaving the distributions and code available but largely unmaintained. A positive exception is the Apache Common Math library which is still in active development and in extensive use in almost all of Gaia's data processing software. Early versions of Apache Common Math were missing linear algebra functionality needed by us but that has improved with time.

In parallel with diminishing efforts for developing numerical libraries we observed a general decline in support for Numerics in Java. For instance the Numerics Working Group of the Java Grande Forum,[1] initially a driving force behind many positive developments around Java Numerics, has effectively ceased to exist. This is a bit worrying along with a general perceived lack of support for Java in the traditional conservative supercomputing scene that in terms of languages remains to be dominated by Fortran and C until today.

In 1998 W. Kahan, one of the key persons behind the IEEE 754 Standard for Binary Floating-Point Arithmetic, delivered a keynote "How Java's Floating-Point Hurts Everyone Everywhere" [5] at a Java HPC workshop. In this contribution Java gets harshly criticized for for a number of IEEE non-compliances which could and should have been avoided by the language designers. In subsequent years Sun was repeatedly asked to rectify the known deficiencies but chose to ignore all complaints and even today those issues raised more than 10 years ago are, to our knowledge, still present in all existing Java implementations. With Java now under Oracle's control it seems unlikely that the situation will change in the foreseeable future. Fortunately, in our view most of the points are fairly subtle in nature and are unlikely to show up as perceivable flaws in "ordinary" numerical application code. We can confirm this for all of Gaia's software and in particular for the AGIS system described here in more detail (Section 2) with one exception: Java does not provide any means to use what IEEE 754 calls 'trap-handlers' for capturing and dealing with numerical exceptions like division by zero. Every arithmetic operation in Java always delivers a valid result and that can make the debugging of

[1] http://math.nist.gov/javanumerics/

numerical code extremely difficult and time consuming. As an example, there was a coding error in the AGIS attitude update that caused the introduction of an NaN value into a large matrix of equations which then propagated and spread through the matrix before eventually leading to NaN in an end result much later. Without the option to have an exception raised at the first occurrence of an NaN condition the only way to find the problem is to check explicitly every intermediate result for NaN (Double.isNaN(x)). For obvious reasons this is not a viable option in a large numerical code.

4.2 Features

One of the things which made java work well for us was its built in features. We were especially happy to have multi threading and distributed programming such as RMI (Remote Method Invocation—allows calling a method on an object on another machine) built in to the language.

4.3 Rise of Java in industry

At the same time (back in the late 90's) we saw in industry a surge of Java programming as it pushed from the the Web language clearly into the back office. Enterprise Java Beans were starting to appear behind web pages accessing databases and wrapping legacy applications to make them network available. Java was no longer an "applet" language for making more interactive web pages—it was handling credit cards and transactions and doing serious work. The fact that Java is 100% portable to all O/Ss where the JVM has been ported to and backward compatible means that using libraries is no longer a painful issue.

Companies such as IBM, Oracle, BEA, Sun and many open source vendors have created Java application servers that support the Java standards in JavaEE produced by Sun.

Since then frameworks to further aid productivity in the development of Java software such as Spring and Hibernate have appeared on the scene and these have helped with the adoption of Java as the technology of choice for developing software. In addition the fact that there are thousands of open source libraries available to use in Java projects has also helped the spread of Java.

4.4 Maintainability, robustness and performance

When developing software there are a number of key issues that have to be addressed including but not limited to Maintainability, Robustness, Scalability.

Addressing maintainability first, Java compared to other languages such as C++ and Fortran offers a number of advantages. The defacto editor for Java, Eclipse, provides a number of features that makes creating and maintaining Java code easier; Graphical syntax highlighting, Refactoring wizards to improve the design of the application; Plugins to produce documentation. Java was also one of the first languages to have a really good implementation of the xUnit testing frameworks with the widely adopted JUnit. With Java 1.5 and the introduction of Generics and Annotations and with techniques such as Aspect Oriented Programming (AOP—not currently used on Gaia) we have the possibility of developing software with less code and the less code there is, the more maintainable it is.

Java is more robust than C++, primarily because the C++ component of all Java programs, the JVM itself, is the same for all Java programs and therefore tried and tested millions of times.

Many people believe that Java is not as scalable as other languages because of the overhead of the Java bytecode interpretation. However the Java Hotspot server compiler, especially the one in Java 1.6, is incredibly efficient at optimising code that is called very frequently.

4.5 Portability and the cloud

Although there are minor problems with Java portability this is usually in Graphical User Interfaces rather than in the type of code forming the majority of the Gaia environment. As stated in Section 3 our simulation code has been ported over many years to many platforms with little effort. Currently development is done on MacOs, Linux and windows systems without issue. Such portability lead us to consider using the Amazon E2C for the Astrometric solution (Section 2). We, unlike CERN [13] are not yet tied in to one architecture. Although these days even CERN use some cloud resource.

With about two person weeks of effort part of our team (Parsons and Olias) got AGIS running on the cloud. Some problems were encountered with the database configuration, although at least Amazon already hand a VM with Oracle on it. Also a scalability problem in our own code was found and remedied—prior to this we had no opportunity to run 1,400 threads. It worked well, at least as well as our cheap in house cluster. Hence we agree with the detractors of the Cloud—it is not a *supercomputer* with super fast interconnect, but then one is not paying super prices either.

In fact although our intention is/was to build a cheap cluster (around 1M euro) for AGIS we estimate all AGIS mission processing for 100 million stars could be done for about 400K euro. When energy is factored in this makes Amazon look very attractive. It largely depends on idle time however—and we in any case would not go without an in house cluster for testing and development. It also appears we now may need to process 500 million sources to counteract possible spatial error correlations (as presented by Berry Holl in Heidelberg in 2009). This would bring the saving close to zero—but one must question if the likes of Amazon can do a cheaper job of maintaining a cluster.

The availability of a much larger number of nodes than we can buy is very interesting both for testing and for production. By using Amazon we could perhaps do our processing faster by using more nodes and still have it cost less than an in house machine. We shall continue to experiment and evaluate this option—in any case the final machine at ESAC for the processing will not be purchased until we are a couple of years into the mission around 2015.

4.6 Virtualisation—non uniqueness

The real power of Java comes from the notion of running on a virtual machine. This should not be, but often is, confused with interpretation such as done in LISP and Smalltalk. This virtue is not shared by many new languages or schemes perhaps the most all encompassing of which is the Microsoft .NET framework in which many languages are compiled to the same virtual machine. We are not trying to say Java is the only language with this feature. In fact with virtualisation suddenly any particular machine/OS/langugae configuration can be equally portable with its own virtual machine. Suddenly putting those legacy apps in a cloud may not be such a hard decision.

4.7 Future of Java

The rise of Java in industry and positive experience we had we Java compared to C++ reinforced our choice of Java as the language for Gaia. This choice was reaffirmed within the Gaia community in 2006 [4]. Indeed at the time of writing all science development missions at ESAC are using Java as their programming language.

We predict that Java will be the language of choice for the foreseeable future because of all the advantages outlined earlier, although we believe that within the Java Virtual Machine, we will see more use of dynamic languages such as Ruby and Groovy for areas of the application that will need to be changed very frequently.

5 Conclusion

One may argue about the definition of High Performance Computing but within the Gaia project since the late 90s we have certainly be doing numerically intensive computing and handling increasingly large amounts of data in a highly distributed manner. All of this is done using Java. We argue that this is a good option for long running projects where portability and maintenance are possibly more important than squeezing the last FLOP from a specific processor.

The portability of Java has served us well in the last decade allowing code to move from SPARC station to Linux boxes, WinTel and even the Marenostrum super computer. Of late we have also leveraged Amazon's E2C resources.

We are not alone in the astronomy world, many other projects are using Java or other high level languages. The ESA missions Herschel, Planck and many archives are Java users. Others such as Spitzer and JWST mentioned using Java at least in part. Other higher level languages such as C# and Python are also in use.

Appendix: Acronyms used in this paper

The following table has been generated from the on-line Gaia acronym list:

Acronym	Description
ADASS	Astronomical Data Analysis Software and Systems
AGIS	Astrometric Global Iterative Solution
BSC	Barcelona Supercomputing Centre
CERN	Centre Europénne pour la Recherche Nucléaire
CESCA	CEntre de Supercomputacío de CAtalunya
CNES	Centre National d'Etudes Spatiales (France)
CPU	Central Processing Unit
DPAC	Data Processing and Analysis Consortium
DPACE	Data Processing and Analysis Consortium Executive
ESA	European Space Agency
ESAC	European Space Astronomy Centre (VilSpa)
FLOP	FLoating-point OPeration
GB	GigaByte
HPC	High-Performance Computing
ICRS	International Celestial Reference System
IEEE	Institute of Electrical and Electronic Engineers
JIT	Just In Time Compiler
JNI	Java Native Interface
JVM	Java Virtual Machine
JWST	James Webb Space Telescope (formerly known as NGST)
MB	MegaByte
OO	Object Oriented
PMD	Software tool to detect software problems
RAM	Random Access Memory
RMI	Remote Method Invocation
SAN	Storage Area Network
SOC	Science Operations Centre
TB	TeraByte
UB	University of Barcelona (Spain)
UML	Unified Modeling Language
VM	Virtual Machine

References

1. Bombrum, A., Lindegren, L., Holl, B., Jordan, S.: Complexity of the Gaia astrometric least-squares problem and the (non-)feasibility of a direct solution method. Astron. Astrophys. **516**, A77 (2010). doi:10.1051/0004-6361/200913503
2. ESA: The Hipparcos and Tycho Catalogues. ESA, ESA SP-1200 (1997)
3. Brooks, F.P. Jr.: No silver bullet essence and accidents of software engineering. Computer **20**, 10–19 (1987). doi:10.1109/MC.1987.1663532
4. Hernandez, J., O'Mullane, W., Huc, C.: Language Recommendation for the DPAC. Tech. Rep., European Space Astronomy Centre, Villafranca, Madrid. http://www.rssd.esa.int/llink/livelink/open/532588 (2006)
5. Kahan, W., Darcy, J.: How Java's Floating-Point Hurts Everyone Everywhere. Tech. Rep., Elect. Eng. & Computer Science, University of California at Berkely. Keynote presented at ACM 1998 Workshop on Java for High-Performance Network Computing held at Standford University. http://www.cs.berkeley.edu/~wkahan/JAVAhurt.pdf (1998)

6. Klioner, S.A.: A practical relativistic model for microarcsecond astrometry in space. Astrophys. J. **125**, 1580–1597 (2003)
7. Makarov, V.V.: Absolute measurements of trigonometric parallaxes with astrometric satellites. Astron. Astrophys. **340**, 309–314 (1998)
8. Murray, C.A.: Vectorial Astrometry. Adam Hilger, Bristol (1983)
9. O'Mullane, W., Lindegren, L.: An object-oriented framework for GAIA data processing. Balt. Astron. **8**, 57–72 (1999)
10. O'Mullane, W., Hazell, A., Bennett, K., Bartelmann, M., Vuerli, C.: ESA survey missions and global processing. In: Manset, N., Veillet, C., Crabtree, D. (eds.) Astronomical Data Analysis Software and Systems IX, Astronomical Society of the Pacific Conference Series, vol. 216, pp. 419–422 (2000)
11. O'Mullane, W., Lammers, U., Bailer-Jones, C., Bastian, U., Brown, A.G.A., Drimmel, R., Eyer, L., Huc, C., Katz, D., Lindegren, L., Pourbaix, D., Luri, X., Torra, J., Mignard, F., van Leeuwen, F.: Gaia data processing architecture. In: Shaw, R.A., Hill, F., Bell, D.J. (eds.) Astronomical Data Analysis Software and Systems XVI, Astronomical Society of the Pacific Conference Series, vol. 376, pp. 99–108 (2007)
12. Perryman, M.A.C., de Boer, K.S., Gilmore, G., Høg, E., Lattanzi, M.G., Lindegren, L., Luri, X., Mignard, F., Pace, O., de Zeeuw, P.T.: GAIA: composition, formation and evolution of the Galaxy. Astron. Astrophys. **369**, 339–363 (2001). doi:10.1051/0004-6361:20010085. arXiv:astro-ph/0101235
13. Shiers, J.: Grid today, clouds on the horizon. Comput. Phys. Commun. **180**(4), 559–563 (2009). doi:10.1016/j.cpc.2008.11.027. http://www.sciencedirect.com/science/article/B6TJ5-4V70NH3-1/2/2fdc77415de62c2e360754c969911850. Special issue based on the Conference on Computational Physics 2008—CCP 2008
14. Wieprecht, E., Brumfit, J., Bakker, J., de Candussio, N., Guest, S., Huygen, R., de Jonge, A., Matthieu, J.J., Osterhage, S., Ott, S., Siddiqui, H., Vandenbussche, B., de Meester, W., Wetzstein, M., Wiezorrek, E., Zaal, P.: The HERSCHEL/PACS common software system as data reduction system. In: Ochsenbein, F., Allen, M.G., Egret, D. (eds.) Astronomical Data Analysis Software and Systems (ADASS) XIII, Astronomical Society of the Pacific Conference Series, vol. 314, pp. 376–379 (2004)

Bibliography

[FM-030], Mignard, F., Drimmel, R. (eds.) 2007, *DPAC: Proposal for the Gaia Data Processing*,
`GAIA-CD-SP-DPAC-FM-030-02`,
URL `http://www.rssd.esa.int/llink/livelink/open/2720336`

[NB-003], Bach, N., Lammers, U., 2008, *AGIS Validation Test Report of CY03-C3*,
`GAIA-C3-TR-ESAC-NB-003`,
URL `http://www.rssd.esa.int/llink/livelink/open/2811474`

[NB-001], Bach, N., Raison, F., Lammers, U., 2008, *AGIS Validation Test Report of CY03-C1*,
`GAIA-C3-TR-ESAC-NB-001`,
URL `http://www.rssd.esa.int/llink/livelink/open/2809823`

[NB-002], Bach, N., Raison, F., Lammers, U., 2008, *AGIS Validation Test Report of CY03-C2*,
`GAIA-C3-TR-ESAC-NB-002`,
URL `http://www.rssd.esa.int/llink/livelink/open/2809930`

[BAS-003], Bastian, U., 2007, *Reference systems, conventions and notations for Gaia*,
`GAIA-CA-SP-ARI-BAS-003`,
URL `http://www.rssd.esa.int/llink/livelink/open/358698`

Bastian, U., Biermann, M., 2005, A&A, 438, 745, ADS Link

[GAIA-ARI-BAS-011-05], Bastian, U., Jordan, S., 2009, *Extended geometric calibration model for Gaia's Astro instrument*,
`GAIA-ARI-BAS-011-05`,
URL `http://www.rssd.esa.int/llink/livelink/open/420453`

Beck, K., 1999, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 1st edn.

Bombrun, A., Lindegren, L., Holl, B., Jordan, S., 2010, A&A, 516, A77, ADS Link

Booch, G., Rumbaugh, J., Jacobson, I., 2005, *The Unified Modeling Language User Guide*, Addison-Wesley Professional, 2nd edn.

Bradley, J., 1727, Royal Society of London Philosophical Transactions Series I, 35, 637, ADS Link

Castillo, E., Conejo, A.J., Pruneda, R.E., Solares, C., 2005, IEEE Transactions on power systems, 20, 1656

[WOM-018], leaders CU1, D.C., 2011, *DPAC Software and System Specification*, `GAIA-C1-SP-DPAC-WOM-018`, URL `http://www.rssd.esa.int/llink/livelink/open/2786798`

Danelutto, M., Dazzi, P., 2006, In: Parallel Computing: Current and Future Issues of High-End Computing, 681–688

de Felice, F., Lattanzi, M.G., Vecchiato, A., Bernacca, P.L., 1998, A&A, 332, 1133, ADS Link

Demichev, A., Foster, D., Kalyaev, V., et al., 2003, ArXiv Computer Science e-prints, ADS Link

ESA, 1997, *The Hipparcos and Tycho Catalogues*, ESA, ESA SP-1200

[ECSS-E-40-1B], ESA Publications Division, 2003, *Space enginneering - Software - Part 1: Principles and requirements*, `ECSS-E-40 Part 1B`

Foster, I., 2006, Journal of Computer Science and Technology, 21 No. 4, 513

[ESA-SCI(2000)4], GAIA Science Advisory Group, 2000, *GAIA. Composition, Formation and Evolution of the Galaxy [The GAIA Study Report (ESA-SCI(2000)4)]*, `ESA-SCI(2000)4`, URL `http://www.rssd.esa.int/llink/livelink/open/359232`

Gamma, E., Helm, R., Johnson, R., Vlissides, J., 1994, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional Computing Series

Gilmore, G.F., de Boer, K.S., Favata, F., et al., 2000, In: Breckinridge, J.B., Jakobsen, P. (eds.) Proc. SPIE Vol. 4013, p. 453-472, UV, Optical, and IR Space Telescopes and Instruments, James B. Breckinridge; Peter Jakobsen; Eds., vol. 4013 of Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, 453–472, ADS Link

[GMV-GDAAS-RP-001], Gonzalez, L., Serraller, I., Torra, J., et al., 2002, *GDAAS Final Report*, `GMV-GDAAS-RP-001`, URL `http://www.rssd.esa.int/llink/livelink/open/357730`

Górski, K.M., Hivon, E., Banday, A.J., et al., 2005, ApJ, 622, 759, ADS Link

Hamilton, W.R., 1844, In: Proceedings of the Royal Irish Academy, vol. 3, 1–16, `http://www.maths.tcd.ie/pub/HistMath/People/Hamilton/OnQuat/OnQuat.pdf`

Hamilton, W.R., 1847, In: Proceedings of the Royal Irish Academy, vol. 3, 1–16, `http://www.maths.tcd.ie/pub/HistMath/People/Hamilton/Quatern2/Quatern2.html`

Hankins, T.L., 1980, *Sir William Rowan Hamilton*, The Johns Hopkins University Press

[JH-001], Hernandez, J., 2012, *Main Database Interface Control Document*, GAIA-C1-SP-ESAC-JH-001, URL http://www.rssd.esa.int/llink/livelink/open/2786145

[JH-004], Hernandez, J., ter Linden, M., 2012, *Main Database Software Requirements Specification*, GAIA-C1-SP-ESAC-JH-004, URL http://www.rssd.esa.int/llink/livelink/open/2698216

[JSH-004], Hoar, J., 2005, *Proposal for Java Workshop 2006*, GAIA-C1-MN-ESAC-JSH-004, URL http://www.rssd.esa.int/llink/livelink/open/530598

[DH-001], Hobbs, D., Lammers, U., 2011, *AGIS Software Test Specification*, GAIA-C3-SP-LU-DH-001, URL http://www.rssd.esa.int/llink/livelink/open/2755720

J.A. Zensus, P.N., P.J. Napier, 1995, *Very Long Baseline Interferometry and the VLBA*, Astronomical Society of the Pacific,, asp conference series vol. 82 edn.

Jacobson, I., Booch, G., Rumbaugh, J., 1999, *The Unified Software Development Process*, Addison-Wesley Professional, 1st edn.

Kane, T.R., Likins, P.W., Levinson, D.A., 1983, *Spacecraft dynamics*, McGraw Hill Book Company, 1 edn.

Klioner, S.A., 2001, ArXiv Astrophysics e-prints, ADS Link

Klioner, S.A., 2003, AJ, 125, 1580, ADS Link

Klioner, S.A., 2005, In: Turon, C., O'Flaherty, K.S., Perryman, M.A.C. (eds.) The Three-Dimensional Universe with Gaia, vol. 576 of ESA Special Publication, 207–+, ADS Link

Klioner, S.A., Soffel, M.H., 2005, In: Turon, C., O'Flaherty, K.S., Perryman, M.A.C. (eds.) The Three-Dimensional Universe with Gaia, vol. 576 of ESA Special Publication, 305–+, ADS Link

Korn, G.A., Korn, T.M., 1961, *Mathematical handbook for scientists and engineers*, McGraw Hill Book Company, 1 edn.

Kruchten, P., 2003, *The Rational Unified Process: An Introduction*, Addison-Wesley Professional, 3rd edn.

[UL-018], Lammers, U., 2006, *First AGIS results with 1 million star/5 years simulated data*, GAIA-C3-PR-ESAC-UL-018, URL http://www.rssd.esa.int/llink/livelink/open/1089973

Lammers, U., O'Mullane, W., Bastian, U., 2004, *Final Report of GDAAS Technical Review*, Tech. rep., ESA, GAIA-UL/WM/UB-004

[UL-015], Lammers, U., Hernandez, J., Hoar, J., O'Mullane, W., 2006, *ESAC implementation of the astrometric global iterative solution*,
`GAIA-C3-TN-ESAC-UL-015`,
URL `http://www.rssd.esa.int/llink/livelink/open/535411`

[UL-021], Lammers, U., Lindegren, L., O'Mullane, W., et al., 2006, *Simulation Requirements for AGIS Development Cycles 1 and 2*,
`GAIA-C3-SP-ESAC-UL-021`,
URL `http://www.rssd.esa.int/llink/livelink/open/2695195`

Larman, C., Basili, V.R., 2003, *Iterative and Incremental Development: A Brief History*, Tech. rep.,
`http://www2.umassd.edu/SWPI/xp/articles/r6047.pdf`

van Leeuwen, F., 2007, *Hipparcos, the New Reduction of the Raw Data*, Springer, Astrophysics and Space Science Library. Vol. 350 edn.

[SAG-LL-014], Lindegren, L., 1998, *The scanning law for GAIA.*,
`SAG-LL-014`,
URL `http://www.rssd.esa.int/llink/livelink/open/357121`

[SAG-LL-025], Lindegren, L., 1998, *Point spread functions for GAIA including abberrations*,
`SAG-LL-025`,
URL `http://www.rssd.esa.int/llink/livelink/open/356774`

[SAG-LL-030], Lindegren, L., 2000, *Attitude parameterization for GAIA*,
`SAG-LL-030`,
URL `http://www.rssd.esa.int/llink/livelink/open/358607`

[GAIA-LL-053], Lindegren, L., 2004, *Chromaticity specification*,
`GAIA-LL-053`,
URL `http://www.rssd.esa.int/llink/livelink/open/365293`

[GAIA-LL-055], Lindegren, L., 2004, *A proposed new algorithm for source updating*,
`GAIA-LL-055`,
URL `http://www.rssd.esa.int/llink/livelink/open/390988`

[GAIA-LL-057], Lindegren, L., 2004, *Scientific requirements for basic angle stability monitoring*,
`GAIA-LL-057`,
URL `http://www.rssd.esa.int/llink/livelink/open/413563`

Lindegren, L., 2005, In: Turon, C., O'Flaherty, K.S., Perryman, M.A.C. (eds.) The Three-Dimensional Universe with Gaia, vol. 576 of ESA Special Publication, 29–+, ADS Link

[LL-071], Lindegren, L., 2007, *Convergence properties of AGIS*,
`GAIA-C3-TN-LU-LL-071`,
URL `http://www.rssd.esa.int/llink/livelink/open/2745180`

[LL-063], Lindegren, L., 2008, *Geometric calibration model for the 1M astrometric GIS demonstration*,
`GAIA-C3-TN-LU-LL-063`,
URL `http://www.rssd.esa.int/llink/livelink/open/510843`

[LL-093], Lindegren, L., 2011, *Selection of Primary Sources in AGIS*,
`GAIA-C3-TN-LU-LL-093`,
URL `http://www.rssd.esa.int/llink/livelink/open/3106140`

Lindegren, L., Lammers, U., Hobbs, D., et al., 2012, A&A, 538, A78, ADS Link

[XL-004], Luri, X., Babusiaux, C., 2007, *Consolidated list of simulation requirements and CU2 planning for cycle 3*,
`GAIA-C2-SP-UB-XL-004`,
URL `http://www.rssd.esa.int/llink/livelink/open/2760525`

Makarov, V.V., 1998, A&A, 340, 309, ADS Link

Miller, W.W., III, Sontag, C., Rose, J.F., 2003, In: Payne, H.E., Jedrzejewski, R.I., Hook, R.N. (eds.) Astronomical Data Analysis Software and Systems XII, vol. 295 of Astronomical Society of the Pacific Conference Series, 261–+, ADS Link

Moore, G.E., 1965, vol. 38

Nieto-Santisteban, M.A., Szalay, A.S., Thakar, A.R., et al., 2005, ArXiv Computer Science e-prints, ADS Link

[WOM-003], O'Mullane, W., 2005, *Large scientific data systems - analysis of some existing projects and their applicability to Gaia*,
`GAIA-C1-TN-ESAC-WOM-003`,
URL `http://www.rssd.esa.int/llink/livelink/open/497678`

[WOM-001], O'Mullane, W., Lammers, U., 2007, *Work breakdown structures for DPAC*,
`GAIA-C1-TN-ESAC-WOM-001`,
URL `http://www.rssd.esa.int/llink/livelink/open/497865`

O'Mullane, W., Lindegren, L., 1999, Baltic Astronomy, 8, 57, ADS Link

[GAIA-SOW-001], O'Mullane, W., Perryman, M., 2000, *GAIA data access and analysis study SOW*,
`GAIA-SOW-001`,
URL `http://www.rssd.esa.int/llink/livelink/open/357710`

O'Mullane, W., Banday, A.J., Górski, K.M., Kunszt, P., Szalay, A.S., 2001, In: Banday, A.J., Zaroubi, S., Bartelmann, M. (eds.) Mining the Sky, 638–+, ADS Link

O'Mullane, W., Li, N., Nieto-Santisteban, M., et al., 2005, In: International Conference on Web Services,
Also MS technote `http://arxiv.org/pdf/cs.DC/0502072`

[WOM-006], O'Mullane, W., Hernandez, J., Hoar, J., et al., 2006, *Tackling AGIS development with Agile programming*,
`GAIA-C1-TN-ESAC-WOM-006`,
URL `http://www.rssd.esa.int/llink/livelink/open/535099`

O'Mullane, W., Lammers, U., Bailer-Jones, C., et al., 2006, ArXiv Astrophysics e-prints, ADS Link

O'Mullane, W., Hoar, J., Lammers, U., 2007, ArXiv e-prints, 712, ADS Link

[WOM-014], O'Mullane, W., Lindegren, L., Lammers, U., 2007, *Source relegation and photon densities for AGIS*,
`GAIA-C3-TN-ESAC-WOM-014`,
URL `http://www.rssd.esa.int/llink/livelink/open/2777837`

[WOM-004], O'Mullane, W., Hernandez, J., Hoar, J., et al., 2009, *Astrometric Global Iterative Solution Software Design Document*,
`GAIA-C3-SP-ESAC-WOM-004`,
URL `http://www.rssd.esa.int/llink/livelink/open/535350`

O'Mullane, W., Lammers, U., Lindegren, L., Hernandez, J., Hobbs, D., 2011a, Experimental Astronomy, 31, 215, ADS Link

O'Mullane, W., Luri, X., Parsons, P., et al., 2011b, Experimental Astronomy, 31, 243, ADS Link

Perryman, M.A.C., de Boer, K.S., Gilmore, G., et al., 2001, A&A, 369, 339, ADS Link

[UB-GDAAS2-TN-035], Portell, J., Figueras, F., Fabricius, C., et al., 2006, *Final revision of the GDAAS2 Large-Scale Test*,
`UB-GDAAS2-TN-035`,
URL `http://www.rssd.esa.int/llink/livelink/open/2694482`

Psiaki, M.L., 2006, vol. 29 No. 3, 695–703

Riley, K.F., Hobson, M., Bence, S., 2006, *Mathematical Methods for Physics and Engineering*, Cambridge University Press, 3 edn.

Rose, J., Akella, R., Binegar, S., et al., 1995, In: Shaw, R.A., Payne, H.E., Hayes, J.J.E. (eds.) Astronomical Data Analysis Software and Systems IV, vol. 77 of Astronomical Society of the Pacific Conference Series, 429–+, ADS Link

Royce, W., 1970, In: Proceedings of IEEE WESCON, 1–9,
`http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf`

[AS-009], Short, A., 2006, *Calibration of Radiation Effects in Gaia CCDs*,
`GAIA-CH-TN-ESA-AS-009`,
URL `http://www.rssd.esa.int/llink/livelink/open/1807224`

Shuster, D., Malcom, 1993, vol. 41, n.4, 439–517

Turon, C., O'Flaherty, K.S., Perryman, M.A.C. (eds.), 2005, *The Three-Dimensional Universe with Gaia*, vol. 576 of ESA Special Publication, ADS Link

Tzu, L., 1994, *The Tao-te Ching*, The Internet Classics Archive, Translated by James Legge

van Leeuwen, F., Fantino, E., 2005, A&A, 439, 791, ADS Link

[AVE-002], Vecchiato, A., 2008, *The astrometric model of GSR1*, `GAIA-C3-TN-INAF-AVE-002`, URL `http://www.rssd.esa.int/llink/livelink/open/2810365`

Vlemmings, W.H.T., Chatterjee, S., Brisken, W.F., et al., 2005, Memorie della Societa Astronomica Italiana, 76, 531, ADS Link

Wilkinson, B., Allen, M., 1999, *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers*, Prentice Hall