Using the Conjugate Gradient method in AGIS

L. Lindegren

GAIA–C3–TN–LU–LL–077–01 (15 October 2008)

Abstract. This note describes how the Conjugate Gradient method could be implemented in the AGIS framework for improved speed and accuracy.

1 Introduction

Since the early phases of the Gaia project, the baseline method for the so-called core astrometric solution has been some variant of what is now known as the Astrometric Global Iterative Solution (AGIS). The core astrometric solution aims to determine the astrometric parameters for a subset of primary sources self-consistently with the instrument attitude and geometric calibration. The basic idea is very simple. From the way observations are made with Gaia it is obvious that

1. if the attitude and calibration parameters were known, it would be a simple matter to solve the (astrometric) source parameters, one source at a time;
2. if the source and calibration parameters were known, it would be a simple matter to solve the attitude parameters, one attitude interval at a time;
3. if the source and attitude parameters were known, it would be a simple matter to solve the calibration parameters.

We may refer to each of the above processes as S (source update), A (attitude update), and C (calibration update), respectively.[1] An intuitively natural way to obtain a solution that is consistent with all three processes is to repeat them cyclically until convergence. This is the basic AGIS algorithm, hereafter referred to as *simple iteration* (Sect. 3.1).

A statistical argument can be invoked to explain why the simple iteration ought to converge to a correct solution. The argument is based on the number of observations that contribute to the determination of a certain parameter in a given iteration, and their distribution in time and space. Consider for example the S process, where the source parameters for a given star is obtained by combining data from many different epochs distributed over the mission – typically about 80 field of view transits over a five-year period. It can be expected that the attitude errors from the different epochs are at least partially independent, so that their effects on the source update to a certain degree cancel out. Since the same observations are also distributed among the different CCDs, we can expect the calibration errors on the individual CCDs to cancel out to a certain extent. The net effect is that the

[1]A fourth process G (global update) is often considered in addition to S, A and C. This is meant to determine parameters that affect all observations, such as the PPN γ parameter. However, from an implementation viewpoint these global parameters could just as well be regarded as calibration parameters, and their determination therefore included in C. The global update is not considered in the following.

attitude and calibration errors propagate into the source update with a damping factor depending on the number of 'independent' observations contributing to the update. Now if we instead consider process A, we see that the attitude at any time t is obtained by combining the observations of a large number of primary sources, namely those that are simultaneously visible in the astrometric field – typically at least a thousand sources. Their astrometric errors will not all be the same, and they therefore propagate into the attitude determination with a certain damping factor. These observations are moreover spread out over all the CCDs in the astrometric field, so that the calibration errors will to a large extent average out. Finally, for process C we can see that a similar averaging effect is at work when propagating the source and attitude errors into the calibration errors. In each AGIS iteration, therefore, the errors from the previous iteration propagate with a damping factor less than unity into the errors of the current iteration. The iteration process will consequently converge, the quicker the smaller this damping factor is.

Naively, from arguments such as the above, one would expect a significant damping of the errors with each iteration – perhaps with a factor of the order of 0.1–0.3, so that the simple iteration would converge geometrically in a relatively small number of iterations. Experience, first with GDAAS [1] and later in AGIS [5] showed however that an initial rapid improvement was soon followed by a much slower convergence. After several tens of iterations, the damping factor was usually as large as 0.9 or even closer to unity. This phenomenon could be explained by elementary theory [3], which also led to a concept for accelerating the convergence [4]. Although this *accelerated iteration* (Sect. 3.2) gave very significant improvements, it was felt that even more might be gained by employing some standard iterative algorithm such as the Conjugate Gradient (CG) method.

CG is considerably more complex than the simple and accelerated iterations, and it is not immediately obvious if and how it can be implemented in the AGIS framework. In trying to answer this question, a main problem is that the formal matrix equations that describe what AGIS is doing *mathematically* sometimes have a rather remote resemblance to how the actual calculations are organized. This makes it non-trivial to translate the formal CG algorithm into processes that can be implemented within AGIS. For example, CG, like most standard methods for the iterative solution of large linear systems $\boldsymbol{Nx} = \boldsymbol{n}$, depends on the assumption that the vector $\boldsymbol{Ny}$ can be computed, with reasonable effort, for arbitrary vector $\boldsymbol{y}$. But what exactly would be the corresponding procedure in AGIS? In order to clarify such issues we begin by analyzing the current (simple and accelerated) iteration methods both in formal language and as implemented in AGIS.

2 The normal equations for the astrometric solution

In the subsequent equations, subscripts ℓ, i, j, and k always have the following meaning:

- index ℓ represents an individual (one- or two-dimensional) astrometric observation;
- index i represents a primary source;
- index j represents an attitude interval, i.e. a time interval for which an independent parametrization of the attitude is possible (typically, the attitude intervals are separated by natural breaks in the data);
- index k is used for successive iterations of the solution.

There is a unique and known mapping from ℓ to (i, j), such that each observation ℓ is associated with exactly one source i and one attitude interval j.

The core astrometric processing is a least-squares solution of the non-linear observation equations for the primary stars, which can be symbolically written

$$\left.\begin{aligned} t_\ell &= f_\ell(\boldsymbol{s}_i, \boldsymbol{a}_j, \boldsymbol{c}) + \text{noise} \\ \mu_\ell &= g_\ell(\boldsymbol{s}_i, \boldsymbol{a}_j, \boldsymbol{c}) + \text{noise} \end{aligned}\right\} \tag{1}$$

where t_ℓ is the observation time and μ_ℓ the AC pixel coordinate (if it was measured) in observation ℓ. This observation depends on a certain subset of the total set of unknowns, namely the source parameters $\boldsymbol{s}_i$, the attitude parameters $\boldsymbol{a}_j$, and the calibration parameters $\boldsymbol{c}$. The total set of unknowns for I primary stars and J attitude intervals is

$$\boldsymbol{x} = (\{\boldsymbol{s}_i\}_{i=0}^{I-1},\, \{\boldsymbol{a}_j\}_{j=0}^{J-1},\, \boldsymbol{c}) \tag{2}$$

With $R_\ell(\boldsymbol{x}) = t_\ell - f_\ell(\boldsymbol{s}_i, \boldsymbol{a}_j, \boldsymbol{c})$ denoting the residual in observation ℓ as function of the model parameters, and σ_ℓ its estimated uncertainty (from the centroiding algorithm), the least-squares processing aims to minimize

$$\chi^2(\boldsymbol{x}) = \sum_\ell \left(\frac{R_\ell(\boldsymbol{x})}{u\sigma_\ell}\right)^2 w\left(\frac{R_\ell(\boldsymbol{x})}{u\sigma_\ell}\right) \tag{3}$$

where $w(z)$ is a down-weighting function (intended to take care of outliers, with $w(z) \simeq 1$ for $|z| < 3$) and u is the unit weight error (ideally $u = 1$). For simplicity we have included in the sum (3) only the contributions from the AL observations; in reality it must of course include the AC observations as well.

It is seen that the minimization of $\chi^2(\boldsymbol{x})$ (for fixed u) is a non-linear problem for two reasons: first because $f_\ell(\boldsymbol{x})$ is non-linear, and secondly because $w(z)$ is non-linear. The non-linearity of $f_\ell(\boldsymbol{x})$ (and $g_\ell(\boldsymbol{x})$) is however very weak as long as the changes in the parameters are small; for example, if they are less than $\epsilon = 0.2$ arcsec then the second-order terms are of order $\epsilon^2 \simeq 0.2$ μarcsec. Therefore, the nonlinearity of the observation model (1) is not an issue for the AGIS iterations.

The outlier treatment via the down-weighting function $w(z)$ does however create significant non-linearity, and consequently a need for iterating the solution, until the proper identification and down-weighting of the outliers have stabilized. This only happens after several iterations. Its interaction with the overall AGIS scheme is a separate problem that will not be addressed here. Assuming that the weight factors in (3) are fixed after some initial iterations, we are left with a linear problem. We write this symbolically

$$\boldsymbol{N}\boldsymbol{x} = \boldsymbol{n} \tag{4}$$

where $\boldsymbol{N}$ is the normal equations matrix, $\boldsymbol{x}$ now represents the *updates* (corrections) to be applied to the total set of parameters, and $\boldsymbol{n}$ is the right-hand side of the normal equations. The fact that $\boldsymbol{n} \neq \boldsymbol{0}$ just means that we have not yet found the optimal set of parameters to minimize (3), but we assume that we are in the linear domain of the problem, so that $\boldsymbol{N}$ is in principle fixed.

Neglecting the dependence of the weights on the parameters, the elements of $\boldsymbol{N}$ and $\boldsymbol{n}$ are

$$N_{pq} = \sum_\ell \left(-\frac{\partial R_\ell}{\partial x_p}\right)\left(-\frac{\partial R_\ell}{\partial x_q}\right)\frac{1}{\tilde{\sigma}_\ell^2}\,, \qquad n_p = \sum_\ell \left(-\frac{\partial R_\ell}{\partial x_p}\right)\frac{R_\ell}{\tilde{\sigma}_\ell^2} \tag{5}$$

where $\tilde{\sigma}_\ell = u\sigma_\ell w(R_\ell/u\sigma_\ell)^{-1/2}$ is the standard error modified by the unit weight error and down-weighting.

3 Formal algorithms

In this section we first describe the AGIS schemes used until now in terms of formal procedures for iteratively solving the linear system

$$\boldsymbol{N}\boldsymbol{x} = \boldsymbol{n} \tag{6}$$

based on the initial guess $\boldsymbol{x}_0$ and using the preconditioner $\boldsymbol{K}$. The significance of the preconditioner is that $\boldsymbol{K}$ should in some sense be an approximation of $\boldsymbol{N}$, and that the system $\boldsymbol{K}\boldsymbol{w} = \boldsymbol{r}$ should be possible to solve, with moderate effort, for arbitrary $\boldsymbol{r}$. Thus, rather than solving systems of the form $\boldsymbol{N}\boldsymbol{w} = \boldsymbol{r}$ (which we cannot), we instead solve systems of the form $\boldsymbol{K}\boldsymbol{w} = \boldsymbol{r}$ (which we can), and use iteration to arrive at the correct result. How quickly the iterations converge depends on how close (in a certain sense) $\boldsymbol{K}$ is to $\boldsymbol{N}$.

Considering only the source and attitude updates, the block structure of the normal equations matrix is

$$\boldsymbol{N} = \begin{bmatrix} \boldsymbol{S} & \boldsymbol{U} \\ \boldsymbol{U}' & \boldsymbol{A} \end{bmatrix} \tag{7}$$

where the source normal matrix $\boldsymbol{S}$ is block diagonal with symmetric 5×5 matrices (one per primary source) along the main diagonal; the attitude normal matrix $\boldsymbol{A}$ is a block band diagonal matrix consisting of 4×4 blocks (for the quaternion components) arranged in a band along the main diagonal with a bandwidth determined by the order of the attitude spline; $\boldsymbol{U}$ is a sparse matrix linking the source and attitude parameters. The calculation of these matrices in terms of the observation equations is detailed in Sect. 5.

Thanks to the simple structures of $\boldsymbol{S}$ and $\boldsymbol{A}$, there are a few obvious choices for the preconditioner $\boldsymbol{K}$. The simplest is

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0}' & \boldsymbol{A} \end{bmatrix} \tag{8}$$

where $\boldsymbol{0}$ is a matrix of zeroes. We may call this the Jacobi preconditioner, because its use in the simple iteration scheme (Sect. 3.1) results in a Jacobi-type block-iteration method. Alternatively, we can use the Gauss–Seidel preconditioner

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{U}' & \boldsymbol{A} \end{bmatrix} \tag{9}$$

which is closely related to the Gauss–Seidel block-iteration method. Both $\boldsymbol{S}$ and $\boldsymbol{A}$ are symmetric and positive definite, so both versions of $\boldsymbol{K}$ are non-singular.

For the Jacobi preconditioner the system $\boldsymbol{K}\boldsymbol{w} = \boldsymbol{r}$ reduces to two independent systems, one for the source part (subscript S) and one for the attitude part (subscript A):

$$\boldsymbol{S}\boldsymbol{w}_S = \boldsymbol{r}_S\,, \qquad \boldsymbol{A}\boldsymbol{w}_A = \boldsymbol{r}_A \tag{10}$$

For the Gauss–Seidel preconditioner the source and attitude parts are coupled,

$$\boldsymbol{S}\boldsymbol{w}_S = \boldsymbol{r}_S\,, \qquad \boldsymbol{A}\boldsymbol{w}_A = \boldsymbol{r}_A - \boldsymbol{U}'\boldsymbol{w}_S \tag{11}$$

so that the source part must first be solved, and its result used in the subsequent solution of the attitude part. This means that the attitude update is based on the updated source parameters. This is normally the case in the current AGIS algorithms (Sects. 3.1 and 3.2), which therefore effectively use the Gauss–Seidel preconditioner.

3.1 Simple iteration

The original AGIS (without acceleration) is an application of the so-called *simple iteration* procedure [2], which can be formally described as follows:

```
initial guess $\boldsymbol{x}_0$
for $k = 0, 1, \ldots$
    $\boldsymbol{r}_k = \boldsymbol{n} - \boldsymbol{N}\boldsymbol{x}_k$                    (12)
    $\boldsymbol{w}_k = \boldsymbol{K}^{-1}\boldsymbol{r}_k$                    (13)
    $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{w}_k$                    (14)
    if $\boldsymbol{x}_{k+1}$ accurate enough then quit
next $k$
```

The stopping criterion is simply that some norm of $\boldsymbol{w}_k$ is below a given threshold. It goes without saying that (13) is just a short-hand notation for 'solve the system of equations $\boldsymbol{K}\boldsymbol{w}_k = \boldsymbol{r}_k$', cf. (10) and (11).

In the description above, the first parameters to be updated are the source parameters. It is however also possible to start the iterations by updating the attitude parameters, then the sources, then the attitude, etc. It is easily seen that this is equivalent to using the preconditioner

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{S} & \boldsymbol{U} \\ \boldsymbol{0}' & \boldsymbol{A} \end{bmatrix} \tag{15}$$

which is the transpose of (9). This is still a Gauss–Seidel-type preconditioner, but applied to the problem where the source and attitude unknowns have been interchanged.

3.2 Accelerated iteration

The accelerated AGIS [4] can be described as follows:[2]

```
initial guess $\boldsymbol{x}_0$
for $k = 0, 1, \ldots$
    $\boldsymbol{r}_k = \boldsymbol{n} - \boldsymbol{N}\boldsymbol{x}_k$    (16)
    $\boldsymbol{w}_k = \boldsymbol{K}^{-1}\boldsymbol{r}_k$    (17)
    if mod$(k, 2) = 0$
        $\omega_k = 1$    (18)
    else
        $c_k = (\boldsymbol{w}'_{k-1}\boldsymbol{w}_k)/(\boldsymbol{w}'_{k-1}\boldsymbol{w}_{k-1})$    (19)
        $\omega_k = 1/(1 - c_k)$    (20)
    endif
    $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \omega_k\boldsymbol{w}_k$    (21)
    if $\boldsymbol{x}_{k+1}$ accurate enough then quit
next $k$
```

Here it is assumed that the acceleration is applied in every second iteration, as recommended in [4]. The stopping criterion is that some norm of $\omega_k\boldsymbol{w}_k$ should be below a given threshold.[3] As in the simple iteration, the Gauss–Seidel preconditioner is used.

The kernel of both the simple and accelerated iteration consists of the two formulae

$$\left.\begin{aligned} \boldsymbol{r} &= \boldsymbol{n} - \boldsymbol{N}\boldsymbol{x} \\ \boldsymbol{w} &= \boldsymbol{K}^{-1}\boldsymbol{r} \end{aligned}\right\} \qquad (22)$$

for calculating the next update $\boldsymbol{w}$ via the intermediate vector $\boldsymbol{r}$. The actual calculations do not exactly follow these equations, although the end result is mathematically the same. To minimize memory usage and allow a high degree of parallelization, the calculations are done one source at a time, while accumulating the modified normal equations for the attitude [the second equation in (11)]. This means that the two formulae in (22) are partially worked on in parallel. This is possible because, formally, it is the *same* vector $\boldsymbol{r}$ that appears in both equations. One consequence of this procedure, when used with the Gauss–Seidel preconditioner, is however that the full vector $\boldsymbol{r}$ is never explicitly calculated. In Sect. 5 we discuss in more detail the implementation of (22) and in particular how it may be possible to compute $\boldsymbol{r}$ as well.

[2]The actual implementation of the accelerated AGIS differs slightly from the algorithm as described above. First, the regression coefficient of successive updates, c_k in (19), is not determined from the full vectors $\boldsymbol{w}_{k-1}$ and $\boldsymbol{w}_k$, but only from a much smaller sub-vector corresponding to the parallaxes of some 1000 primary sources. Secondly, even from these sub-vectors, the coefficient c_k is determined by a more robust method than the unweighted least-squares regression suggested by (19). Thirdly, the update is not exactly applied as in (21). Instead, only the source parameters are updated according to (21), which is followed by the second step in (11) for the attitude update, using the updated source parameters. Thus, the acceleration factor ω_k is only indirectly applied to the attitude updates via the link matrix $\boldsymbol{U}$, but not directly to the attitude part of $\boldsymbol{w}_k$. All of these modifications were introduced for practical reasons. It is not known how they affect the convergence, but it is possible that a more rigorous implementation of (16)–(21) might give improved convergence. Also, it is not obvious that the algorithm would fail if the acceleration were applied in every iteration (except $k = 0$).

[3]Actually, it would be better to use the norm of $\max(\omega_k, \omega_{k-1})\boldsymbol{w}_k$ as a stopping criterion.

4 The Conjugate Gradient method

4.1 Formal algorithm

We now consider the Conjugate Gradient (CG) method with preconditioning. A suitable starting point is the algorithm given in Fig. 5.2 of [6]. With a slight adaptation of notations, the algorithm is:

```
initial guess $\boldsymbol{x}_0$
$\boldsymbol{r}_0 = \boldsymbol{n} - \boldsymbol{N}\boldsymbol{x}_0$                                  (23)
for $k = 1, 2, \ldots$
    $\boldsymbol{w}_{k-1} = \boldsymbol{K}^{-1}\boldsymbol{r}_{k-1}$                             (24)
    $\rho_{k-1} = \boldsymbol{r}'_{k-1}\boldsymbol{w}_{k-1}$                               (25)
    if $k = 1$
        $\boldsymbol{p}_k = \boldsymbol{w}_{k-1}$                                   (26)
    else
        $\beta_{k-1} = \rho_{k-1}/\rho_{k-2}$                           (27)
        $\boldsymbol{p}_k = \boldsymbol{w}_{k-1} + \beta_{k-1}\boldsymbol{p}_{k-1}$                     (28)
    endif
    $\boldsymbol{q}_k = \boldsymbol{N}\boldsymbol{p}_k$                                         (29)
    $\alpha_k = \rho_{k-1}/(\boldsymbol{p}'_k\boldsymbol{q}_k)$                               (30)
    $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k\boldsymbol{p}_k$                               (31)
    $\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k\boldsymbol{q}_k$                               (32)
    if $\boldsymbol{x}_k$ accurate enough then quit
next $k$
```

4.2 Analysis of the algorithm in the AGIS context

Comparing with the simple iteration, we see that the first few steps (23)–(24) (with $k = 1$) are the same, except that the computed vector $\boldsymbol{w}_{k-1}$ is not directly used as an update to $\boldsymbol{x}_{k-1}$. The 'acceleration factor' α_k is computed by means of the vectors $\boldsymbol{r}_{k-1}$, $\boldsymbol{w}_{k-1} = \boldsymbol{p}_k$ and $\boldsymbol{q}_k$. This immediately poses two questions: first, how to compute the full vector $\boldsymbol{r}_{k-1}$ (i.e., including the attitude part); secondly, how to compute $\boldsymbol{q}_k$ by (29), which has no direct counterpart in the simple (or accelerated) iteration. Finally we note that the $\boldsymbol{r}$ vector for the next iteration is computed by the recursion formula (32) rather than directly from $\boldsymbol{r} = \boldsymbol{n} - \boldsymbol{N}\boldsymbol{x}$.[4]

Concerning (29), we note that $\boldsymbol{p}_k$ is very much like an update to $\boldsymbol{x}_{k-1}$, in fact it only differs from the next update by the scalar factor α_k, which presumably is of order unity. Therefore we can regard $\boldsymbol{p}_k$ as a vector of *trial updates* resulting in the *trial right-hand side*

$$\tilde{\boldsymbol{r}}_k = \boldsymbol{n} - \boldsymbol{N}(\boldsymbol{x}_{k-1} + \boldsymbol{p}_k) = (\boldsymbol{n} - \boldsymbol{N}\boldsymbol{x}_{k-1}) - \boldsymbol{N}\boldsymbol{p}_k = \boldsymbol{r}_{k-1} - \boldsymbol{q}_k \tag{33}$$

[4]Using recursion rather than the direct formula saves a lot of calculations but could make the procedure sensitive to the accumulation of rounding errors. One possible remedy could be to 'reset' the residuals with the direct formula after the recursion has been used for a certain number of iterations. However, as discussed in Sect. 8.1 of [6], this could have a negative impact on the convergence – in fact, according to that reference, a stagnation in the convergence is observed in many important situations. Finding a reliable updating procedure for $\boldsymbol{r}$ is one of the features that remain to be investigated.

Since the previous right-hand side $\boldsymbol{r}_{k-1}$ is presumed to be known [it is for example needed in (25)], we obtain the desired vector as

$$\boldsymbol{q}_k = \boldsymbol{r}_{k-1} - \tilde{\boldsymbol{r}}_k \tag{34}$$

The above calculation of the trial right-hand sides corresponds to the first part of the kernel algorithm, i.e. Eq. (22a), only with modified (trial) parameters. The second part of the kernel algorithm, Eq. (22b), has its CG counterpart in (24) for the next iteration. The problem is that the vector used in (24) is $\boldsymbol{r}_{k-1}$, not $\tilde{\boldsymbol{r}}_{k-1}$ as obtained from (33). Therefore we cannot simply insert (22) at the appropriate place of the CG algorithm.

However, this problem can be overcome by carrying the idea of the trial solution one step further. Suppose that instead of (24) we calculate

$$\tilde{\boldsymbol{w}}_{k-1} = \boldsymbol{K}^{-1}\tilde{\boldsymbol{r}}_{k-1} \tag{35}$$

From (32) and (34) we have

$$\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k \boldsymbol{q}_k = \boldsymbol{r}_{k-1} - \alpha_k(\boldsymbol{r}_{k-1} - \tilde{\boldsymbol{r}}_k) = (1-\alpha_k)\boldsymbol{r}_{k-1} + \alpha_k \tilde{\boldsymbol{r}}_k \tag{36}$$

Symbolically pre-multiplying with $\boldsymbol{K}^{-1}$ gives a recursion formula for $\boldsymbol{w}_k$

$$\boldsymbol{w}_k = (1-\alpha_k)\boldsymbol{w}_{k-1} + \alpha_k \tilde{\boldsymbol{w}}_k \tag{37}$$

With this artifact, we can now re-write the CG algorithm as follows:

```
initial guess $\boldsymbol{x}_0$
$\boldsymbol{r}_0 = \boldsymbol{n} - \boldsymbol{N}\boldsymbol{x}_0$                                  (38)
$\boldsymbol{w}_0 = \boldsymbol{K}^{-1}\boldsymbol{r}_0$                                     (39)
$\rho_0 = \boldsymbol{r}'_0\boldsymbol{w}_0$                                         (40)
$\boldsymbol{p}_1 = \boldsymbol{w}_0$                                             (41)
for $k = 1, 2, \ldots$
    $\tilde{\boldsymbol{r}}_k = \boldsymbol{n} - \boldsymbol{N}(\boldsymbol{x}_{k-1} + \boldsymbol{p}_k)$                  (42)
    $\tilde{\boldsymbol{w}}_k = \boldsymbol{K}^{-1}\tilde{\boldsymbol{r}}_k$                                 (43)
    $\boldsymbol{q}_k = \boldsymbol{r}_{k-1} - \tilde{\boldsymbol{r}}_k$                                (44)
    $\alpha_k = \rho_{k-1}/(\boldsymbol{p}'_k\boldsymbol{q}_k)$                            (45)
    $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k\boldsymbol{p}_k$                           (46)
    if $\boldsymbol{x}_k$ accurate enough then quit
    $\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k\boldsymbol{q}_k$                           (47)
    $\boldsymbol{w}_k = (1-\alpha_k)\boldsymbol{w}_{k-1} + \alpha_k\tilde{\boldsymbol{w}}_k$                (48)
    $\rho_k = \boldsymbol{r}'_k\boldsymbol{w}_k$                                     (49)
    $\beta_k = \rho_k/\rho_{k-1}$                                    (50)
    $\boldsymbol{p}_{k+1} = \boldsymbol{w}_k + \beta_k\boldsymbol{p}_k$                           (51)
next $k$
```

We see that (38)–(39) and (42)–(43) now have exactly the same form as the kernel processing (22) and are therefore amenable for implementation in the AGIS framework.

5 Calculation of the normals and the full right-hand side

A remaining issue for the implementation of the CG algorithm is the calculation of the full right-hand side $\boldsymbol{r}$ in (22), which normally would not be obtained in AGIS but which is needed several times in (38)–(51). In order to find out how this can be done it is necessary to analyze the kernel algorithm (22) in greater detail.

Following on to the development in Sects. 1–2, let i and j be the source and attitude interval applicable to observation ℓ and $\boldsymbol{s}_i$, $\boldsymbol{a}_j$ the corresponding *updates* to the source and attitude parameter vectors (both are sub-vectors of $\boldsymbol{x}$). The normalized observation equation is

$$\boldsymbol{F}_{\ell i}\boldsymbol{s}_i + \boldsymbol{G}_{\ell j}\boldsymbol{a}_j = h_\ell \tag{52}$$

where $\boldsymbol{F}_{\ell i} = -(\partial R_\ell/\partial \boldsymbol{s}_i')/\tilde{\sigma_\ell}$, $\boldsymbol{G}_{\ell j} = -(\partial R_\ell/\partial \boldsymbol{a}_j')/\tilde{\sigma_\ell}$, and $h_\ell = R_\ell/\tilde{\sigma_\ell}$.

The matrix $\boldsymbol{S}$ in (7) consists of 5×5 symmetric matrices $\boldsymbol{S}_i$ along the main diagonal:

$$\boldsymbol{S}_i = \sum_{\ell\in L_i} \boldsymbol{F}_{\ell i}'\boldsymbol{F}_{\ell i} \tag{53}$$

where L_i is the set of observations concerned with source i. Similarly, the matrix $\boldsymbol{A}$ consists of symmetric matrices $\boldsymbol{A}_j$ along the main diagonal:

$$\boldsymbol{A}_j = \sum_{\ell\in L_j} \boldsymbol{G}_{\ell j}'\boldsymbol{G}_{\ell j} \tag{54}$$

where L_j is the set of observations in attitude interval j. The link matrix $\boldsymbol{U}$ consists of sub-matrices

$$\boldsymbol{U}_{ij} = \sum_{\ell\in L_i\cap L_j} \boldsymbol{F}_{\ell i}'\boldsymbol{G}_{\ell j} \tag{55}$$

The right-hand side for source i is

$$\boldsymbol{r}_i = \sum_{\ell\in L_i} \boldsymbol{F}_{\ell i}'h_\ell \tag{56}$$

and the right-hand side for attitude interval j is

$$\boldsymbol{r}_j = \sum_{\ell\in L_j} \boldsymbol{G}_{\ell j}'h_\ell \tag{57}$$

In the following we denote by $\hat{\boldsymbol{r}}_j$ the modified right-hand side appearing in the second equation of (11). In terms of the source updates $\boldsymbol{w}_i = \boldsymbol{S}_i^{-1}\boldsymbol{r}_i$ it is

$$\hat{\boldsymbol{r}}_j = \boldsymbol{r}_j - \sum_i \boldsymbol{U}_{ij}'\boldsymbol{w}_i \tag{58}$$

The calculation of $\boldsymbol{w}$ in (22) can now be organized in the following way:

```
for each attitude interval $j$
    $[\boldsymbol{A}_j\ \hat{\boldsymbol{r}}_j] = \boldsymbol{0}$                                  (59)
next $j$
for each source $i$
    $[\boldsymbol{S}_i\ \boldsymbol{r}_i] = \boldsymbol{0}$                                  (60)
    for $\ell \in L_i$
        calculate $\boldsymbol{F}_{\ell i},\ \boldsymbol{G}_{\ell j},\ h_\ell$                      (61)
        $[\boldsymbol{S}_i\ \boldsymbol{r}_i]$ += $\boldsymbol{F}'_{\ell i}[\boldsymbol{F}_{\ell i}\ h_\ell]$                  (62)
    next $\ell$
    solve $[\boldsymbol{S}_i\ \boldsymbol{r}_i] \quad \Rightarrow \quad \boldsymbol{w}_i = \boldsymbol{S}_i^{-1}\boldsymbol{r}_i$            (63)
    for $\ell \in L_i$
        $\hat{h}_\ell = h_\ell - \boldsymbol{F}_{\ell i}\boldsymbol{w}_i$                          (64)
        $[\boldsymbol{A}_j\ \hat{\boldsymbol{r}}_j]$ += $\boldsymbol{G}'_{\ell j}[\boldsymbol{G}_{\ell j}\ \hat{h}_\ell]$                 (65)
    next $\ell$
next $i$
for each attitude interval$j$
    solve $[\boldsymbol{A}_j\ \hat{\boldsymbol{r}}_j] \quad \Rightarrow \quad \boldsymbol{w}_j = \boldsymbol{A}_j^{-1}\hat{\boldsymbol{r}}_j$           (66)
next $j$
```

Here and in the following, $[\boldsymbol{M}\ \boldsymbol{m}_1\ \boldsymbol{m}_2\ \ldots]$ stands for a system of linear equations with matrix $\boldsymbol{M}$ and right-hand sides $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, etc.

This is probably not exactly how AGIS is currently organized. In particular, the modified residuals $\hat{h}_\ell$ in (64) may be recomputed directly from the observations and updated source parameters, rather than by means of $\boldsymbol{F}_{\ell i}$; also, $\boldsymbol{G}_{\ell j}$ may not be computed until it is needed in (65). It would however appear expedient to compute $\boldsymbol{F}_{\ell i}$, $\boldsymbol{G}_{\ell j}$ and h_ℓ at the same time, as in (61) above, since they could share a lot of the calculations, and to avoid recomputing the residuals for the attitude update. In either case, it is seen that the required attitude right-hand sides $\boldsymbol{r}_j$ can be obtained by a modest addition to the above scheme:

```
for each attitude interval $j$
    $[\boldsymbol{A}_j\ \hat{\boldsymbol{r}}_j\ \boldsymbol{r}_j] = \boldsymbol{0}$                                  (67)
next $j$
for each source $i$
    $[\boldsymbol{S}_i\ \boldsymbol{r}_i] = \boldsymbol{0}$                                        (68)
    for $\ell \in L_i$
        calculate $\boldsymbol{F}_{\ell i},\ \boldsymbol{G}_{\ell j},\ h_\ell$                       (69)
        $[\boldsymbol{S}_i\ \boldsymbol{r}_i] \mathrel{+}= \boldsymbol{F}'_{\ell i}[\boldsymbol{F}_{\ell i}\ h_\ell]$                  (70)
    next $\ell$
    solve $[\boldsymbol{S}_i\ \boldsymbol{r}_i] \quad\Rightarrow\quad \boldsymbol{w}_i = \boldsymbol{S}_i^{-1}\boldsymbol{r}_i$              (71)
    for $\ell \in L_i$
        $\hat{h}_\ell = h_\ell - \boldsymbol{F}_{\ell i}\boldsymbol{w}_i$                            (72)
        $[\boldsymbol{A}_j\ \hat{\boldsymbol{r}}_j\ \boldsymbol{r}_j] \mathrel{+}= \boldsymbol{G}'_{\ell j}[\boldsymbol{G}_{\ell j}\ \hat{h}_\ell\ h_\ell]$          (73)
    next $\ell$
next $i$
for each attitude interval$j$
    (partially) solve $[\boldsymbol{A}_j\ \hat{\boldsymbol{r}}_j\ \boldsymbol{r}_j] \quad\Rightarrow\quad \boldsymbol{w}_j = \boldsymbol{A}_j^{-1}\hat{\boldsymbol{r}}_j$   (74)
next $j$
```

6 Conclusions

It appears that the Conjugate Gradient method can be incorporated in the AGIS framework as described by (38)–(51), where the kernel process, formally described by (22), could be implemented as in (67)–(74).

It is proposed that the algorithm is first tested by means of AGISLab.

References

[1] Portell J., Figueras F., Fabricius C., López Martı B., Luri X., Jordi C., Torra J., 2006: *Final revision of the GDAAS2 Large-Scale Test (LST)*, UB-GDAAS2-TN-035

[2] Greenbaum A., 1997: *Iterative Methods for Solving Linear Systems*, SIAM

[3] Lindegren L., 2007: *Convergence properties of AGIS*, GAIA-C3-TN-LU-LL-071-01

[4] Lindegren L., 2007: *Accelerating the convergence of AGIS*, GAIA-C3-TN-LU-LL-074-01

[5] Raison F., Lindegren L., *AGIS Software Testing Report*, GAIA-C3-TR-ESAC-FRN-002-01

[6] van der Vorst H., 2003: *Iterative Krylov Methods for Large Linear Systems*, Cambridge University Press