

Computational aspects of the equivalent-layer technique: review

Vanderlei C. Oliveira Jr.¹, Diego Takahashi¹, André L. A. Reis² and Valéria C. F. Barbosa^{1,*}

¹*Observatório Nacional, Department of Geophysics, Rio de Janeiro, Brasil*

²*Universidade do Estado do Rio de Janeiro, Department of Applied Geology, Rio de Janeiro, Brasil*

Correspondence*: Valéria C.F. Barbosa
valcris@on.br

2 ABSTRACT

3 Equivalent-layer technique is a powerful tool for processing potential-field data in the space
4 domain. However, the greatest hindrance for using the equivalent-layer technique is its high
5 computational cost for processing massive data sets. The large amount of computer memory
6 usage to store the full sensitivity matrix combined with the computational time required for
7 matrix-vector multiplications and to solve the resulting linear system, are the main drawbacks
8 that made unfeasible the use of the equivalent-layer technique for a long time. More recently, the
9 advances in computational power propelled the development of methods to overcome the heavy
10 computational cost associated with the equivalent-layer technique. We present a comprehensive
11 review of the computation aspects concerning the equivalent-layer technique addressing how
12 previous works have been dealt with the computational cost of this technique. Historically, the
13 high computational cost of the equivalent-layer technique has been overcome by using a variety
14 of strategies such as: moving data-window scheme, column- and row-action updates of the
15 sensitivity matrix, reparametrization, sparsity induction of the sensitivity matrix, iterative methods
16 using the full sensitivity matrix, iterative deconvolution by using the concept of block-Toeplitz
17 Toeplitz-block (BTTB) matrices and direct deconvolution. We compute the number of floating-point
18 operations of some of these strategies adopted in the equivalent-layer technique to show their
19 effectiveness in reducing the computational demand. Numerically, we also address the stability of
20 some of these strategies used in the equivalent-layer technique by comparing with the stability
21 via the classic equivalent-layer technique with the zeroth-order Tikhonov regularization. We
22 show that even for the most computationally efficient methods, which can save up to 10^9 flops,
23 the stability of the linear system is maintained. The two most efficient strategies, iterative and
24 direct deconvolutions, can process large datasets quickly and yield good results. However, direct
25 deconvolution has some drawbacks. Real data from Carajás Mineral Province, Brazil, is also
26 used to validate the results showing a potential field transformation.

27 **Keywords:** equivalent layer, gravimetry, fast algorithms, computational cost, stability analysis

1 INTRODUCTION

The equivalent-layer technique has been used by exploration geophysicists for processing potential-field data since the late 1960s (Dampney, 1969). This technique is based on a widely accepted principle, which states that a discrete set of observed potential-field data due to 3D sources can be approximated by that due to a discrete set of virtual sources (such as point masses, dipoles, prisms, doublets). From a theoretical point of view, the equivalent-layer technique is grounded on potential theory (Kellogg, 1967) and consists in considering that the potential field data can be approximated by a linear combination of harmonic functions describing the potential field due to the virtual sources. These sources, commonly called equivalent sources, are arranged on a layer with finite horizontal dimensions and located below the observations. In the classical approach, a linear inverse problem is solved to estimate the physical property of each equivalent source subject to fit the observations. Then, the estimated physical-property distribution on the equivalent layer is used to accomplish the desired potential-field transformation (e.g., interpolation, upward/downward continuation, reduction to the pole). The later step is done by multiplying the estimated physical-property distribution by the matrix of Green's functions associated with the desired potential-field transformation.

Because the linear inverse problem to be solved in the equivalent-layer technique is set up with a full sensitivity matrix, its computational cost strongly depends on the number of potential-field observations and can be very inefficient for dealing with massive data sets. To overcome this problem, computationally efficient methods based on equivalent-layer technique have arose in the late 1980s. To our knowledge, the first method towards improving the efficiency was proposed by Leão and Silva (1989), who used an overlapping moving-window scheme spanning the data set. The strategy adopted in Leão and Silva (1989) involves solving several smaller, regularized linear inverse problems instead of one large problem. This strategy uses a small data window and distributes equivalent sources on a small regular grid at a constant depth below the data surface, with the sources' window extending beyond the boundaries of the data window. Because of the spatial layouts of observed data and equivalent sources in Leão and Silva (1989), the small sensitivity submatrix containing the coordinates of the data and equivalent sources within a window remains constant for all data windows. This holds true regardless of the specific locations of the data and equivalent sources within each window. For each position of the data window, this scheme consists in computing the processed field at the center of the data window only, and the next estimates of the processed field are obtained by shifting the data window across the entire dataset. More recently, Soler and Uieda (2021) extended the method introduced by Leão and Silva (1989) to accommodate irregularly spaced data collected on a non-flat surface. Unlike Leão and Silva (1989), in the generalization proposed by Soler and Uieda (2021), the sensitivity submatrix that includes the coordinates of the data and equivalent sources needs to be computed for each window. Soler and Uieda (2021) developed a computational approach to further enhance the efficiency of the equivalent-layer technique by combining two strategies. The first one — the block-averaging source locations — reduces the number of model parameters and the second strategy — the gradient-boosted algorithm — reduces the size of the linear system to be solved by iteratively fitting the equivalent source model along overlapping windows. It is worth noting that the equivalent-layer strategy of using a moving-window scheme either in Leão and Silva (1989) or in Soler and Uieda (2021) is similar to discrete convolution.

As another strategy to reduce the computational workload of the equivalent-layer technique, some authors have employed column- and row-action updates, which are commonly applied to image reconstruction methods (e.g., Elfving et al., 2017). These methods involve iterative calculations of a single column and a single row of the sensitivity matrix, respectively. Following the strategy column-action update,

71 Cordell (1992) proposed a computational method in which a single equivalent source positioned below a
72 measurement station is iteratively used to compute both the predicted data and residual data for all stations.
73 In Cordell's method, a single column of the sensitivity matrix is calculated per iteration, meaning that
74 a single equivalent source contributes to data fitting in each iteration. Guspí and Novara (2009) further
75 extended Cordell's method by applying it to scattered magnetic observations. Following the strategy of
76 column-action update, Mendonça and Silva (1994) developed an iterative procedure where one data point
77 is incorporated at a time, and a single row of the sensitivity matrix is calculated per iteration. This strategy
78 adopted by Mendonça and Silva (1994) is known as *equivalent data concept*. This concept is based on
79 the principle that certain data points within a dataset are redundant and, as a result, do not contribute to
80 the final solution. On the other hand, there is a subset of observations known as equivalent data, which
81 effectively contributes to the final solution and fits the remaining redundant data. In their work, Mendonça
82 and Silva (1994) adopted an iterative approach to select a substantially smaller subset of equivalent data
83 from the original dataset.

84 The next strategy involves reparametrizing the equivalent layer with the objective of solving a smaller
85 linear inverse problem by reducing the dimension of the model space. Oliveira Jr. et al. (2013) reduced
86 the model parameters by approximating the equivalent-source layer by a piecewise-polynomial function
87 defined on a set of user-defined small equivalent-source windows. The estimated parameters are the
88 polynomial coefficients for each window and they are much smaller than the original number of equivalent
89 sources. By using the subspace method, Mendonça (2020) reparametrizes the equivalent layer, which
90 involves reducing the dimension of the linear system from the original parameter-model space to a lower-
91 dimensional subspace. The subspace bases span the parameter-model space and they are constructed by
92 applying the singular value decomposition to the matrix containing the gridded data.

93 Following the strategy of sparsity induction, Li and Oldenburg (2010) transformed the full sensitivity
94 matrix into a sparse one using orthonormal compactly supported wavelets. Barnes and Lumley (2011)
95 proposed an alternative approach to introduce sparsity based on the use of quadtree discretization to group
96 equivalent sources far from the computation points. Those authors explore the induced sparsity by using
97 specific iterative methods to solve the linear system.

98 The strategy named iterative methods estimates iteratively the parameter vector that represents a
99 distribution over an equivalent layer. Xia and Sprowl (1991) and Xia et al. (1993) have developed efficient
100 iterative algorithms for updating the distribution of physical properties within the equivalent layer in the
101 wavenumber and space domains, respectively. Specifically, in Xia and Sprowl's (1991) method the physical-
102 property distribution is updated by using the ratio between the squared depth to the equivalent source and
103 the gravitational constant multiplied by the residual between the observed and predicted observation at the
104 measurement station. Siqueira et al. (2017) developed an iterative solution where the sensitivity matrix
105 is transformed into a diagonal matrix with constant terms through the use of the *excess mass criterion*
106 and of the positive correlation between the observed gravity data and the masses on the equivalent layer.
107 The fundamentals of the Siqueira et al.'s method is based on the Gauss' theorem (e.g., Kellogg, 1967,
108 p. 43) and the total excess of mass (e.g., Blakely, 1996, p. 60). All these iterative methods use the full
109 and dense sensitivity matrix to calculate the predicted data and residual data in the whole survey data per
110 iteration. Hence, the iterative methods proposed by Xia and Sprowl (1991), Xia et al. (1993) and Siqueira
111 et al. (2017) neither compress nor reparametrize the sensitivity matrix. Jirigalatu and Ebbing (2019) also
112 proposed an iterative equivalent layer that uses the full and dense sensitivity matrix. However, in their
113 approach, Jirigalatu and Ebbing (2019) efficiently compute the predicted data and residual data for the
114 entire survey per iteration in the wavenumber domain.

115 Following the strategy of the iterative deconvolution, Takahashi et al. (2020, 2022) developed fast and
116 effective equivalent-layer techniques for processing, respectively, gravity and magnetic data by modifying
117 the forward modeling to estimate the physical-property distribution over the equivalent layer through a
118 2D discrete fast convolution. These methods took advantage of the Block-Toeplitz Toeplitz-block (BTTB)
119 structure of the sensitivity matrices, allowing them to be calculated by using only their first column. In
120 practice, the forward modeling uses a single equivalent source, which significantly reduces the required
121 RAM memory.

122 The method introduced by Takahashi et al. (2020, 2022) can be reformulated to eliminate the need for
123 conjugate gradient iterations. This reformulation involves employing a *direct deconvolution* approach (e.g.,
124 Aster et al., 2019, p. 220) with *Wiener filter* (e.g., Gonzalez and Woods, 2002, p. 263).

125 Here, we present a comprehensive review of diverse strategies to solve the linear system of the equivalent
126 layer alongside an analysis of the computational cost and stability of these strategies. To do this analysis,
127 we are using the floating-point operations count to evaluate the performance of a selected set of methods.
128 To test the stability, we are using the linear system sensitivity to noise as a comparison parameter for the
129 fastest of these methods alongside the classical normal equations. A potential-field transformation will also
130 be used to evaluate the quality of the equivalent sources estimation results using both synthetic and real
131 data from Carajás Mineral Province, Brazil.

2 FUNDAMENTALS

132 Let \mathbf{d} be a $D \times 1$ vector, whose i -th element d_i is the observed potential field at the position (x_i, y_i, z_i) ,
 133 $i \in \{1 : D\}$, of a topocentric Cartesian system with x , y and z axes pointing to north, east and down,
 134 respectively. Consider that d_i can be satisfactorily approximated by a harmonic function

$$f_i = \sum_{j=1}^P g_{ij} p_j, \quad i \in \{1 : D\}, \quad (1)$$

135 where, p_j represents the scalar physical property of a virtual source (i.e., monopole, dipole, prism) located
 136 at (x_j, y_j, z_j) , $j \in \{1 : P\}$ and

$$g_{ij} \equiv g(x_i - x_j, y_i - y_j, z_i - z_j), \quad z_i < \min\{z_j\}, \quad \forall i \in \{1 : D\}, \quad (2)$$

137 is a harmonic function, where $\min\{z_j\}$ denotes the minimum z_j , or the vertical coordinate of the shallowest
 138 virtual source. These virtual sources are called *equivalent sources* and they form an *equivalent layer*. In
 139 matrix notation, the potential field produced by all equivalent sources at all points (x_i, y_i, z_i) , $i \in \{1 : D\}$,
 140 is given by:

$$\mathbf{f} = \mathbf{G}\mathbf{p}, \quad (3)$$

141 where \mathbf{p} is a $P \times 1$ vector with j -th element p_j representing the scalar physical property of the j -th
 142 equivalent source and \mathbf{G} is a $D \times P$ matrix with element g_{ij} given by equation 2.

143 The equivalent-layer technique consists in solving a linear inverse problem to determine a parameter
 144 vector \mathbf{p} leading to a predicted data vector \mathbf{f} (equation 3) *sufficiently close to* the observed data vector \mathbf{d} ,
 145 whose i -th element d_i is the observed potential field at (x_i, y_i, z_i) . The notion of *closeness* is intrinsically
 146 related to the concept of *vector norm* (e.g., Golub and Van Loan, 2013, p. 68) or *measure of length* (e.g.,
 147 Menke, 2018, p. 41). Because of that, almost all methods for determining \mathbf{p} actually estimate a parameter
 148 vector $\tilde{\mathbf{p}}$ minimizing a length measure of the difference between \mathbf{f} and \mathbf{d} (see subsection 3.1). Given an
 149 estimate $\tilde{\mathbf{p}}$, it is then possible to compute a potential field transformation

$$\mathbf{t} = \mathbf{A}\tilde{\mathbf{p}}, \quad (4)$$

150 where \mathbf{t} is a $T \times 1$ vector with k -th element t_k representing the transformed potential field at the position
 151 (x_k, y_k, z_k) , $k \in \{1 : T\}$, and

$$a_{kj} \equiv a(x_k - x_j, y_k - y_j, z_k - z_j), \quad z_k < \min\{z_j\}, \quad \forall k \in \{1 : T\}, \quad (5)$$

152 is a harmonic function representing the kj -th element of the $T \times P$ matrix \mathbf{A} .

153 2.1 Spatial distribution and total number of equivalent sources

154 There is no well-established criteria to define the optimum number P or the spatial distribution of the
 155 equivalent sources. We know that setting an equivalent layer with more (less) sources than potential-field
 156 data usually leads to an underdetermined (overdetermined) inverse problem (e.g., Menke, 2018, p. 52–53).
 157 Concerning the spatial distribution of the equivalent sources, the only condition is that they must rely on a
 158 surface that is located below and does not cross that containing the potential field data. Soler and Uieda
 159 (2021) present a practical discussion about this topic.

160 From a theoretical point of view, the equivalent layer reproducing a given potential field data set cannot
 161 cross the true gravity or magnetic sources. This condition is a consequence of recognizing that the equivalent
 162 layer is essentially an indirect solution of a boundary value problem of potential theory (e.g., Roy, 1962;
 163 Zidarov, 1965; Dampney, 1969; Li et al., 2014; Reis et al., 2020). In practical applications, however, there
 164 is no guarantee that this condition is satisfied. Actually, it is widely known from practical experience (e.g.,
 165 Gonzalez et al., 2022) that the equivalent-layer technique works even for the case in which the layer cross
 166 the true sources.

167 Regarding the depth of the equivalent layer, Dampney (1969) proposed a criterion based on horizontal data
 168 sampling, suggesting that the equivalent-layer depth should be between two and six times the horizontal grid
 169 spacing, considering evenly spaced data. However, when dealing with a survey pattern that has unevenly
 170 spaced data, Reis et al. (2020) adopted an alternative empirical criterion. According to their proposal,
 171 the depth of the equivalent layer should range from two to three times the spacing between adjacent
 172 flight lines. The criteria of Dampney (1969) and Reis et al. (2020) are valid for planar equivalent layers.
 173 Cordell (1992) have proposed and an alternative criterion for scattered data that leads to an undulating
 174 equivalent layer. This criterion have been slightly modified by Guspí et al. (2004), Guspí and Novara
 175 (2009) and Soler and Uieda (2021), for example, and consists in setting one equivalent source below
 176 each datum at a depth proportional to the horizontal distance to the nearest neighboring data points. Soler
 177 and Uieda (2021) have compared different strategies for defining the equivalent sources depth for the
 178 specific problem of interpolating gravity data, but they have not found significant differences between them.
 179 Regarding the horizontal layout, Soler and Uieda (2021) proposed the block-averaged sources locations
 180 in which the survey area is divided into horizontal blocks and one single equivalent source is assigned
 181 to each block. The horizontal coordinates of the single source in a given block is defined by the average
 182 horizontal coordinates of the observation points at the block. According to Soler and Uieda (2021), this
 183 block-averaged layout may prevent aliasing of the interpolated values, specially when the observations
 184 are unevenly sampled. This strategy also reduces the number of equivalent sources without affecting the
 185 accuracy of the potential-field interpolation. Besides, it reduces the computational load for estimating the
 186 physical property on the equivalent layer.

187 2.2 Matrix G

188 Generally, the harmonic function g_{ij} (equation 2) is defined in terms of the inverse distance between the
 189 observation point (x_i, y_i, z_i) and the j -th equivalent source at (x_j, y_j, z_j) ,

$$\frac{1}{r_{ij}} \equiv \frac{1}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}}, \quad (6)$$

190 or by its partial derivatives of first and second orders, respectively given by

$$\partial_\alpha \frac{1}{r_{ij}} \equiv \frac{-(\alpha_i - \alpha_j)}{r_{ij}^3}, \quad \alpha \in \{x, y, z\}, \quad (7)$$

191 and

$$\partial_{\alpha\beta} \frac{1}{r_{ij}} \equiv \begin{cases} \frac{3(\alpha_i - \alpha_j)^2}{r_{ij}^5}, & \alpha = \beta, \\ \frac{3(\alpha_i - \alpha_j)(\beta_i - \beta_j)}{r_{ij}^5} - \frac{1}{r_{ij}^3}, & \alpha \neq \beta, \end{cases} \quad \alpha, \beta \in \{x, y, z\}. \quad (8)$$

192 In this case, the equivalent layer is formed by punctual sources representing monopoles or dipoles (e.g.,
 193 Dampney, 1969; Emilia, 1973; Leão and Silva, 1989; Cordell, 1992; Oliveira Jr. et al., 2013; Siqueira et al.,
 194 2017; Reis et al., 2020; Takahashi et al., 2020; Soler and Uieda, 2021; Takahashi et al., 2022). Another
 195 common approach consists in not defining g_{ij} by using equations 6–8, but other harmonic functions
 196 obtained by integrating them over the volume of regular prisms (e.g., Li and Oldenburg, 2010; Barnes and
 197 Lumley, 2011; Li et al., 2014; Jirgalatu and Ebbing, 2019). There are also some less common approaches
 198 defining the harmonic function g_{ij} (equation 2) as the potential field due to plane faces with constant
 199 physical property (Hansen and Miyazaki, 1984), doublets (Silva, 1986) or by computing the double
 200 integration of the inverse distance function with respect to z (Guspí and Novara, 2009).

201 A common assumption for most of the equivalent-layer methods is that the harmonic function g_{ij}
 202 (equation 2) is independent on the actual physical relationship between the observed potential field and
 203 their true sources (e.g., Cordell, 1992; Guspí and Novara, 2009; Li et al., 2014). Hence, g_{ij} can be
 204 defined according to the problem. The only condition imposed to this function is that it decays to zero
 205 as the observation point (x_i, y_i, z_i) goes away from the position (x_j, y_j, z_j) of the j -th equivalent source.
 206 However, several methods use a function g_{ij} that preserves the physical relationship between the observed
 207 potential field and their true sources. For the case in which the observed potential field is gravity data, g_{ij}
 208 is commonly defined as a component of the gravitational field produced at (x_i, y_i, z_i) by a point mass or
 209 prism located at (x_j, y_j, z_j) , with unit density. On the other hand, g_{ij} is commonly defined as a component
 210 of the magnetic induction field produced at (x_i, y_i, z_i) by a dipole or prism located at (x_j, y_j, z_j) , with unit
 211 magnetization intensity, when the observed potential field is magnetic data.

212 The main challenge in the equivalent-layer technique is the computational complexity associated with
 213 handling large datasets. This complexity arises because the sensitivity matrix \mathbf{G} (equation 3) is dense
 214 regardless of the harmonic function g_{ij} (equation 2) employed. In the case of scattered potential-field
 215 data, the structure of \mathbf{G} is not well-defined, regardless of the spatial distribution of the equivalent sources.
 216 However, in a specific scenario where (i) each potential-field datum is directly associated with a single
 217 equivalent source located directly below it, and (ii) both the data and sources are based on planar and
 218 regularly spaced grids, Takahashi et al. (2020, 2022) demonstrate that \mathbf{G} exhibits a block-Toeplitz Toeplitz-
 219 block (BTTB) structure. In such cases, the product of \mathbf{G} and an arbitrary vector can be efficiently computed
 220 using a 2D fast Fourier transform as a discrete convolution.

3 LINEAR INVERSE PROBLEM OF EQUIVALENT-LAYER TECHNIQUE

221 3.1 General formulation

222 A general formulation for almost all equivalent-layer methods can be achieved by first considering that
 223 the $P \times 1$ parameter vector \mathbf{p} (equation 3) can be reparameterized into a $Q \times 1$ vector \mathbf{q} according to:

$$\mathbf{p} = \mathbf{H} \mathbf{q}, \quad (9)$$

224 where \mathbf{H} is a $P \times Q$ matrix. The predicted data vector \mathbf{f} (equation 3) can then be rewritten as follows:

$$\mathbf{f} = \mathbf{G} \mathbf{H} \mathbf{q}. \quad (10)$$

225 Note that the original parameter vector \mathbf{p} is defined in a P -dimensional space whereas the reparameterized
 226 parameter vector \mathbf{q} (equation 9) lies in a Q -dimensional space. For convenience, we use the terms P -space
 227 and Q -space to designate them.

228 In this case, the problem of estimating a parameter vector $\tilde{\mathbf{p}}$ minimizing a length measure of the difference
 229 between \mathbf{f} (equation 3) and \mathbf{d} is replaced by that of estimating an auxiliary vector $\tilde{\mathbf{q}}$ minimizing the goal
 230 function

$$\Gamma(\mathbf{q}) = \Phi(\mathbf{q}) + \mu \Theta(\mathbf{q}), \quad (11)$$

231 which is a combination of particular measures of length given by

$$\Phi(\mathbf{q}) = (\mathbf{d} - \mathbf{f})^\top \mathbf{W}_d (\mathbf{d} - \mathbf{f}), \quad (12)$$

232 and

$$\Theta(\mathbf{q}) = (\mathbf{q} - \bar{\mathbf{q}})^\top \mathbf{W}_q (\mathbf{q} - \bar{\mathbf{q}}), \quad (13)$$

233 where the regularization parameter μ is a positive scalar controlling the trade-off between the data-misfit
 234 function $\Phi(\mathbf{q})$ and the regularization function $\Theta(\mathbf{q})$; \mathbf{W}_d is a $D \times D$ symmetric matrix defining the relative
 235 importance of each observed datum d_i ; \mathbf{W}_q is a $Q \times Q$ symmetric matrix imposing prior information on \mathbf{q} ;
 236 and $\bar{\mathbf{q}}$ is a $Q \times 1$ vector of reference values for \mathbf{q} that satisfies

$$\bar{\mathbf{p}} = \mathbf{H} \bar{\mathbf{q}}, \quad (14)$$

237 where $\bar{\mathbf{p}}$ is a $P \times 1$ vector containing reference values for the original parameter vector \mathbf{p} .

238 After obtaining an estimate $\tilde{\mathbf{q}}$ for the reparameterized parameter vector \mathbf{q} (equation 9), the estimate $\tilde{\mathbf{p}}$ for
 239 the original parameter vector (equation 3) is computed by

$$\tilde{\mathbf{p}} = \mathbf{H} \tilde{\mathbf{q}}. \quad (15)$$

240 The reparameterized vector $\tilde{\mathbf{q}}$ is obtained by first computing the gradient of $\Gamma(\mathbf{q})$,

$$\nabla \Gamma(\mathbf{q}) = -2 \mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d (\mathbf{d} - \mathbf{f}) + 2\mu \mathbf{W}_q (\mathbf{q} - \bar{\mathbf{q}}). \quad (16)$$

241 Then, by considering that $\nabla \Gamma(\tilde{\mathbf{q}}) = \mathbf{0}$ (equation 16), where $\mathbf{0}$ is a vector of zeros, as well as adding and
 242 subtracting the term $(\mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d \mathbf{G} \mathbf{H}) \bar{\mathbf{q}}$, we obtain

$$\tilde{\boldsymbol{\delta}}_q = \mathbf{B} \boldsymbol{\delta}_d, \quad (17)$$

243 where

$$\tilde{\mathbf{q}} = \tilde{\boldsymbol{\delta}}_q + \bar{\mathbf{q}}, \quad (18)$$

$$\boldsymbol{\delta}_d = \mathbf{d} - \mathbf{G} \mathbf{H} \bar{\mathbf{q}}, \quad (19)$$

$$\mathbf{B} = (\mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d \mathbf{G} \mathbf{H} + \mu \mathbf{W}_q)^{-1} \mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d, \quad (20)$$

246 or, equivalently (Menke, 2018, p. 62),

$$\mathbf{B} = \mathbf{W}_q^{-1} \mathbf{H}^\top \mathbf{G}^\top (\mathbf{G} \mathbf{H} \mathbf{W}_q^{-1} \mathbf{H}^\top \mathbf{G}^\top + \mu \mathbf{W}_d^{-1})^{-1}. \quad (21)$$

247 Evidently, we have considered that all inverses exist in equations 20 and 21.

248 The $Q \times D$ matrix \mathbf{B} defined by equation 20 is commonly used for the case in which $D > Q$, i.e., when
 249 there are more data than parameters (overdetermined problems). In this case, we consider that the estimate

250 $\tilde{\mathbf{q}}$ is obtained by solving the following linear system for $\tilde{\boldsymbol{\delta}}_q$ (equation 18):

$$\left(\mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d \mathbf{G} \mathbf{H} + \mu \mathbf{W}_q \right) \tilde{\boldsymbol{\delta}}_q = \mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d \boldsymbol{\delta}_d . \quad (22)$$

251 On the other hand, for the cases in which $D < Q$ (underdetermined problems), matrix \mathbf{B} is usually defined
252 according to equation 21. In this case, the general approach involves estimating $\tilde{\mathbf{q}}$ in two steps. The first
253 consists in solving a linear system for a dummy vector, which is subsequently used to compute $\tilde{\mathbf{q}}$ by a
254 matrix-vector product as follows:

$$\begin{aligned} \left(\mathbf{G} \mathbf{H} \mathbf{W}_q^{-1} \mathbf{H}^\top \mathbf{G}^\top + \mu \mathbf{W}_d^{-1} \right) \mathbf{u} &= \boldsymbol{\delta}_d , \\ \tilde{\boldsymbol{\delta}}_q &= \mathbf{W}_q^{-1} \mathbf{H}^\top \mathbf{G}^\top \mathbf{u} \end{aligned} \quad (23)$$

255 where \mathbf{u} is a dummy vector. After obtaining $\tilde{\boldsymbol{\delta}}_q$ (equations 22 and 23), the estimate $\tilde{\mathbf{q}}$ is computed with
256 equation 18.

257 3.2 Formulation without reparameterization

258 Note that, for the particular case in which $\mathbf{H} = \mathbf{I}_P$ (equation 9), where \mathbf{I}_P is the identity of order P ,
259 $P = Q$, $\mathbf{p} = \mathbf{q}$, $\bar{\mathbf{p}} = \bar{\mathbf{q}}$ (equation 14) and $\tilde{\mathbf{p}} = \tilde{\mathbf{q}}$ (equation 15). In this case, the linear system (equations 22
260 and 23) is directly solved for

$$\tilde{\boldsymbol{\delta}}_p = \tilde{\mathbf{p}} - \bar{\mathbf{p}} , \quad (24)$$

261 instead of $\tilde{\boldsymbol{\delta}}_q$ (equation 18).

262 3.3 Linear system solvers

263 According to their properties, the linear systems associated with over and underdetermined problems
264 (equations 22 and 23) can be solved by using *direct methods* such as LU, Cholesky or QR factorization, for
265 example (Golub and Van Loan, 2013, sections 3.2, 4.2 and 5.2). These methods involve factorizing the
266 linear system matrix in a product of “simple” matrices (i.e., triangular, diagonal or orthogonal). Here, we
267 consider the *Cholesky factorization*, (Golub and Van Loan, 2013, p. 163).

268 Let us consider a real linear system $\mathbf{M} \mathbf{x} = \mathbf{y}$, where \mathbf{M} is a symmetric and positive definite matrix
269 (Golub and Van Loan, 2013, p. 159). In this case, the Cholesky factorization consists in computing

$$\mathbf{M} = \mathcal{G} \mathcal{G}^\top , \quad (25)$$

270 where \mathcal{G} is a lower triangular matrix called *Cholesky factor* and having positive diagonal entries. Given \mathcal{G} ,
271 the original linear system is replaced by two triangular systems, as follows:

$$\begin{aligned} \mathcal{G} \mathbf{s} &= \mathbf{y} \\ \mathcal{G}^\top \mathbf{x} &= \mathbf{s} \end{aligned} \quad (26)$$

272 where \mathbf{s} is a dummy vector. For the overdetermined problem (equation 22), $\mathbf{M} =$
273 $(\mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d \mathbf{G} \mathbf{H} + \mu \mathbf{W}_q)$, $\mathbf{x} = \tilde{\boldsymbol{\delta}}_q$ and $\mathbf{y} = (\mathbf{H}^\top \mathbf{G}^\top \mathbf{W}_d \boldsymbol{\delta}_d)$. For the underdetermined problem
274 (equation 23), $\mathbf{M} = (\mathbf{G} \mathbf{H} \mathbf{W}_q^{-1} \mathbf{H}^\top \mathbf{G}^\top + \mu \mathbf{W}_d^{-1})$, $\mathbf{x} = \mathbf{u}$ and $\mathbf{y} = \boldsymbol{\delta}_d$.

275 The use of direct methods for solving large linear systems may be problematic due to computer (i) storage
 276 of large matrices and (ii) time to perform matrix operations. This problem may be specially complicated in
 277 equivalent-layer technique for the cases in which the sensitivity matrix \mathbf{G} does not have a well-defined
 278 structure (sec. 2.2)

279 These problems can be overcome by solving the linear system using an iterative method. These methods
 280 produce a sequence of vectors that typically converge to the solution at a reasonable rate. The main
 281 computational cost associated with these methods is usually some matrix-vector products per iteration. The
 282 *conjugate gradient* (CG) is a very popular iterative method for solving linear systems in equivalent-layer
 283 methods. This method was originally developed to solve systems having a square and positive definite
 284 matrix. There are two adapted versions of the CG method. The first is called *conjugate gradient normal*
 285 *equation residual* (CNR) Golub and Van Loan (2013, sec. 11.3) or *conjugate gradient least squares*
 286 (CGLS) (Aster et al., 2019, p. 165) and is used to solve overdetermined problems (equation 22). The second
 287 is called *conjugate gradient normal equation error* (CGNE) method Golub and Van Loan (2013, sec. 11.3)
 288 and is used to solve the underdetermined problems (equation 23). Algorithm 1 outlines the CGLS method
 289 applied to the overdetermined problem (equation 22).

4 FLOATING-POINT OPERATIONS

290 Two important factors affecting the efficiency of a given matrix algorithm are the storage and amount of
 291 required arithmetic. Here, we quantify this last factor associated with different computational strategies to
 292 solve the linear system of the equivalent-layer technique (section 7). To do it, we opted by counting *flops*,
 293 which are floating point additions, subtractions, multiplications or divisions (Golub and Van Loan, 2013,
 294 p. 12–14). This is a non-hardware dependent approach that allows us to do direct comparison between
 295 different equivalent-layer methods. Most of the flops count used here can be found in Golub and Van Loan
 296 (2013, p. 12, 106, 107 and 164).

297 Let us consider the case in which the overdetermined problem (equation 22) is solved by Cholesky
 298 factorization (equations 25 and 26) directly for the parameter vector $\tilde{\mathbf{p}}$ by considering the particular case in
 299 which $\mathbf{H} = \mathbf{I}_P$ (equation 9 and subsection 3.2), $\mu = 0$ (equation 11), $\mathbf{W}_d = \mathbf{I}_D$ (equation 12) and $\bar{\mathbf{p}} = \mathbf{0}$
 300 (equation 14), where \mathbf{I}_P and \mathbf{I}_D are the identities of order P and D , respectively. Based on the information
 301 provided in table 1, the total number of flops can be determined by aggregating the flops required for
 302 various computations. These computations include the matrix-matrix and matrix-vector products $\mathbf{G}^\top \mathbf{G}$
 303 and $\mathbf{G}^\top \mathbf{d}$, the Cholesky factor \mathcal{G} , and the solution of triangular systems. Thus, we can express the total
 304 number of flops as follows:

$$f_{\text{Cholesky}} = 1/3D^3 + 2D^2 + 2(P^2 + P)D. \quad (27)$$

305 The same particular overdetermined problem can be solved by using the CGLS method (Algorithm 1).
 306 In this case, we use table 1 again to combine the total number of flops associated with the matrix-vector
 307 and inner products defined in line 3, before starting the iteration, and the 3 saxpys, 2 inner products and 2
 308 matrix-vector products per iteration (lines 7 – 12). By considering a maximum number of iterations ITMAX,
 309 we obtain

$$f_{\text{CGLS}} = 2P(D + 1) + \text{ITMAX} [2P(2D + 3) + 4D]. \quad (28)$$

310 The same approach used to deduce equations 27 and 28 is applied to compute the total number of flops for
 311 the selected equivalent-layer methods discussed in section 7.

To simplify our analysis, we do not consider the number of flops required to compute the sensitivity matrix \mathbf{G} (equation 3) or the matrix \mathbf{A} associated with a given potential-field transformation (equation 4) because they depend on the specific harmonic functions g_{ij} and a_{ij} (equations 2 and 5). We also neglect the required flops to compute \mathbf{H} , \mathbf{W}_d , \mathbf{W}_q (equations 9, 12 and 13), \bar{p} (equation 14), retrieve $\tilde{\mathbf{q}}$ from $\tilde{\delta}_q$ (equation 18) and computing δ_d (equation 19).

5 NUMERICAL STABILITY

All equivalent-layer methods aim at obtaining an estimate $\tilde{\mathbf{p}}$ for the parameter vector \mathbf{p} (equation 3), which contains the physical property of the equivalent sources. Some methods do it by first obtaining an estimate $\tilde{\mathbf{q}}$ for the reparameterized parameter vector \mathbf{q} (equation 9) and then using it to obtain $\tilde{\mathbf{p}}$ (equation 15). The stability of a solution $\tilde{\mathbf{p}}$ against noise in the observed data is rarely addressed. Here, we follow the numerical stability analysis presented in Siqueira et al. (2017).

For a given equivalent-layer method (section 7), we obtain an estimate $\tilde{\mathbf{p}}$ assuming noise-free potential-field data \mathbf{d} . Then, we create L different noise-corrupted data \mathbf{d}^ℓ , $\ell \in \{1 : L\}$, by adding L different sequences of pseudorandom Gaussian noise to \mathbf{d} , all of them having zero mean. From each \mathbf{d}^ℓ , we obtain an estimate $\tilde{\mathbf{p}}^\ell$. Regardless of the particular equivalent-layer method used, the following inequality (Aster et al., 2019, p. 66) holds true:

$$\Delta p^\ell \leq \kappa \Delta d^\ell, \quad \ell \in \{1 : L\}, \quad (29)$$

where κ is the constant of proportionality between the model perturbation

$$\Delta p^\ell = \frac{\|\tilde{\mathbf{p}}^\ell - \tilde{\mathbf{p}}\|}{\|\tilde{\mathbf{p}}\|}, \quad \ell \in \{1 : L\}, \quad (30)$$

and the data perturbation

$$\Delta d^\ell = \frac{\|\mathbf{d}^\ell - \mathbf{d}\|}{\|\mathbf{d}\|}, \quad \ell \in \{1 : L\}, \quad (31)$$

with $\|\cdot\|$ representing the Euclidean norm. The constant κ acts as the condition number associated with the pseudo-inverse in a given linear inversion. The larger (smaller) the value of κ , the more unstable (stable) is the estimated solution. Because of that, we designate κ as *stability parameter*. Equation 29 shows a linear relationship between the model perturbation Δp^ℓ and the data perturbation Δd^ℓ (equations 30 and 31). We estimate the κ (equation 29) associated with a given equivalent-layer method as the slope of the straight line fitted to the *numerical stability curve* formed by the L points $(\Delta p^\ell, \Delta d^\ell)$.

6 NOTATION FOR SUBVECTORS AND SUBMATRICES

Here, we use a notation inspired on that presented by Van Loan (1992, p. 4) to represent subvectors and submatrices. Subvectors of \mathbf{d} , for example, are specified by $\mathbf{d}[i]$, where i is a list of integer numbers that “pick out” the elements of \mathbf{d} forming the subvector $\mathbf{d}[i]$. For example, $i = (1, 6, 4, 6)$ gives the subvector $\mathbf{d}[i] = [d_1 \ d_6 \ d_4 \ d_6]^\top$. Note that the list i of indices may be sorted or not and it may also have repeated indices. For the particular case in which the list has a single element $i = (i)$, then it can be used to extract the i -th element $d_i \equiv \mathbf{d}[i]$ of \mathbf{d} . Sequential lists can be represented by using the colon notation. We consider two types of sequential lists. The first has starting index is smaller than the final index and increment of 1.

The second has starting index is greater than the final index and increment of -1 . For example,

$$\begin{aligned}\mathbf{i} = (3 : 8) &\Leftrightarrow \mathbf{d}[\mathbf{i}] = [d_3 \ d_4 \ \dots \ d_8]^\top \\ \mathbf{i} = (8 : 3) &\Leftrightarrow \mathbf{d}[\mathbf{i}] = [d_8 \ d_7 \ \dots \ d_3]^\top \\ \mathbf{i} = (: 8) &\Leftrightarrow \mathbf{d}[\mathbf{i}] = [d_1 \ d_2 \ \dots \ d_8]^\top \\ \mathbf{i} = (3 :) &\Leftrightarrow \mathbf{d}[\mathbf{i}] = [d_3 \ d_4 \ \dots \ d_D]^\top\end{aligned}$$

335 where D is the number of elements forming \mathbf{d} .

The notation above can also be used to define submatrices of a $D \times P$ matrix \mathbf{G} . For example, $\mathbf{i} = (2, 7, 4, 6)$ and $\mathbf{j} = (1, 3, 8)$ lead to the submatrix

$$\mathbf{G}[\mathbf{i}, \mathbf{j}] = \begin{bmatrix} g_{21} & g_{23} & g_{28} \\ g_{71} & g_{73} & g_{78} \\ g_{41} & g_{43} & g_{48} \\ g_{61} & g_{63} & g_{68} \end{bmatrix}.$$

Note that, in this case, the lists \mathbf{i} and \mathbf{j} “pick out”, respectively, the rows and columns of \mathbf{G} that form the submatrix $\mathbf{G}[\mathbf{i}, \mathbf{j}]$. The i -th row of \mathbf{G} is given by the $1 \times P$ vector $\mathbf{G}[i, :]$. Similarly, the $D \times 1$ vector $\mathbf{G}[:, j]$ represents the j -th column. Finally, we may use the colon notation to define the following submatrix:

$$\mathbf{i} = (2 : 5), \mathbf{j} = (3 : 7) \Leftrightarrow \mathbf{G}[\mathbf{i}, \mathbf{j}] = \begin{bmatrix} g_{23} & g_{24} & g_{25} & g_{26} & g_{27} \\ g_{33} & g_{34} & g_{35} & g_{36} & g_{37} \\ g_{43} & g_{44} & g_{45} & g_{46} & g_{47} \\ g_{53} & g_{54} & g_{55} & g_{56} & g_{57} \end{bmatrix},$$

336 which contains the contiguous elements of \mathbf{G} from rows 2 to 5 and from columns 3 to 7.

7 COMPUTATIONAL STRATEGIES

337 The linear inverse problem of the equivalent-layer technique (section 3) for the case in which there are
338 large volumes of potential-field data requires dealing with:

- 339 (i) the large computer memory to store large and full matrices;
- 340 (ii) the long computation time to multiply a matrix by a vector; and
- 341 (iii) the long computation time to solve a large linear system of equations.

342 Here, we review some strategies aiming at reducing the computational cost of the equivalent-layer technique.
343 We quantify the computational cost by using flops (section 4) and compare the results with those obtained
344 for Cholesky factorization and CGLS (equations 27 and 28). We focus on the overall strategies used by the
345 selected methods.

346 7.1 Moving window

347 The initial approach to enhance the computational efficiency of the equivalent-layer technique is
348 commonly denoted *moving window* and involves first splitting the observed data $d_i, i \in \{1 : D\}$, into
349 M overlapping subsets (or data windows) formed by D^m data each, $m \in \{1 : M\}$. The data inside the

350 m -th window are usually adjacent to each other and have indices defined by an integer list \mathbf{i}^m having
 351 D^m elements. The number of data D^m forming the data windows are not necessarily equal to each other.
 352 Each data window has a $D^m \times 1$ observed data vector $\mathbf{d}^m \equiv \mathbf{d}[\mathbf{i}^m]$. The second step consists in defining
 353 a set of P equivalent sources with scalar physical property p_j , $j \in \{1 : P\}$, and also split them into M
 354 overlapping subsets (or source windows) formed by P^m data each, $m \in \{1 : M\}$. The sources inside the
 355 m -th window have indices defined by an integer list \mathbf{j}^m having P^m elements. Each source window has a
 356 $P^m \times 1$ parameter vector \mathbf{p}^m and is located right below the corresponding m -th data window. Then, each
 357 $\mathbf{d}^m \equiv \mathbf{d}[\mathbf{i}^m]$ is approximated by

$$\mathbf{f}^m = \mathbf{G}^m \mathbf{p}^m, \quad (32)$$

358 where $\mathbf{G}^m \equiv \mathbf{G}[\mathbf{i}^m, \mathbf{j}^m]$ is a submatrix of \mathbf{G} (equation 3) formed by the elements computed with equation
 359 2 using only the data and equivalent sources located inside the window m -th. The main idea of the moving-
 360 window approach is using the $\tilde{\mathbf{p}}^m$ estimated for each window to obtain (i) an estimate $\tilde{\mathbf{p}}$ of the parameter
 361 vector for the entire equivalent layer or (ii) a given potential-field transformation \mathbf{t} (equation 4). The main
 362 advantages of this approach is that (i) the estimated parameter vector $\tilde{\mathbf{p}}$ or transformed potential field are
 363 not obtained by solving the full, but smaller linear systems and (ii) the full matrix \mathbf{G} (equation 3) is never
 364 stored.

365 Leão and Silva (1989) presented a pioneer work using the moving-window approach. Their method
 366 requires a regularly-spaced grid of observed data on a horizontal plane z_0 . The data windows are defined by
 367 square local grids of $\sqrt{D'} \times \sqrt{D'}$ adjacent points, all of them having the same number of points D' . The
 368 equivalent sources in the m -th data window are located below the observation plane, at a constant vertical
 369 distance Δz_0 . They are arranged on a regular grid of $\sqrt{P'} \times \sqrt{P'}$ adjacent points following the same
 370 grid pattern of the observed data. The local grid of sources for all data windows have the same number
 371 of elements P' . Besides, they are vertically aligned, but expands the limits of their corresponding data
 372 windows, so that $D' < P'$. Because of this spatial configuration of observed data and equivalent sources,
 373 we have that $\mathbf{G}^m = \mathbf{G}'$ (equation 32) for all data windows (i.e., $\forall m \in \{1 : M\}$), where \mathbf{G}' is a $D' \times P'$
 374 constant matrix.

375 By omitting the normalization strategy used by Leão and Silva (1989), their method consists in directly
 376 computing the transformed potential field t_c^m at the central point $(x_c^m, y_c^m, z_0 + \Delta z_0)$ of each data window
 377 as follows:

$$t_c^m = (\mathbf{a}')^\top \mathbf{B}' \mathbf{d}^m, \quad m \in \{1 : M\}, \quad (33)$$

378 where \mathbf{a}' is a $P' \times 1$ vector with elements computed by equation 5 by using all equivalent sources in the
 379 m -th window and only the coordinate of the central point in the m -th data window and

$$\mathbf{B}' = (\mathbf{G}')^\top \left[\mathbf{G}' (\mathbf{G}')^\top + \mu \mathbf{I}_{D'} \right]^{-1} \quad (34)$$

380 is a particular case of matrix \mathbf{B} associated with underdetermined problems (equation 21) for the particular
 381 case in which $\mathbf{H} = \mathbf{W}_q = \mathbf{I}_{P'}$ (equations 9 and 13), $\mathbf{W}_d = \mathbf{I}_{D'}$ (equation 12), $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14), where
 382 $\mathbf{I}_{P'}$ and $\mathbf{I}_{D'}$ are identity matrices of order P' and D' , respectively, and $\mathbf{0}$ is a vector of zeros. Due to the
 383 presumed spatial configuration of the observed data and equivalent sources, \mathbf{a}' and \mathbf{G}' are the same for all
 384 data windows. Hence, only the data vector \mathbf{d}^m is modified according to the position of the data window.
 385 Note that equation 33 combines the potential-field transformation (equation 4) with the solution of the
 386 undetermined problem (equation 23).

387 The method proposed by Leão and Silva (1989) can be outlined by the Algorithm 2. Note that Leão and
 388 Silva (1989) directly compute the transformed potential t_c^m at the central point of each data window without
 389 explicitly computing and storing an estimated for \mathbf{p}^m (equation 32). It means that their method allows
 390 computing a single potential-field transformation. A different transformation or the same one evaluated at
 391 different points require running their moving-data window method again.

392 The total number of flops in Algorithm 2 depends on computing the $P' \times D'$ matrix \mathbf{B}' (equation 34) in
 393 line 6 and use it to define the $1 \times P'$ vector $(\mathbf{a}')^\top \mathbf{B}'$ (line 7) before starting the iterations and computing
 394 an inner product (equation 33) per iteration. We consider that the total number of flops associated with \mathbf{B}'
 395 is obtained by the matrix-matrix product $\mathbf{G}' (\mathbf{G}')^\top$, its inverse and then the premultiplication by $(\mathbf{G}')^\top$. By
 396 using table 1 and considering that inverse is computed via Cholesky factorization, we obtain that the total
 397 number of flops for lines 6 and 7 is $2(D')^2 P' + 7(D')^3/6 + 2(D')^2 P'$. Then, the total number of flops for
 398 Algorithm 2 is

$$f_{LS89} = 7/6(D')^3 + 4P'(D')^2 + M 2P'. \quad (35)$$

399 Soler and Uieda (2021) generalized the method proposed by Leão and Silva (1989) for irregularly spaced
 400 data on an undulating surface. A direct consequence of this generalization is that a different submatrix
 401 $\mathbf{G}^m \equiv \mathbf{G}[\mathbf{i}^m, \mathbf{j}^m]$ (equation 32) must be computed for each window. Differently from Leão and Silva
 402 (1989), Soler and Uieda (2021) store the computed $\tilde{\mathbf{p}}^m$ for all windows and subsequently use them to obtain
 403 a desired potential-field transformation (equation 4) as the superposed effect of all windows. The estimated
 404 $\tilde{\mathbf{p}}^m$ for all windows are combined to form a single $P \times 1$ vector $\tilde{\mathbf{p}}$, which is an estimate for original
 405 parameter vector \mathbf{p} (equation 3). For each data window, Soler and Uieda (2021) solve an overdetermined
 406 problem (equation 22) for $\tilde{\mathbf{p}}^m$ by using $\mathbf{H} = \mathbf{W}_q = \mathbf{I}_{P^m}$ (equations 9 and 13), \mathbf{W}_d^m (equation 12) equal to
 407 a diagonal matrix of weights for the data inside the m -th window and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14), so that

$$[(\mathbf{G}^m)^\top \mathbf{W}_d^m \mathbf{G}^m + \mu \mathbf{I}_{P'}] \tilde{\mathbf{p}}^m = (\mathbf{G}^m)^\top \mathbf{W}_d^m \mathbf{d}^m. \quad (36)$$

408 Unlike Leão and Silva (1989), Soler and Uieda (2021) do not adopt a sequential order of the data windows;
 409 rather, they adopt a randomized order of windows in their iterations. The overall steps of the method
 410 proposed by Soler and Uieda (2021) are defined by the Algorithm 3. For convenience, we have omitted the
 411 details about the randomized window order, normalization strategy employed and block-averaged sources
 412 layout proposed by those authors (see subsection 2.1). Note that this algorithm starts with a residuals vector
 413 \mathbf{r} that is iteratively updated. The iterative algorithm in Soler and Uieda (2021) estimates a solution ($\tilde{\mathbf{p}}^m$ in
 414 equation 36) using the data and the equivalent sources that fall within a moving-data window; however, it
 415 calculates the predicted data and the residual data in the whole survey data. Next, the residual data that fall
 416 within a new position of the data window is used as input data to estimate a new solution within the data
 417 window which, in turn, is used to calculate a new predicted data and a new residual data in the whole
 418 survey data.

419 The computational cost of Algorithm 3 can be defined in terms of the linear system (equation 36) to be
 420 solved for each window (line 10) and the subsequent updates in lines 11 and 12. We consider that the linear
 421 system cost can be quantified by the matrix-matrix and matrix-vector products $(\mathbf{G}^m)^\top \mathbf{G}^m$ and $(\mathbf{G}^m)^\top \mathbf{d}^m$,
 422 respectively, and solution of the linear system (line 10) via Cholesky factorization (equations 25 and 26).
 423 The following updates represent a saxpy without scalar-vector product (line 11) and a matrix-vector product
 424 (line 12). In this case, according to table 1, the total number of flops associated with Algorithm 3 is given

425 by:

$$f_{\text{SU21}} = M \left[\frac{1}{3}(P')^3 + 2(D' + 1)(P')^2 + (4D' + 1)P' \right], \quad (37)$$

426 where P' and D' represent, respectively, the average number of equivalent sources and data at each window.427 **7.2 Column-action update**428 We call the computational strategy *column-action update* because a single source is used to calculate the
429 predicted data and the residual data in the whole survey data. Hence, a single column of the sensitivity
430 matrix \mathbf{G} (equation 3) is calculated iteratively.431 Cordell (1992) proposed a computational strategy that was later used by Guspí and Novara (2009) and
432 relies on first defining one equivalent source located right below each observed data d_i , $i \in \{1 : D\}$, at
433 a vertical coordinate $z_i + \Delta z_i$, where Δz_i is proportional to the distance from the i -th observation point
434 (x_i, y_i, z_i) to its closest neighbor. The second step consists in updating the physical property p_j of a single
435 equivalent source, $j \in \{1 : D\}$ and remove its predicted potential field from the observed data vector \mathbf{d} ,
436 producing a residuals vector \mathbf{r} . At each iteration, the single equivalent source is the one located vertically
437 beneath the observation station of the maximum data residual. Next, the predicted data produced by this
438 single source is calculated over all of the observation points and a new data residual \mathbf{r} and the $D \times 1$
439 parameter vector \mathbf{p} containing the physical property of all equivalent sources are updated iteratively. During
440 each subsequent iteration, Cordell's method either incorporates a single equivalent source or adjusts an
441 existing equivalent source to match the maximum amplitude of the current residual field. The convergence
442 occurs when all of the residuals are bounded by an envelope of prespecified expected error. At the end, the
443 algorithm produces an estimate $\tilde{\mathbf{p}}$ for the parameter vector yielding a predicted potential field \mathbf{f} (equation
444 3) satisfactorily fitting the observed data \mathbf{d} according to a given criterion. Note that the method proposed
445 by Cordell (1992) iteratively solves the linear $\mathbf{G}\tilde{\mathbf{p}} \approx \mathbf{d}$ with a $D \times D$ matrix \mathbf{G} . At each iteration, only a
446 single column of \mathbf{G} (equation 3) is used. An advantage of this *column-action update approach* is that the
447 full matrix \mathbf{G} is never stored.448 Algorithm 4 delineates the Cordell's method. Note that a single column $\mathbf{G}[:, i_{\max}]$ of the $D \times D$ matrix \mathbf{G}
449 (equation 3) is used per iteration, where i_{\max} is the index of the maximum absolute value in \mathbf{r} . As pointed out
450 by Cordell (1992), the method does not necessarily decrease monotonically along the iterations. Besides,
451 the method may not converge depending on how the vertical distances Δz_i , $i \in \{1 : D\}$, controlling the
452 depths of the equivalent sources are set. According to Cordell (1992), the maximum absolute value r_{\max}
453 in \mathbf{r} decreases robustly at the beginning and oscillates within a narrowing envelope for the subsequent
454 iterations.455 Guspí and Novara (2009) generalized Cordell's method to perform reduction to the pole and other
456 transformations on scattered magnetic observations by using two steps. The first step involves computing
457 the vertical component of the observed field using equivalent sources while preserving the magnetization
458 direction. In the second step, the vertical observation direction is maintained, but the magnetization
459 direction is shifted to the vertical. The main idea employed by both Cordell (1992) and Guspí and Novara
460 (2009) is an iterative scheme that uses a single equivalent source positioned below a measurement station
461 to compute both the predicted data and residual data for all stations. This approach entails a computational
462 strategy where a single column of the sensitivity matrix \mathbf{G} (equation 3) is calculated per iteration.463 The total number of flops in Algorithm 4 consists in finding the maximum absolute value in vector \mathbf{r}
464 (line 6) before the while loop. Per iteration, there is a saxpy (line 11) and another search for the maximum
465 absolute value in vector \mathbf{r} (line 12). By considering that selecting the maximum absolute value in a $D \times 1$

466 vector is a $D \log_2(D)$ operation (e.g., Press et al., 2007, p. 420), the total number of flops in Algorithm 38
 467 is given by:

$$f_{C92} = D \log(D) + \text{ITMAX} [2D + D \log_2(D)] . \quad (38)$$

468 7.3 Row-action update

469 We call the computational strategy *row-action update* because a single row of the sensitivity matrix
 470 \mathbf{G} (equation 3) is calculated iteratively. Hence, the equivalent-layer solution is updated by processing a
 471 new datum (one matrix row) at each iteration. To reduce the total processing time and memory usage of
 472 equivalent-layer technique, Mendonça and Silva (1994) proposed a strategy called *equivalent data concept*.
 473 The equivalent data concept is grounded on the principle that there is a subset of redundant data that does
 474 not contribute to the final solution and thus can be dispensed. Conversely, there is a subset of observations,
 475 called equivalent data, that contributes effectively to the final solution and fits the remaining observations
 476 (redundant data). Iteratively, Mendonça and Silva (1994) selected the subset of equivalent data that is
 477 substantially smaller than the original dataset. This selection is carried out by incorporating one data point
 478 at a time.

479 Mendonça and Silva (1994) proposes an algebraic reconstruction technique (ART) (e.g., van der Sluis
 480 and van der Vorst, 1987, p. 58) to estimate a parameter vector $\tilde{\mathbf{p}}$ for a regular grid of P equivalent sources
 481 on a horizontal plane z_0 . Such methods iterate on the linear system rows to estimate corrections for the
 482 parameter vector, which may substantially save computer time and memory required to compute and store
 483 the full linear system matrix along the iterations. The convergence of such *row-update methods* depends
 484 on the linear system condition. The main advantage of such methods is not computing and storing the
 485 full linear system matrix, but iteratively using its rows. In contrast to ART-type algorithms, the rows in
 486 Mendonça and Silva (1994) are not processed sequentially. Instead, in Mendonça and Silva (1994), the
 487 rows are introduced according to their residual magnitudes (maximum absolute value in \mathbf{r}), which are
 488 computed based on the estimate over the equivalent layer from the previous iteration. The particular ART
 489 method proposed by Mendonça and Silva (1994) considers that

$$\mathbf{d} = \begin{bmatrix} \mathbf{d}_e \\ \mathbf{d}_r \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \mathbf{G}_e \\ \mathbf{R}_r \end{bmatrix}, \quad (39)$$

490 where \mathbf{d}_e and \mathbf{d}_r are $D_e \times 1$ and $D_r \times 1$ vectors and \mathbf{G}_e and \mathbf{G}_r are $D_e \times P$ and $D_r \times P$ matrices,
 491 respectively. Mendonça and Silva (1994) designate \mathbf{d}_e and \mathbf{d}_r as, respectively, *equivalent* and *redundant*
 492 data. With the exception of a normalization strategy, Mendonça and Silva (1994) calculate a $P \times 1$ estimated
 493 parameter vector $\tilde{\mathbf{p}}$ by solving an underdetermined problem (equation 23) involving only the equivalent
 494 data \mathbf{d}_e (equation 39) for the particular case in which $\mathbf{H} = \mathbf{W}_p = \mathbf{I}_P$ (equations 9 and 13), $\mathbf{W}_d = \mathbf{I}_{D_e}$
 495 (equation 12) and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14), which results in

$$\begin{aligned} (\mathbf{F} + \mu \mathbf{I}_{D_e}) \mathbf{u} &= \mathbf{d}_e \\ \tilde{\mathbf{p}} &= \mathbf{G}_e^\top \mathbf{u} , \end{aligned} \quad (40)$$

496 where \mathbf{F} is a computationally-efficient $D_e \times D_e$ matrix that approximates $\mathbf{G}_e \mathbf{G}_e^\top$. Mendonça and Silva
 497 (1994) presume that the estimated parameter vector $\tilde{\mathbf{p}}$ obtained from equation 40 leads to a $D_r \times 1$ residuals
 498 vector

$$\mathbf{r} = \mathbf{d}_r - \mathbf{G}_r \tilde{\mathbf{p}} \quad (41)$$

499 having a maximum absolute value $r_{\max} \leq \epsilon$, where ϵ is a predefined tolerance.

500 The overall method of Mendonça and Silva (1994) is defined by Algorithm 5. It is important noting
 501 that the number D_e of equivalent data in \mathbf{d}_e increases by one per iteration, which means that the order
 502 of the linear system in equation 40 also increases by one at each iteration. Those authors also propose
 503 a computational strategy based on Cholesky factorization (e.g., Golub and Van Loan, 2013, p. 163) for
 504 efficiently updating $(\mathbf{F} + \mu \mathbf{I}_{D_e})$ at a given iteration (line 16 in Algorithm 5) by computing only its new
 505 elements with respect to those computed in the previous iteration.

506 7.4 Reparameterization

507 Another approach for improving the computational performance of equivalent-layer technique consists
 508 in setting a $P \times Q$ reparameterization matrix \mathbf{H} (equation 9) with $Q \ll P$. This strategy has been used
 509 in applied geophysics for decades (e.g., Skilling and Bryan, 1984; Kennett et al., 1988; Oldenburg et al.,
 510 1993; Barbosa et al., 1997) and is known as *subspace method*. The main idea relies in reducing the linear
 511 system dimension from the original P -space to a lower-dimensional subspace (the Q -space). An estimate
 512 $\tilde{\mathbf{q}}$ for the reparameterized parameter vector \mathbf{q} is obtained in the Q -space and subsequently used to obtain
 513 an estimate $\tilde{\mathbf{p}}$ for the parameter vector \mathbf{p} (equation 3) in the P -space by using equation 9. Hence, the key
 514 aspect of this *reparameterization approach* is solving an appreciably smaller linear inverse problem for $\tilde{\mathbf{q}}$
 515 than that for the original parameter vector $\tilde{\mathbf{p}}$ (equation 3).

516 Oliveira Jr. et al. (2013) have used this approach to describe the physical property distribution on the
 517 equivalent layer in terms of piecewise bivariate polynomials. Specifically, their method consists in splitting
 518 a regular grid of equivalent sources into source windows inside which the physical-property distribution
 519 is described by bivariate polynomial functions. The key aspect of their method relies on the fact that the
 520 total number of coefficients required to define the bivariate polynomials is considerably smaller than the
 521 original number of equivalent sources. Hence, they formulate a linear inverse problem for estimating the
 522 polynomial coefficients and use them later to compute the physical property distribution on the equivalent
 523 layer.

524 The method proposed by Oliveira Jr. et al. (2013) consists in solving an overdetermined problem (equation
 525 22) for estimating the polynomial coefficients $\tilde{\mathbf{q}}$ with $\mathbf{W}_d = \mathbf{I}_D$ (equation 12) and $\bar{\mathbf{q}} = \mathbf{0}$ (equation 14), so
 526 that

$$(\mathbf{H}^\top \mathbf{G}^\top \mathbf{G} \mathbf{H} + \mu \mathbf{W}_q) \tilde{\mathbf{q}} = \mathbf{H}^\top \mathbf{G}^\top \mathbf{d}, \quad (42)$$

527 where $\mathbf{W}_q = \mathbf{H}^\top \mathbf{W}_p \mathbf{H}$ is defined by a matrix \mathbf{W}_p representing the zeroth- and first-order Tikhonov
 528 regularization (e.g., Aster et al., 2019, p. 103). Note that, in this case, the prior information is defined in the
 529 P -space for the original parameter vector \mathbf{p} and then transformed to the Q -space. Another characteristic of
 530 their method is that it is valid for processing irregularly-spaced data on an undulating surface.

531 Mendonça (2020) also proposed a reparameterization approach for the equivalent-layer technique. Their
 532 approach, however, consists in setting \mathbf{H} as a truncated singular value decomposition (SVD) (e.g., Aster
 533 et al., 2019, p. 55) of the observed potential field. Differently from Oliveira Jr. et al. (2013), however, the
 534 method of Mendonça (2020) requires a regular grid of potential-field data on horizontal plane. Another
 535 difference is that these authors uses $\mathbf{W}_q = \mathbf{I}_Q$ (equation 13), which means that the regularization is defined
 536 directly in the Q -space.

537 We consider an algorithm (not shown) that solves the overdetermined problem (equation 22) by combining
 538 the reparameterization with CGLS method (Algorithm 1). It starts with a reparameterization step defined

539 by defining a matrix $\mathbf{C} = \mathbf{G} \mathbf{H}$ (equation 10). Then, the CGLS (Algorithm 1) is applied by replacing \mathbf{G}
 540 with \mathbf{C} . In this case, the linear system is solved by the reparameterized parameter vector $\tilde{\mathbf{q}}$ instead of $\tilde{\mathbf{p}}$.
 541 At the end, the estimated $\tilde{\mathbf{q}}$ is transformed into $\tilde{\mathbf{p}}$ (equation 15). Compared to the original CGLS shown
 542 in Algorithm 1, the algorithm discussed here has the additional flops associated with the matrix-matrix
 543 product to compute \mathbf{C} and the matrix-vector product of equation 15 outside the while loop. Then, according
 544 to table 1, the total number of flops given by:

$$f_{\text{reparam.}} = 2Q(DP + D + 1) + 2PQ + \text{ITMAX} [2Q(2D + 3) + 4D] . \quad (43)$$

545 The important aspect of this approach is that, for the case in which $Q \ll P$ (equation 9), the number of
 546 flops per iteration can be substantially decreased with respect to those associated with Algorithm 1. In this
 547 case, the flops decrease per iteration compensates the additional flops required to compute \mathbf{C} and obtain $\tilde{\mathbf{p}}$
 548 from $\tilde{\mathbf{q}}$ (equation 15).

549 7.5 Sparsity induction

550 Li and Oldenburg (2010) proposed a method that applies the discrete wavelet transform to introduce
 551 sparsity into the original dense matrix \mathbf{G} (equation 3). Those authors approximate a planar grid of potential-
 552 field data by a regularly-spaced grid of equivalent sources, so that the number of data D and sources P is
 553 the same, i.e., $D = P$. Specifically, Li and Oldenburg (2010) proposed a method that applies the wavelet
 554 transform to the original dense matrix \mathbf{G} and sets to zero the small coefficients that are below a given
 555 threshold, which results in an approximating sparse representation of \mathbf{G} in the wavelet domain. They first
 556 consider the following approximation

$$\mathbf{d}_w \approx \mathbf{G}_s \mathbf{p}_w , \quad (44)$$

557 where

$$\mathbf{d}_w = \mathcal{W} \mathbf{d} , \quad \mathbf{p}_w = \mathcal{W} \mathbf{p} , \quad (45)$$

558 are the observed data and parameter vector in the wavelet domain; \mathcal{W} is a $D \times D$ orthogonal matrix
 559 defining a discrete wavelet transform; and \mathbf{G}_s is a sparse matrix obtained by setting to zero the elements of

$$\mathbf{G}_w = \mathcal{W} \mathbf{G} \mathcal{W}^\top \quad (46)$$

560 with absolute value smaller than a given threshold.

561 Li and Oldenburg (2010) solve a normalized inverse problem in the wavelet domain. Specifically, they
 562 first define a matrix

$$\mathbf{G}_L = \mathbf{G}_s \mathbf{L}^{-1} \quad (47)$$

563 and a normalized parameter vector

$$\mathbf{p}_L = \mathbf{L} \mathbf{p}_w , \quad (48)$$

564 where \mathbf{L} is a diagonal and invertible matrix representing an approximation of the first-order Tikhonov
 565 regularization in the wavelet domain. Then they solve an overdetermined problem (equation 22) to obtain
 566 an estimate $\tilde{\mathbf{p}}_L$ for \mathbf{p}_L (equation 48), with \mathbf{G}_L (equation 47), $\mathbf{H} = \mathbf{I}_P$ (equations 9), $\mu = 0$ (equation 11),
 567 $\mathbf{W}_d = \mathbf{I}_D$ (equation 12) and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14) via conjugate-gradient method (e.g., Golub and Van
 568 Loan, 2013, sec. 11.3). Finally, Li and Oldenburg (2010) compute an estimate $\tilde{\mathbf{p}}$ for the original parameter
 569 vector given by

$$\tilde{\mathbf{p}} = \mathcal{W}^\top (\mathbf{L}^{-1} \tilde{\mathbf{p}}_L) , \quad (49)$$

570 where the term within parenthesis is an estimate $\tilde{\mathbf{p}}_w$ of the parameter vector \mathbf{p}_w (equation 45) in the wavelet
 571 domain and matrix \mathcal{W}^\top represents an inverse wavelet transform.

572 Barnes and Lumley (2011) also proposed a computationally efficient method for equivalent-layer
 573 technique by inducing sparsity into the original sensitivity matrix \mathbf{G} (equation 3). Their approach consists
 574 in setting a $P \times Q$ reparameterization matrix \mathbf{H} (equation 9) with $Q \approx 1.7 P$. Note that, differently from
 575 Oliveira Jr. et al. (2013) and Mendonça (2020), Barnes and Lumley (2011) do not use the reparameterization
 576 with the purpose of reducing the number of the parameters. Instead, they use a reparameterization scheme
 577 that groups distant equivalent sources into blocks by using a bisection process. This scheme leads to
 578 a quadtree representation of the physical-property distribution on the equivalent layer, so that matrix
 579 \mathbf{GH} (equation 10) is notably sparse. Barnes and Lumley (2011) explore this sparsity in solving the
 580 overdetermined problem for $\tilde{\mathbf{q}}$ (equation 42) via conjugate-gradient method (e.g., Golub and Van Loan,
 581 2013, sec. 11.3).

582 It is difficult to predict the exact sparsity obtained from the methods proposed by Li and Oldenburg (2010)
 583 and Barnes and Lumley (2011) because it depends on several factors, including the observed potential-field
 584 data. According to Li and Oldenburg (2010), their wavelet approach results in a sparse matrix having $\approx 2\%$
 585 of the elements in \mathbf{G}_w (equation 46). The reparameterization proposed by Barnes and Lumley (2011) leads
 586 to a sparse matrix \mathbf{GH} (equation 10) with only $\approx 1\%$ of non-zero elements. These sparsity patterns can be
 587 efficiently explored, for example, in computing the required matrix-vector products along the iterations of
 588 the CGLS method (Algorithm 1).

589 7.6 Iterative methods using the full matrix \mathbf{G}

590 Xia and Sprowl (1991) introduced an iterative method for estimating the parameter vector $\tilde{\mathbf{p}}$ (equation 3),
 591 which was subsequently adapted to the Fourier domain by Xia et al. (1993). Their method uses the full
 592 and dense sensitivity matrix \mathbf{G} (equation 3) (without applying any compression or reparameterization, for
 593 example) to compute the predicted data at all observation points per iteration. More than two decades later,
 594 Siqueira et al. (2017) have proposed an iterative method similar to that presented by Xia and Sprowl (1991).
 595 The difference is that Siqueira et al.'s algorithm was deduced from the *Gauss' theorem* (e.g., Kellogg, 1967,
 596 p. 43) and the *total excess of mass* (e.g., Blakely, 1996, p. 60). Besides, Siqueira et al. (2017) have included
 597 a numerical analysis showing that their method produces very stable solutions, even for noise-corrupted
 598 potential-field data.

599 The iterative method proposed by Siqueira et al. (2017) is outlined in Algorithm 6, presumes an equivalent
 600 layer formed by monopoles (point masses) and can be applied to irregularly-spaced data on an undulating
 601 surface. Note that the residuals \mathbf{r} are used to compute a correction $\Delta\mathbf{p}$ for the parameter vector at each
 602 iteration (line 11), which requires a matrix-vector product involving the full matrix \mathbf{G} . Interestingly, this
 603 approach for estimating the physical property distribution on an equivalent layer is the same originally
 604 proposed by Bott (1960) for estimating the basement relief under sedimentary basins. The methods of Xia
 605 and Sprowl (1991) and Siqueira et al. (2017) were originally proposed for processing gravity data, but can
 606 be potentially applied to any harmonic function because they actually represent iterative solutions of the
 607 classical *Dirichlet's problem* or the *first boundary value problem of potential theory* (Kellogg, 1967, p.
 608 236) on a plane.

609 Recently, Jirigalatu and Ebbing (2019) presented another iterative method for estimating a parameter
 610 vector $\tilde{\mathbf{p}}$ (equation 3). With the purpose of combining different potential-field data, their method basically
 611 modifies that shown in Algorithm 6 by changing the initial approximation and the iterative correction for
 612 the parameter vector. Specifically, Jirigalatu and Ebbing (2019) replace line 5 by $\tilde{\mathbf{p}} = \mathbf{0}$, where $\mathbf{0}$ is a vector

613 of zeros, and line 11 by $\Delta\mathbf{p} = \omega \mathbf{G}^\top \mathbf{r}$, where ω is a positive scalar defined by trial and error. Note that
 614 this modified approach requires two matrix-vector products involving the full matrix \mathbf{G} per iteration. To
 615 overcome the high computational cost of these two products, Jirigalatu and Ebbing (2019) set an equivalent
 616 layer formed by prisms and compute their predicted potential field in the wavenumber domain by using the
 617 Gauss-FFT technique Zhao et al. (2018).

618 The iterative method proposed by Siqueira et al. (2017) (Algorithm 6) requires an entrywise product (line
 619 11), a half saxpy (line 12) and a saxpy (lines 13 and 14). Then, we get from table 1 that the total number of
 620 flops is given by:

$$f_{S0B17} = 2D + \text{ITMAX} (2D^2 + 3D) . \quad (50)$$

621 Note that the number of flops per iteration in f_{S0B17} (equation 50) has the same order of magnitude, but is
 622 smaller than that in f_{CGLS} (equation 28).

623 7.7 Iterative deconvolution

624 Recently, Takahashi et al. (2020, 2022) proposed the *convolutional equivalent-layer method*, which
 625 explores the structure of the sensitivity matrix \mathbf{G} (equation 3) for the particular case in which (i) there
 626 is a single equivalent source right below each potential-field datum and (ii) both data and sources rely
 627 on planar and regularly spaced grids. Specifically, they consider a regular grid of D potential-field data
 628 at points (x_i, y_i, z_0) , $i \in \{1 : D\}$, on a horizontal plane z_0 . The data indices i may be ordered along the
 629 x - or y -direction, which results in an x - or y -oriented grid, respectively. They also consider a single
 630 equivalent source located right below each datum, at a constant vertical coordinate $z_0 + \Delta z$, $\Delta z > 0$. In
 631 this case, the number of data and equivalent sources are equal to each other (i.e., $D = P$) and \mathbf{G} (equation
 632 3) assumes a *doubly block Toeplitz* (Jain, 1989, p. 28) or *block-Toeplitz-block* (BTTB) (Chan and
 633 Jin, 2007, p. 67) structure formed by $N_B \times N_B$ blocks, where each block has $N_b \times N_b$ elements, with
 634 $D = N_B N_b$. This particular structure allows formulating the product of \mathbf{G} and an arbitrary vector as a *fast*
 635 *discrete convolution via Fast Fourier Transform* (FFT) (Van Loan, 1992, section 4.2).

636 Consider, for example, the particular case in which $N_B = 4$, $N_b = 3$ and $D = 12$. In this case, \mathbf{G}
 637 (equation 3) is a 12×12 block matrix given by

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}^0 & \mathbf{G}^1 & \mathbf{G}^2 & \mathbf{G}^3 \\ \mathbf{G}^{-1} & \mathbf{G}^0 & \mathbf{G}^1 & \mathbf{G}^2 \\ \mathbf{G}^{-2} & \mathbf{G}^{-1} & \mathbf{G}^0 & \mathbf{G}^1 \\ \mathbf{G}^{-3} & \mathbf{G}^{-2} & \mathbf{G}^{-1} & \mathbf{G}^0 \end{bmatrix}_{D \times D}, \quad (51)$$

638 where each block \mathbf{G}^n , $n \in \{(1 - N_B) : (N_B - 1)\}$, is a 3×3 Toeplitz matrix. Takahashi et al. (2020,
 639 2022) have deduced the specific relationship between blocks \mathbf{G}^n and \mathbf{G}^{-n} and also between a given block
 640 \mathbf{G}^n and its transposed $(\mathbf{G}^n)^\top$ according to the harmonic function g_{ij} (equation 2) defining the element ij
 641 of the sensitivity matrix \mathbf{G} (equation 3) and the orientation of the data grid.

642 Consider the matrix-vector products

$$\mathbf{G} \mathbf{v} = \mathbf{w} \quad (52)$$

643 and

$$\mathbf{G}^\top \mathbf{v} = \mathbf{w} , \quad (53)$$

644 involving a $D \times D$ sensitivity matrix \mathbf{G} (equation 3) defined in terms of a given harmonic function g_{ij}
 645 (equation 2), where

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}^0 \\ \vdots \\ \mathbf{v}^{N_B-1} \end{bmatrix}_{D \times 1}, \quad \mathbf{w} = \begin{bmatrix} \mathbf{w}^0 \\ \vdots \\ \mathbf{w}^{N_B-1} \end{bmatrix}_{D \times 1}, \quad (54)$$

646 are arbitrary partitioned vectors formed by N_B sub-vectors \mathbf{v}^n and \mathbf{w}^n , $n \in \{0 : (N_B - 1)\}$, all of them
 647 having N_b elements. Equations 52 and 53 can be computed in terms of an auxiliary matrix-vector product

$$\mathbf{G}_c \mathbf{v}_c = \mathbf{w}_c, \quad (55)$$

648 where

$$\mathbf{v}_c = \begin{bmatrix} \mathbf{v}_c^0 \\ \vdots \\ \mathbf{v}_c^{N_B-1} \\ \mathbf{0} \end{bmatrix}_{4D \times 1}, \quad \mathbf{w}_c = \begin{bmatrix} \mathbf{w}_c^0 \\ \vdots \\ \mathbf{w}_c^{N_B-1} \\ \mathbf{0} \end{bmatrix}_{4D \times 1}, \quad (56)$$

649 are partitioned vectors formed by $2N_b \times 1$ sub-vectors

$$\mathbf{v}_c^n = \begin{bmatrix} \mathbf{v}_c^n \\ \mathbf{0} \end{bmatrix}_{2N_b \times 1}, \quad \mathbf{w}_c^n = \begin{bmatrix} \mathbf{w}_c^n \\ \mathbf{0} \end{bmatrix}_{2N_b \times 1}, \quad (57)$$

650 and \mathbf{G}_c is a $4D \times 4D$ *doubly block circulant* (Jain, 1989, p. 28) or *block-circulant circulant-block* (BCCB)
 651 (Chan and Jin, 2007, p. 76) matrix. What follows aims at explaining how the original matrix-vector products
 652 defined by equations 52 and 53, involving a $D \times D$ BTTB matrix \mathbf{G} exemplified by equation 51, can be
 653 efficiently computed in terms of the auxiliary matrix-vector product given by equation 55, which has a
 654 $4D \times 4D$ BCCB matrix \mathbf{G}_c .

655 Matrix \mathbf{G}_c (equation 55) is formed by $2N_B \times 2N_B$ blocks, where each block \mathbf{G}_c^n , $n \in \{(1 - N_B) : (N_B - 1)\}$ is a $2N_b \times 2N_b$ circulant matrix. For the case in which the original matrix-vector product is that
 656 defined by equation 52, the first column of blocks forming the BCCB matrix \mathbf{G}_c is given by
 657

$$\mathbf{G}_c[:, :2N_b] = \begin{bmatrix} \mathbf{G}_c^0 \\ \mathbf{G}_c^{-1} \\ \vdots \\ \mathbf{G}_c^{1-N_B} \\ \mathbf{0} \\ \mathbf{G}_c^{N_B-1} \\ \vdots \\ \mathbf{G}_c^1 \end{bmatrix}_{4D \times 2N_b}, \quad (58)$$

658 with blocks \mathbf{G}_c^n having the first column given by

$$\mathbf{G}_c^n[:, 1] = \begin{bmatrix} \mathbf{G}^n[:, 1] \\ 0 \\ (\mathbf{G}^n[1, N_b : 2])^\top \end{bmatrix}_{2N_b \times 2N_b}, \quad n \in \{(1 - N_B) : (N_B - 1)\}, \quad (59)$$

659 where \mathbf{G}^n are the blocks forming the BTTB matrix \mathbf{G} (equation 51). For the case in which the original
 660 matrix-vector product is that defined by equation 53, the first column of blocks forming the BCCB matrix
 661 \mathbf{G}_c is given by

$$\mathbf{G}_c[:, : 2N_b] = \begin{bmatrix} \mathbf{G}_c^0 \\ \mathbf{G}_c^1 \\ \vdots \\ \mathbf{G}_c^{N_B-1} \\ \mathbf{0} \\ \mathbf{G}_c^{1-N_B} \\ \vdots \\ \mathbf{G}_c^{-1} \end{bmatrix}_{4D \times 2N_b}, \quad (60)$$

662 with blocks \mathbf{G}_c^n having the first column given by

$$\mathbf{G}_c^n[:, 1] = \begin{bmatrix} (\mathbf{G}^n[1, :])^\top \\ 0 \\ \mathbf{G}^n[N_b : 2, 1] \end{bmatrix}_{2N_b \times 2N_b}, \quad n \in \{(1 - N_B) : (N_B - 1)\}. \quad (61)$$

663 The complete matrix \mathbf{G}_c (equation 55) is obtained by properly downshifting the block columns $\mathbf{G}_c[:, : 2N_b]$
 664 defined by equation 58 or 60. Similarly, the n -th block \mathbf{G}_c^n of \mathbf{G}_c is obtained by properly downshifting
 665 the first columns $\mathbf{G}_c^\ell[:, 1]$ defined by equation 59 or 61.

666 Note that \mathbf{G}_c (equation 55) is a $4D \times 4D$ matrix and \mathbf{G} (equation 51) is a $D \times D$ matrix. It seems weird
 667 to say that computing $\mathbf{G}_c \mathbf{v}_c$ is more efficient than directly computing $\mathbf{G} \mathbf{v}$. To understand this, we need first
 668 to use the fact that BCCB matrices are diagonalized by the 2D unitary discrete Fourier transform (DFT)
 669 (e.g., Davis, 1979, p. 31). Because of that, \mathbf{G}_c can be written as

$$\mathbf{G}_c = (\mathcal{F}_{2N_B} \otimes \mathcal{F}_{2N_b})^* \Lambda (\mathcal{F}_{2N_B} \otimes \mathcal{F}_{2N_b}), \quad (62)$$

670 where the symbol “ \otimes ” denotes the Kronecker product (e.g., Horn and Johnson, 1991, p. 243), \mathcal{F}_{2N_B} and
 671 \mathcal{F}_{2N_b} are the $2N_B \times 2N_B$ and $2N_b \times 2N_b$ unitary DFT matrices (e.g., Davis, 1979, p. 31), respectively,
 672 the superscript “ $*$ ” denotes the complex conjugate and Λ is a $4D \times 4D$ diagonal matrix containing the
 673 eigenvalues of \mathbf{G}_c . Due to the diagonalization of the matrix \mathbf{G}_c , equation 55 can be rewritten by using
 674 equation 62 and premultiplying both sides of the result by $(\mathcal{F}_{2N_B} \otimes \mathcal{F}_{2N_b})$, i.e.,

$$\Lambda (\mathcal{F}_{2N_B} \otimes \mathcal{F}_{2N_b}) \mathbf{v}_c = (\mathcal{F}_{2N_B} \otimes \mathcal{F}_{2N_b}) \mathbf{w}_c. \quad (63)$$

675 By following Takahashi et al. (2020), we rearrange equation 63 as follows

$$\mathcal{L} \circ (\mathcal{F}_{2N_B} \mathbf{v}_c \mathcal{F}_{2N_b}) = \mathcal{F}_{2N_B} \mathbf{W}_c \mathcal{F}_{2N_b} \quad (64)$$

676 where “ \circ ” denotes the Hadamard product (e.g., Horn and Johnson, 1991, p. 298) and \mathcal{L} , \mathbf{V}_c and \mathbf{W}_c are
 677 $2N_B \times 2N_b$ matrices obtained by rearranging, along their rows, the elements forming the diagonal of Λ
 678 (equation 62), vector \mathbf{v}_c and vector \mathbf{w}_c (equation 56), respectively. Then, by premultiplying both sides of
 679 equation 64 by $\mathcal{F}_{2N_B}^*$ and then postmultiplying both sides by $\mathcal{F}_{2N_b}^*$, we obtain

$$\mathcal{F}_{2N_B}^* [\mathcal{L} \circ (\mathcal{F}_{2N_B} \mathbf{v}_c \mathcal{F}_{2N_b})] \mathcal{F}_{2N_b}^* = \mathbf{W}_c. \quad (65)$$

680 Finally, we get from equation 62 that matrix \mathcal{L} can be computed by using only the first column $\mathbf{G}_c[:, 1]$ of
 681 the BCCB matrix \mathbf{G}_c (equation 55) according to (Takahashi et al., 2020)

$$\mathcal{L} = \sqrt{4D} \mathcal{F}_{2N_B} \mathcal{C} \mathcal{F}_{2N_b}, \quad (66)$$

682 where \mathcal{C} is a $2N_B \times 2N_b$ matrix obtained by rearranging, along its rows, the elements of $\mathbf{G}_c[:, 1]$ (equation
 683 55). It is important noting that the matrices \mathcal{C} and \mathcal{L} (equation 66) associated with the BTTB matrix \mathbf{G}
 684 (equation 51) are different from those associated with \mathbf{G}^\top .

685 The whole procedure to compute the original matrix-vector products $\mathbf{G}\mathbf{v}$ (equation 52) and $\mathbf{G}^\top\mathbf{v}$
 686 (equation 53) consists in (i) rearranging the elements of the vector \mathbf{v} and the first column $\mathbf{G}[:, 1]$ of matrix
 687 \mathbf{G} into the matrices \mathcal{V}_c and \mathcal{C} (equations 65 and 66), respectively; (ii) computing terms $\mathcal{F}_{2N_B} \mathcal{A} \mathcal{F}_{2N_b}$ and
 688 $\mathcal{F}_{2N_B}^* \mathcal{A} \mathcal{F}_{2N_b}^*$, where \mathcal{A} is a given matrix, and a Hadamard product to obtain \mathcal{W}_c (equation 65); and (iii)
 689 retrieve the elements of vector \mathbf{w} (equation 52) from \mathcal{W}_c (equation 65). It is important noting that the steps
 690 (i) and (iii) do not have any computational cost because they involve only reorganizing elements of vectors
 691 and matrices. Besides, the terms $\mathcal{F}_{2N_B} \mathcal{A} \mathcal{F}_{2N_b}$ and $\mathcal{F}_{2N_B}^* \mathcal{A} \mathcal{F}_{2N_b}^*$ in step (ii) represent, respectively, the
 692 2D Discrete Fourier Transform (2D-DFT) and the 2D Inverse Discrete Fourier Transform (2D-IDFT) of \mathcal{A} .
 693 These transforms can be efficiently computed by using the 2D Fast Fourier Transform (2D-FFT). Hence,
 694 the original matrix-vector products $\mathbf{G}\mathbf{v}$ (equation 52) and $\mathbf{G}^\top\mathbf{v}$ (equation 53) can be efficiently computed
 695 by using the 2D-FFT.

696 Algorithms 7 and 8 show pseudo-codes for the convolutional equivalent-layer method proposed by
 697 Takahashi et al. (2020, 2022). Note that those authors formulate the overdetermined problem (equation
 698 22) of obtaining an estimate $\tilde{\mathbf{p}}$ for the parameter vector \mathbf{p} (equation 3) as an *iterative deconvolution* via
 699 *conjugate gradient normal equation residual* (CGNR) Golub and Van Loan (2013, sec. 11.3) or *conjugate*
 700 *gradient least squares* (CGLS) (Aster et al., 2019, p. 165) method. They consider $\mathbf{H} = \mathbf{I}_P$ (equation 9),
 701 $\mu = 0$ (equation 11), $\mathbf{W}_d = \mathbf{W}_q = \mathbf{I}_P$ (equations 12 and 13) and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14). As shown by
 702 Takahashi et al. (2020, 2022), the CGLS produces stable estimates $\tilde{\mathbf{p}}$ for the parameter vector \mathbf{p} (equation
 703 3) in the presence of noisy potential-field data \mathbf{d} . This is a well-known property of the CGLS method (e.g.,
 704 Aster et al., 2019, p. 166).

705 The key aspect of Algorithm 7 is replacing the matrix-vector products of CGLS (Algorithm 1) by fast
 706 convolutions (Algorithm 8). A fast convolution requires one 2D-DFT, one 2D-IDFT and an entrywise
 707 product of matrices. We consider that the 2D-DFT/IDFT are computed with 2D-FFT and requires
 708 $\lambda(4D) \log_2(4D)$ flops, where $\lambda = 5$ is compatible with a radix-2 FFT (Van Loan, 1992, p. 16), and
 709 the entrywise product $24D$ flops because it involves two complex matrices having $4D$ elements (Golub
 710 and Van Loan, 2013, p. 36). Hence, Algorithm 8 requires $\lambda(16D) \log_2(4D) + 26D$ flops, whereas a
 711 conventional matrix-vector multiplication involving a $D \times D$ matrix requires $2D^2$ (table 1). Finally,
 712 Algorithm 7 requires two 2D-FFTs (lines 4 and 5), one fast convolution and an inner product (line 8)
 713 previously to the while loop. Per iteration, there are three saxpys (lines 12, 15 and 16), two inner products
 714 (lines 14 and 17) and two fast convolutions (lines 13 and 17), so that:

$$f_{\text{T0B20}} = \lambda(16D) \log_2(4D) + 26D + \text{ITMAX} [\lambda(16D) \log_2(4D) + 58D]. \quad (67)$$

715 7.8 Direct deconvolution

716 The method proposed by Takahashi et al. (2020, 2022) can be reformulated to avoid the iterations of the
 717 conjugate gradient method. This alternative formulation consists in considering that $\mathbf{v} = \mathbf{p}$ and $\mathbf{w} = \mathbf{d}$ in

718 equation 52, where \mathbf{p} is the parameter vector (equation 3) and \mathbf{d} the observed data vector. In this case, the
 719 equality “=” in equation 52 becomes an approximation “ \approx ”. Then, equation 64 is manipulated to obtain

$$\mathcal{V}_c \approx \mathcal{F}_{2N_B}^* \left[(\mathcal{F}_{2N_B} \mathcal{W}_c \mathcal{F}_{2N_b}) \circ \check{\mathcal{L}} \right] \mathcal{F}_{2N_b}^*, \quad (68)$$

720 where

$$\check{\mathcal{L}} = \mathcal{L}^* \oslash (\mathcal{L} \circ \mathcal{L}^* + \zeta \mathbf{1}), \quad (69)$$

721 $\mathbf{1}$ is a $4D \times 4D$ matrix of ones, “ \oslash ” denotes entrywise division and ζ is a positive scalar. Note that $\zeta = 0$
 722 leads to $\mathbf{1} \oslash \mathcal{L}$. In this case, the entrywise division may be problematic due to the elements of \mathcal{L} having
 723 absolute value equal or close to zero. So, a small ζ is set to avoid this problem in equation 69. Next, we use
 724 $\check{\mathcal{L}}$ to obtain a matrix \mathcal{V}_c from equation 68. Finally, the elements of the estimated parameter vector $\tilde{\mathbf{p}}$ are
 725 retrieved from the first quadrant of \mathcal{V}_c . This procedure represents a *direct deconvolution* (e.g., Aster et al.,
 726 2019, p. 220) using a *Wiener filter* (e.g., Gonzalez and Woods, 2002, p. 263).

727 The required total number of flops associated with the direct deconvolution aggregates one 2D-FFT
 728 to compute matrix \mathcal{L} (equation 66), one entrywise product $\mathcal{L} \circ \mathcal{L}^*$ involving complex matrices and one
 729 entrywise division to compute $\check{\mathcal{L}}$ (equation 69) and a fast convolution (Algorithm 8) to evaluate equation
 730 68, which results in:

$$f_{\text{deconv.}} = \lambda (12D) \log_2(4D) + 72D. \quad (70)$$

731 Differently from the convolutional equivalent-layer method proposed by Takahashi et al. (2020, 2022), the
 732 alternative direct deconvolution presented here produces an estimated parameter vector $\tilde{\mathbf{p}}$ directly from
 733 the observed data \mathbf{d} , in a single step, avoiding the conjugate gradient iterations. On the other hand, the
 734 alternative method presented here requires estimating a set of tentative parameter vectors $\tilde{\mathbf{p}}$ for different
 735 predefined ζ . Besides, there must be criterion to chose the best $\tilde{\mathbf{p}}$ from this tentative set. This can be
 736 made, for example, by using the well-known *L-curve* (Hansen, 1992). From a computational point of view,
 737 the number of CGLS iterations in the method proposed by Takahashi et al. (2020, 2022) is equivalent to
 738 the number of tentative estimated parameter vectors required to form the L-curve in the proposed direct
 739 deconvolution.

8 NUMERICAL SIMULATIONS

740 8.1 Flops count

741 Figure 1 shows the total number of flops for solving the overdetermined problem (equation 22) with
 742 different equivalent-layer methods (equations 27, 28, 35, 37, 38, 43, 50, 67, and 70), by considering
 743 the particular case in which $\mathbf{H} = \mathbf{I}_P$ (equation 9 and subsection 3.2), $\mu = 0$ (equation 11), $\mathbf{W}_d = \mathbf{I}_D$
 744 (equation 12) and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14), where \mathbf{I}_P and \mathbf{I}_D are the identities of order P and D , respectively.
 745 The flops are computed for different number of potential-field data ranging from 10,000 to 1,000,000.
 746 Figure 1 shows that the moving data-window strategy by using Leão and Silva's 1989 method and direct
 747 deconvolution are the fastest methods.

748 The control parameters to run the equivalent-layer methods shown in Figure 1 are the following: i) in
 749 CGLS, reparameterization approaches (e.g., Oliveira Jr. et al., 2013; Mendonça, 2020), Siqueira et al.
 750 (2017), and Takahashi et al. (2020) (equations 28 38), 43, 50 and 67) we set ITMAX = 50; ii) Cordell
 751 (1992) we set ITMAX = 6 D ; iii) in Leão and Silva (1989) (equation 35) we set $D' = 49$ (7×7) and
 752 $P' = 225$ (15×15); and iv) in Soler and Uieda (2021) (equation 37) we set $D' = P' = 900$ (30×30).

753 8.2 Synthetic potential-field data

754 We create a model composed of several rectangular prisms that can be split into three groups. The
 755 first is composed of 300 small cubes with top at 0 m and side lengths defined according to a pseudo-
 756 random variable having uniform distribution from 100 to 200 m. Their density contrasts are defined by
 757 a pseudo-random variable uniformly distributed from 1000 to 2000 kg/m³. These prisms produce the
 758 short-wavelength component of the simulated gravity data. The 4 prisms forming the second group of our
 759 model have tops varying from 10 to 100 m and bottom from 1010 to 1500 m. They have density contrasts
 760 of 1500, -1800, -3000 and 1200 kg/m³ and side lengths varying from 1000 to 4000 m. These prisms
 761 produce the mid-wavelength component of the simulated gravity data. There is also a single prisms with
 762 top at 1000 m, bottom at 1500 m and side lengths of 4000 and 6000 m. This prisms has density contrast is
 763 -900 kg/m³ and produces the long-wavelength of our synthetic gravity data.

764 We have computed noise-free gravity disturbance and gravity-gradient tensor components produced by
 765 our model on a regularly spaced grid of 50×50 points at $z = -100$ m (Figure 2). We have also simulated
 766 additional $L = 20$ gravity disturbance data sets \mathbf{d}^ℓ , $\ell \in \{1 : L\}$, by adding pseudo-random Gaussian noise
 767 with zero mean and crescent standard deviations to the noise-free data (not shown). The standard deviations
 768 vary from 0.5% to 10% of the maximum absolute value in the noise-free data, which corresponds to 0.21
 769 and 4.16 mGal, respectively.

770 8.3 Stability analysis and gravity-gradient components

771 We set a planar equivalent layer of point masses having one source below each datum at a constant
 772 vertical coordinate $z \approx 512.24$ m. This depth was set by following the Dampney's (1969) criterion (see
 773 Subsection 2.1), so that the vertical distance Δz between equivalent sources and the simulated data is equal
 774 to $3 \times$ the grid spacing ($\Delta x = \Delta y \approx 204.08$ m). Note that, in this case, the layer has a number of sources
 775 P equal to the number of data D .

776 We have applied the Cholesky factorization (equations 25 and 26), CGLS (Algorithm 1), the iterative
 777 method of Siqueira et al. (2017) (Algorithm 6), the iterative deconvolution (Algorithms 7 and 8) proposed
 778 by Takahashi et al. (2020) (Algorithm 7) and the direct deconvolution (equations 68 and 69) with four
 779 different values for the parameter ζ to the 21 gravity data sets.

780 For each method, we have obtained one estimate $\tilde{\mathbf{p}}$ from the noise-free gravity data \mathbf{d} and $L = 20$
 781 estimates $\tilde{\mathbf{p}}^\ell$ from the noise-corrupted gravity data \mathbf{d}^ℓ , $\ell \in \{1 : L\}$, for the planar equivalent layer of point
 782 masses, totaling 21 estimated parameter vectors and 20 pairs $(\Delta p^\ell, \Delta d^\ell)$ of model and data perturbations
 783 (equations 30 and 31). Figure 3 shows the numerical stability curves computed with each method for the
 784 synthetic gravity data.

785 All these 21 estimated parameters vectors were obtained by solving the overdetermined problem (equation
 786 22) with the same method for the particular case in which $\mathbf{H} = \mathbf{I}$ (equation 9 and subsection 3.2),
 787 $\mathbf{W}_d = \mathbf{W}_q = \mathbf{I}$ (equations 12 and 13) and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14), where \mathbf{I} is the identity of order D .

788 Figure 3 shows how the numerical stability curves vary as the level of the noise is increased. We can see
 789 that for all methods, a linear tendency is observed as it is expected. The inclination of the straight line
 790 indicates the stability of each method. As shown in Figure 3, the direct deconvolution with $\zeta = 0$ exhibits a
 791 high slope, which indicates high instability and emphasizes the necessity of using the Wiener filter ($\zeta > 0$
 792 in equation 69).

793 The estimated stability parameters κ (equation 29) obtained for the Cholesky factorization, CGLS and
 794 iterative deconvolution are close to each other (Figures 3). They are slightly smaller than that obtained for
 795 the iterative method of Siqueira et al. (2017) (S0B17). Note that by varying the parameter ζ (equation 69) it
 796 is possible to obtain different stability parameters κ for the direct deconvolution. There is no apparent rule
 797 to set ζ . A practical criterion can be the maximum ζ producing a satisfactory data fit. Overshoot values
 798 tend to exaggeratedly smooth the predicted data.

799 We inverted the noise-corrupted gravity disturbance with the highest noise level (not shown) to estimate
 800 an equivalent layer (not shown) via iterative deconvolution (Algorithm 7). Figure 4(G) shows the residuals
 801 (in mGal) between the predicted and noise-corrupted gravity disturbances. As we can see, the residuals
 802 are uniformly distributed on simulated area and suggest that the equivalent layer produces a good data fit.
 803 This can be verified by inspecting the histogram of the residuals between the predicted and noise-corrupted
 804 gravity disturbances shown in panel (G) of Figure 5.

805 Using the estimated layer, we have computed the gravity-gradient data (not shown) at the observations
 806 points. Figures 4 (A)–(F) show the residuals (in Eötvös) between the predicted (not shown) and noise-free
 807 gravity-gradient data (Figure 2). These figures show that the iterative deconvolution (Algorithm 7) could
 808 predict the six components of the gravity-gradient tensor with a good precision, which can also be verified
 809 in the corresponding histograms shown in Figure 5.

810 In the supplementary material, we show the residuals between the gravity data predicted by the equivalent
 811 layer estimated by using the following methods: i) the CGLS method (Algorithm 1); ii) the Cholesky
 812 factorization (equations 25 and 26); iii) the iterative method proposed by Siqueira et al. (2017) (Algorithm
 813 6); iv) the direct deconvolution with optimal value of $\zeta = 10^{-22}$ (equation 69); and v) the iterative method
 814 proposed by Cordell (1992) (Algorithm 4).

9 APPLICATIONS TO FIELD DATA

815 In this section, we show the applications of the iterative (Algorithm 7) and direct (equations 68 and 69)
 816 deconvolutions to field data sets over the Carajás Mineral Province (CMP) in the Amazon craton (Moroni
 817 et al., 2001; Villas and Santos, 2001). This area is known for its intensive mineral exploration such as iron,
 818 copper, gold, manganese, and, recently, bauxite.

819 9.1 Geological setting

820 The Amazon Craton is one of the largest and least-known Archean-Proterozoic areas in the world,
 821 comprehending a region with a thousand square kilometers. It is one of the main tectonic units in South
 822 America, which is covered by five Phanerozoic basins: Maranhão (Northeast), Amazon (Central), Xingu-
 823 Alto Tapajós (South), Parecis (Southwest), and Solimões (West). The Craton is limited by the Andean
 824 Orogenic Belt to the west and the by Araguaia Fold Belt to the east and southeast. The Amazon craton has
 825 been subdivided into provinces according to two models, one geochronological and the other geophysical-
 826 structural (Amaral, 1974; Teixeira et al., 1989; Tassinari and Macambira, 1999). Thus, seven geological
 827 provinces with distinctive ages, evolution, and structural patterns can be observed, namely: (i) Carajás with
 828 two domains - the Mesoarchean Rio Maria and Neoarchean Carajás; (ii) Archean-Paleoproterozoic Central
 829 Amazon, with Iriri-Xingu and Curuá-Mapuera domains; (ii) Trans-Amazonian (Ryacian), with the Amapá
 830 and Bacajá domains; (iv) the Orosinian Tapajós-Parima, with Peixoto de Azevedo, Tapajós, Uaimiri, and
 831 Parima domains; (v) Rondônia-Juruena (Statherian), with Jamari, Juruena, and Jauru domains; (vi) The
 832 Statherian Rio Negro, with Rio Negro and Imeri domains; and (vii) Sunsás (Meso-Neoproterozoic), with
 833 Santa Helena and Nova Brasilândia domains (Santos et al., 2000). Nevertheless, we focus this work only
 834 on the Carajás Province.

835 The Carajás Mineral Province (CMP) is located in the east-southeast region of the craton, within an old
 836 tectonically stable nucleus in the South American Plate that became tectonically stable at the beginning of
 837 Neoproterozoic (Salomao et al., 2019). This area has been the target of intensive exploration at least since
 838 the final of the '60s, after the discovery of large iron ore deposits. There are several greenstone belts in the
 839 region, among them are the Andorinhas, Inajá, Cumaru, Carajás, Serra Leste, Serra Pelada, and Sapucaia
 840 (Santos et al., 2000). The mineralogic and petrologic studies in granite stocks show a variety of minerals
 841 found in the province, such as amphibole, plagioclase, biotite, ilmenite, and magnetite (Cunha et al., 2016).

842 9.2 Potential-field data

843 The field data used here were obtained from an airborne survey conducted by Lasa Prospecções S/A.
 844 and Microsurvey Aerogeofísica Consultoria Científica Ltda. between April/2013 and October/2014. The
 845 survey area covers $\approx 58000 \text{ km}^2$ between latitudes $-8^\circ / -5^\circ$ and longitudes $-53^\circ / -49.5^\circ$ referred to the
 846 WGS-84 datum. We obtained the horizontal coordinates x and y already in the UTM zone 22S. The flight
 847 and tie lines are spaced at 3 km and 12 km, with orientation along directions $N - S$ and $E - W$, respectively.
 848 The data are placed at an approximately constant distance of 900 m above the ground. Figure 6 shows the
 849 $D = 500,000$ aerogravimetric data on a grid of 1000×500 observation points with $\Delta x = 358.12 \text{ m}$ and
 850 $\Delta y = 787.62 \text{ m}$.

851 9.3 Potential-field transformation

852 We applied the equivalent-layer technique to the observed data (Figure 6) with the purpose of illustrating
 853 how to estimate the gravity-gradient tensor over the study area. We used an equivalent layer layout with
 854 one source located below each datum (so that $P = D$) on a horizontal plane having a vertical distance

855 $\Delta z \approx 2362.86$ m from the observation plane. This setup is defined by setting $\Delta z \approx 3 dy$, which follows the
856 same strategy of Reis et al. (2020). We solve the linear inverse problem for estimating the physical-property
857 distribution on the layer by using the iterative deconvolution (Algorithm 7) with a maximum number of 50
858 iterations. Actually, the algorithm have converged with only 18 iterations.

859 Figure 7(G) shows the histogram of the residuals between the predicted (not shown) and observed data
860 (Figure 6). As we can see, the iterative deconvolution produced an excellent data fit. By using the estimated
861 layer, we have computed the gravity-gradient tensor components at the observation points. The results are
862 shown in Figures 7(A)–(F).

863 Considering the processing time, the iterative deconvolution took ≈ 1.98 s to execute the 18
864 iterations for estimating the physical-property distribution on the layer by inverting the $D = 500,000$
865 observed data. The code was run in a modest computer with 16,0 GiB of memory and processor
866 12th Gen Intel Core i9 – 12900H $\times 20$. Given the estimated equivalent layer, the gravity-gradient
867 components shown in Figure 7 were computed in ≈ 0.52 s. These results demonstrate the efficiency
868 of the iterative deconvolution method in processing large datasets.

10 DISCUSSION AND CONCLUSION

869 We have presented a review of the strategies used to overcome the intensive computational cost of the
870 equivalent-layer technique for processing potential-field data. Each of these strategies is rarely used
871 individually; rather, some developed equivalent-layer methods combine more than one strategy to make
872 them computationally efficient in handling large-scale data sets. This comprehensive review addresses the
873 following specific strategies for reducing the computational cost of equivalent-layer technique.

874 The first one is the moving data-window scheme spanning the data set. This strategy solves several much
875 smaller, regularized linear inverse problems instead of a single large one. Each linear inversion is solved
876 using the potential-field observations and equivalent sources within a given moving window and can be
877 applied to both regularly or irregularly spaced data sets. If the data and the sources are distributed on planar
878 and regularly spaced grids, this strategy offers a significant advantage because the sensitivity submatrix of
879 a given moving window remains the same for all windows. Otherwise, the computational efficiency of the
880 equivalent-layer technique using the moving-window strategy decreases because the sensitivity submatrix
881 for each window must be computed.

882 The second and third strategies, referred to as the column-action and row-action updates, involve
883 iteratively calculating a single column and a single row of the sensitivity matrix, respectively. By following
884 the column-action update strategy, a single column of the sensitivity matrix is calculated during each
885 iteration. This implies that a single equivalent source contributes to the fitting of data in each iteration.
886 Conversely, in the row-action update strategy, a single row of the sensitivity matrix is calculated per
887 iteration, which means that one potential-field observation is incorporated in each iteration, forming a new
888 subset of equivalent data much smaller than the original data. Both strategies (column- and row-action
889 updates) have a great advantage because a single column or a single row of the sensitivity matrix is
890 calculated iteratively. However, to our knowledge, the strategy of the column-action update presents some
891 issues related to convergence, and the strategy of the row-action update can also have issues if the number
892 of equivalent data is not significantly smaller than the original number of data points.

893 The fourth strategy is the sparsity induction of the sensitivity matrix using wavelet compression, which
894 involves transforming a full sensitivity matrix into a sparse one with only a few nonzero elements. The
895 developed equivalent-layer methods using this strategy achieve sparsity by setting matrix elements to
896 zero if their values are smaller than a predefined threshold. We highlight two methods that employ the
897 sparsity induction strategy. The first method, known as wavelet-compression equivalent layer, compresses
898 the coefficients of the original sensitivity matrix using discrete wavelet transform, achieves sparsity in the
899 sensitivity matrix, and solves the inverse problem in the wavelet domain without an explicit regularization
900 parameter. The regularized solution in the wavelet domain is estimated using a conjugate gradient (CG)
901 least squares algorithm, where the number of iterations serves as a regularization factor. The second
902 equivalent-layer method that uses the sparsity induction strategy applies quadtree discretization of the
903 parameters over the equivalent layer, achieves sparsity in the sensitivity matrix, and solves the inverse
904 problem using CG algorithm. In quadtree discretization, equivalent sources located far from the observation
905 point are grouped together to form larger equivalent sources, reducing the number of parameters to be
906 estimated. Computationally, the significant advantage of the equivalent-layer methods employing wavelet
907 compression and quadtree discretization is the sparsity induction in the sensitivity matrix, which allows
908 for fast iteration of the CG algorithm. However, we acknowledge that this strategy requires computing
909 the full and dense sensitivity matrix, which can be considered a drawback when processing large-scale
910 potential-field data.

911 The fifth strategy is the reparametrization of the original parameters to be estimated in the equivalent-layer
912 technique. In this strategy, the developed equivalent-layer methods reduce the dimension of the linear
913 system of equations to be solved by estimating a lower-dimensional parameter vector. We highlight two
914 methods that used the reparametrization strategy: i) the polynomial equivalent layer (PEL) and; ii) the
915 lower-dimensional subspace of the equivalent layer. In the PEL, there is an explicit reparametrization
916 of the equivalent layer by representing the unknown distribution over the equivalent layer as a set of
917 piecewise-polynomial functions defined on a set of equivalent-source windows. The PEL method estimates
918 the polynomial coefficients of all equivalent-source windows. Hence, PEL reduces the dimension of the
919 linear system of equations to be solved because the polynomial coefficients within all equivalent-source
920 windows are much smaller than both the number of equivalent sources and the number of data points.
921 In the lower-dimensional subspace of the equivalent layer, there is an implicit reparametrization of the
922 equivalent layer by reducing the linear system dimension from the original and large-model space to a
923 lower-dimensional subspace. The lower-dimensional subspace is grounded on eigenvectors of the matrix
924 composed by the gridded data set. The main advantage of the reparametrization of the equivalent layer is to
925 deal with lower-dimensional linear system of equations. However, we acknowledge that this strategy may
926 impose an undesirable smoothing effect on both the estimated parameters over the equivalent layer and the
927 predicted data.

928 The sixth strategy involves an iterative scheme in which the estimated distribution over the equivalent
929 layer is updated iteratively. Following this strategy, the developed equivalent-layer methods differ either in
930 terms of the expression used for the estimated parameter correction or the domain utilized (wavenumber or
931 space domains). The iterative estimated correction may have a physical meaning, such as the excess mass
932 constraint. All the iterative methods are efficient as they can handle irregularly spaced data on an undulating
933 surface, and the updated corrections for the parameter vector at each iteration are straightforward, involving
934 the addition of a quantity proportional to the data residual. However, they have a disadvantage because the
935 iterative strategy requires computing the full and dense sensitivity matrix to compute the predicted and
936 residual data in all observation stations per iteration.

937 The seventh strategy is called iterative deconvolutional of the equivalent layer. This strategy deals with
938 regularly spaced grids of data stations and equivalent sources which are located at a constant height and
939 depth, respectively. Specifically, one source is placed directly below each observation station, which results
940 in sensitivity matrices with a BTTB (Block-Toeplitz Toeplitz-Block) structure. It is possible to embed the
941 BTTB matrix into a matrix of Block-Circulant Circulant-Block (BCCB) structure, which requires only
942 one equivalent source. This allows for fast matrix-vector product using a 2D fast Fourier transform (2D
943 FFT). As a result, the potential-field forward modeling can be calculated using a 2D FFT with only one
944 equivalent source required. The main advantages of this strategy are that the entire sensitivity matrices
945 do not need to be formed or stored; only their first columns are required. Additionally, it allows for a
946 highly efficient iteration of the CG algorithm. However, the iterative deconvolutional of the equivalent
947 layer requires observations and equivalent sources aligned on a horizontal and regularly-spaced grid.

948 The eighth strategy is a direct deconvolution method, which is a mathematical process very common in
949 geophysics. However, to our knowledge, direct deconvolution has never been used to solve the inverse
950 problem associated with the equivalent-layer technique. From the mathematical expressions in the iterative
951 deconvolutional equivalent layer with BTTB matrices, direct deconvolution arises naturally since it is an
952 operation inverse to convolution. The main advantage of applying the direct deconvolution strategy in
953 the equivalent layer is that it avoids, for example, the iterations of the CG algorithm. However, the direct

954 deconvolution is known to be an unstable operation. To mitigate this instability, the Wiener deconvolution
955 method can be adopted.

956 We show in this work that the computational cost of the equivalent layer can vary from up to 10^9 flops
957 depending on the method without compromising the linear system stability. The moving data-window
958 scheme and direct deconvolution are the fastest methods; however, they both have drawbacks. To be
959 computationally efficient, the moving data-window scheme and the direct deconvolution require data and
960 equivalent sources that are distributed on planar and regularly spaced grids. Moreover, they both requires
961 choosing an optimum parameter of stabilization. We stress that the direct deconvolution has an additional
962 disadvantage in terms of a higher data residual and border effects over the equivalent layer after processing.
963 These effects can be seen from the upward continuation of the real data from Carajás.

964 We draw the readers' attention to the possibility of combining more than one aforementioned strategies
965 for reducing the computational cost of the equivalent-layer technique.

CONFLICT OF INTEREST STATEMENT

966 The authors declare that the research was conducted in the absence of any commercial or financial
967 relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

968 VCOJr and VCFB: Study conception and mathematical deductions. VCOJr: Algorithms. VCOJr and DT:
969 Python codes and synthetic data applications. VCOJr and ALAR: Real data applications. VCOJr and
970 VCFB: Result analysis and draft manuscript preparation.

FUNDING

971 VCFB was supported by fellowships from CNPq (grant 309624/2021-5) and FAPERJ (grant
972 26/202.582/2019). VCOJr was supported by fellowships from CNPq (grant 315768/2020-7) and FAPERJ
973 (grant E-26/202.729/2018). DT was supported by a Post-doctoral scholarship from CNPq (grant
974 300809/2022-0).

ACKNOWLEDGMENTS

975 We thank the brazilian federal agencies CAPES, CNPq, state agency FAPERJ and Observatório Nacional
976 research institute and Universidade do Estado do Rio de Janeiro.

DATA AVAILABILITY STATEMENT

977 The datasets generated for this study can be found in the frontiers-paper Github repository link:
978 <https://github.com/XXXXXX/eqlayer-review-computational>.

REFERENCES

- 979 Amaral, G. (1974). *Geologia Pré-Cambriana da Região Amazônica*. Ph.D. thesis, Universidade de São
980 Paulo
- 981 Aster, R. C., Borchers, B., and Thurber, C. H. (2019). *Parameter Estimation and Inverse Problems*
982 (Elsevier), 3 edn.
- 983 Barbosa, V. C. F., Silva, J. B., and Medeiros, W. E. (1997). Gravity inversion of basement relief using
984 approximate equality constraints on depths. *Geophysics* 62, 1745–1757
- 985 Barnes, G. and Lumley, J. (2011). Processing gravity gradient data. *GEOPHYSICS* 76, I33–I47. doi:10.
986 1190/1.3548548
- 987 Blakely, R. J. (1996). *Potential Theory in Gravity and Magnetic Applications* (Cambridge University
988 press)
- 989 Bott, M. H. P. (1960). The use of Rapid Digital Computing Methods for Direct Gravity Interpretation
990 of Sedimentary Basins. *Geophysical Journal International* 3, 63–67. doi:10.1111/j.1365-246X.1960.
991 tb00065.x
- 992 Chan, R. H.-F. and Jin, X.-Q. (2007). *An introduction to iterative Toeplitz solvers*, vol. 5 (SIAM)
- 993 Cordell, L. (1992). A scattered equivalent-source method for interpolation and gridding of potential-field
994 data in three dimensions. *Geophysics* 57, 629–636

- 995 Cunha, I. R., Dall'Agnol, R., and Feio, G. R. L. (2016). Mineral chemistry and magnetic petrology of the
996 archean Planalto Suite, Carajás Province – Amazonian Craton: Implications for the evolution of ferroan
997 archean granites. *Journal of South American Earth Sciences* 67, 100–121
- 998 Dampney, C. N. G. (1969). The equivalent source technique. *GEOPHYSICS* 34, 39–53. doi:10.1190/1.
999 1439996
- 1000 Davis, P. J. (1979). *Circulant matrices* (John Wiley & Sons, Inc.)
- 1001 Elfving, T., Hansen, P. C., and Nikazad, T. (2017). Convergence analysis for column-action methods in
1002 image reconstruction. *Numerical Algorithms* 74, 905–924. doi:10.1007/s11075-016-0176-x
- 1003 Emilia, D. A. (1973). Equivalent sources used as an analytic base for processing total magnetic field
1004 profiles. *GEOPHYSICS* 38, 339–348. doi:10.1190/1.1440344
- 1005 Golub, G. H. and Van Loan, C. F. (2013). *Matrix Computations*. Johns Hopkins Studies in the Mathematical
1006 Sciences (Johns Hopkins University Press), 4 edn.
- 1007 Gonzalez, R. C. and Woods, R. E. (2002). *Digital Image Processing* (Prentice Hall), 2 edn.
- 1008 Gonzalez, S. P., Barbosa, V. C. F., and Oliveira Jr., V. C. (2022). Analyzing the ambiguity of the remanent-
1009 magnetization direction separated into induced and remanent magnetic sources. *Journal of Geophysical
1010 Research: Solid Earth* 127, 1–24. doi:10.1029/2022JB024151
- 1011 Guspí, F., Introcaso, A., and Introcaso, B. (2004). Gravity-enhanced representation of measured geoid
1012 undulations using equivalent sources. *Geophysical Journal International* 159, 1–8. doi:10.1111/j.
1013 1365-246X.2004.02364.x
- 1014 Guspí, F. and Novara, I. (2009). Reduction to the pole and transformations of scattered magnetic data using
1015 Newtonian equivalent sources. *GEOPHYSICS* 74, L67–L73. doi:10.1190/1.3170690
- 1016 Hansen, P. C. (1992). Analysis of discrete ill-posed problems by means of the l-curve. *SIAM Review* 34,
1017 561–580. doi:10.1137/1034115
- 1018 Hansen, R. O. and Miyazaki, Y. (1984). Continuation of potential fields between arbitrary surfaces.
1019 *GEOPHYSICS* 49, 787–795. doi:10.1190/1.1441707
- 1020 Horn, R. A. and Johnson, C. R. (1991). *Topics in Matrix Analysis* (Cambridge University Press), 1 edn.
- 1021 Jain, A. K. (1989). *Fundamentals of Digital Image Processing* (Pearson), 1 edn.
- 1022 Jirigalatu, J. and Ebbing (2019). A fast equivalent source method for airborne gravity gradient data.
1023 *Geophysics* 84, G75–G82. doi:10.1190/GEO2018-0366.1
- 1024 Kellogg, O. D. (1967). *Foundations of Potential Theory* (Springer-Verlag), reprint from the first edition of
1025 1929 edn.
- 1026 Kennett, B., Sambridge, M., and Williamson, P. (1988). Subspace methods for large inverse problems with
1027 multiple parameter classes. *Geophysical Journal International* 94, 237–247
- 1028 Leão, J. W. D. and Silva, J. B. C. (1989). Discrete linear transformations of potential field data. *Geophysics*
1029 54, 497–507. doi:10.1190/1.1442676
- 1030 Li, Y., Nabighian, M., and Oldenburg, D. W. (2014). Using an equivalent source with positivity for low-
1031 latitude reduction to the pole without striation. *GEOPHYSICS* 79, J81–J90. doi:10.1190/geo2014-0134.
1032 1
- 1033 Li, Y. and Oldenburg, D. W. (2010). Rapid construction of equivalent sources using wavelets.
1034 *GEOPHYSICS* 75, L51–L59. doi:10.1190/1.3378764
- 1035 Mendonça, C. A. (2020). Subspace method for solving large-scale equivalent layer and density mapping
1036 problems. *GEOPHYSICS* 85, G57–G68. doi:10.1190/geo2019-0302.1
- 1037 Mendonça, C. A. and Silva, J. B. C. (1994). The equivalent data concept applied to the interpolation of
1038 potential field data. *Geophysics* 59, 722–732. doi:10.1190/1.1443630
- 1039 Menke, W. (2018). *Geophysical data analysis: Discrete inverse theory* (Elsevier), 4 edn.

- 1040 Moroni, M., Girardi, V., and Ferrario, A. (2001). The Serra Pelada Au-PGE deposit, Serra dos Carajás (Pará
1041 state, Brazil): geological and geochemical indications for a composite mineralising process. *Mineralium*
1042 *Deposita* 36, 768–785
- 1043 Oldenburg, D., McGillivray, P., and Ellis, R. (1993). Generalized subspace methods for large-scale inverse
1044 problems. *Geophysical Journal International* 114, 12–20
- 1045 Oliveira Jr., V. C., Barbosa, V. C. F., and Uieda, L. (2013). Polynomial equivalent layer. *GEOPHYSICS* 78,
1046 G1–G13. doi:10.1190/geo2012-0196.1
- 1047 Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical recipes: the art of*
1048 *scientific computing* (Cambridge University Press), 3 edn.
- 1049 Reis, A. L. A., Oliveira Jr., V. C., and Barbosa, V. C. F. (2020). Generalized positivity constraint on
1050 magnetic equivalent layers. *Geophysics* 85, 1–45. doi:10.1190/geo2019-0706.1
- 1051 Roy, A. (1962). Ambiguity in geophysical interpretation. *GEOPHYSICS* 27, 90–99. doi:10.1190/1.
1052 1438985
- 1053 Salomao, G. N., Dall’Agnol, R., Angelica, R. S., Figueiredo, M. A., Sahoo, P. K., Medeiros-Filho, C. A.,
1054 et al. (2019). Geochemical mapping and estimation of background concentrations in soils of Carajás
1055 mineral province, eastern Amazonian Craton, Brazil. *Geochemistry: Exploration, Environment, Analysis*
1056 19, 431–447
- 1057 Santos, J. O. S., Hartmann, L. A., Gaudette, H. E., Groves, D. I., Mcnaughton, M. J., and Fletcher, I. R.
1058 (2000). A new understanding of the provinces of the Amazon Craton based on integration of field
1059 mapping and U-Pb and Sm-Nd geochronology. *Gondwana Research* 3, 453–488
- 1060 Silva, J. B. C. (1986). Reduction to the pole as an inverse problem and its application to low-latitude
1061 anomalies. *GEOPHYSICS* 51, 369–382. doi:10.1190/1.1442096
- 1062 Siqueira, F., Oliveira Jr., V. C., and Barbosa, V. C. F. (2017). Fast iterative equivalent-layer technique for
1063 gravity data processing: A method grounded on excess mass constraint. *GEOPHYSICS* 82, G57–G69.
1064 doi:10.1190/GEO2016-0332.1
- 1065 Skilling, J. and Bryan, R. (1984). Maximum entropy image reconstruction-general algorithm. *Monthly*
1066 *Notices of the Royal Astronomical Society*, Vol. 211, NO. 1, P. 111, 1984 211, 111
- 1067 Soler, S. R. and Uieda, L. (2021). Gradient-boosted equivalent sources. *Geophysical Journal International*
1068 227, 1768–1783. doi:10.1093/gji/ggab297
- 1069 Takahashi, D., Oliveira Jr., V. C., and Barbosa, V. C. (2022). Convolutional equivalent layer for magnetic
1070 data processing. *Geophysics* 87, 1–59
- 1071 Takahashi, D., Oliveira Jr., V. C., and Barbosa, V. C. F. (2020). Convolutional equivalent layer for gravity
1072 data processing. *GEOPHYSICS* 85, G129–G141. doi:10.1190/geo2019-0826.1
- 1073 Tassinari, C. C. and Macambira, M. J. (1999). Geochronological provinces of the Amazonian Craton.
1074 *Episodes* 22, 174–182
- 1075 Teixeira, W., Tassinari, C., Cordani, U. G., and Kawashita, K. (1989). A review of the geochronology of
1076 the Amazonian Craton: Tectonic implications. *Precambrian Research* 42, 213–227
- 1077 van der Sluis, A. and van der Vorst, H. A. (1987). Numerical solution of large, sparse linear algebraic
1078 systems arising from tomographic problems. In *Seismic tomography with applications in global*
1079 *seismology and exploration geophysics*, ed. G. Nolet (D. Reidel Publishing Company), chap. 3. 49–83
- 1080 Van Loan, C. F. (1992). *Computational Frameworks for the fast Fourier transform*. Frontiers in Applied
1081 Mathematics (SIAM)
- 1082 Villas, R. N. and Santos, M. (2001). Gold deposits of the Carajás mineral province: deposit types and
1083 metallogenesis. *Mineralium Deposita* 36, 300–331

- 1084 Xia, J. and Sprowl, D. R. (1991). Correction of topographic distortion in gravity data. *Geophysics* 56,
1085 537–541
- 1086 Xia, J., Sprowl, D. R., and Adkins-Helgeson, D. (1993). Correction of topographic distortions in potential-
1087 field data; a fast and accurate approach. *Geophysics* 58, 515–523. doi:10.1190/1.1443434
- 1088 Zhao, G., Chen, B., Chen, L., Liu, J., and Ren, Z. (2018). High-accuracy 3D Fourier forward modeling
1089 of gravity field based on the Gauss-FFT technique. *Journal of Applied Geophysics* 150, 294–303.
1090 doi:10.1016/j.jappgeo.2018.01.002
- 1091 Zidarov, D. (1965). Solution of some inverse problems of applied geophysics. *Geophysical Prospecting*
1092 13, 240–246. doi:10.1111/j.1365-2478.1965.tb01932.x

11 ALGORITHMS

Algorithm 1: Generic pseudo-code for the CGLS applied to the overdetermined problem (equation 22) for the particular case in which $\mathbf{H} = \mathbf{I}_P$ (equation 9 and subsection 3.2), $\mu = 0$ (equation 11), $\mathbf{W}_d = \mathbf{I}_D$ (equation 12) and $\bar{\mathbf{p}} = \mathbf{0}$ (equation 14), where \mathbf{I}_P and \mathbf{I}_D are the identities of order P and D , respectively.

Initialization :

- 1 Compute \mathbf{G} ;
- 2 Set $\mathbf{r} = \mathbf{d}$ and compute $\delta = \|\mathbf{r}\|/D$;
- 3 Compute $\boldsymbol{\vartheta} = \mathbf{G}^\top \mathbf{r}$ and $\rho_0 = \boldsymbol{\vartheta}^\top \boldsymbol{\vartheta}$;
- 4 Set $\tilde{\mathbf{p}} = \mathbf{0}$, $\tau = 0$ and $\boldsymbol{\eta} = \mathbf{0}$;
- 5 $m = 1$;
- 6 **while** ($\delta > \epsilon$) **and** ($m < \text{ITMAX}$) **do**
- 7 Update $\boldsymbol{\eta} \leftarrow \boldsymbol{\vartheta} + \tau \boldsymbol{\eta}$;
- 8 Compute $\boldsymbol{\nu} = \mathbf{G} \boldsymbol{\eta}$;
- 9 Compute $v = \rho_0 / (\boldsymbol{\nu}^\top \boldsymbol{\nu})$;
- 10 Update $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}} + v \boldsymbol{\eta}$;
- 11 Update $\mathbf{r} \leftarrow \mathbf{r} - v \boldsymbol{\nu}$ and $\delta \leftarrow \|v \boldsymbol{\nu}\|/D$;
- 12 Compute $\boldsymbol{\vartheta} = \mathbf{G}^\top \mathbf{r}$ and $\rho = \boldsymbol{\vartheta}^\top \boldsymbol{\vartheta}$;
- 13 Compute $\tau = \rho / \rho_0$;
- 14 Update $\rho_0 \leftarrow \rho$;
- 15 $m \leftarrow m + 1$;
- 16 **end**

Algorithm 2: Generic pseudo-code for the method proposed by Leão and Silva (1989).

Initialization :

- 1 Set the indices \mathbf{i}^m for each data window, $m \in \{1 : M\}$;
- 2 Set the indices \mathbf{j}^m for each source window, $m \in \{1 : M\}$;
- 3 Set the constant depth $z_0 + \Delta z_0$ for all equivalent sources ;
- 4 Compute the vector \mathbf{a}' associated with the desired potential-field transformation ;
- 5 Compute the matrix \mathbf{G}' ;
- 6 Compute the matrix \mathbf{B}' (equation 34) ;
- 7 Compute the vector $(\mathbf{a}')^\top \mathbf{B}'$;
- 8 $m = 1$;
- 9 **while** $m < M$ **do**
- 10 Compute t_c^m (equation 33) ;
- 11 $m \leftarrow m + 1$;
- 12 **end**

Algorithm 3: Generic pseudo-code for the method proposed by Soler and Uieda (2021).

Initialization :

```

1 Set the indices  $\mathbf{i}^m$  for each data window,  $m \in \{1 : M\}$  ;
2 Set the indices  $\mathbf{j}^m$  for each source window,  $m \in \{1 : M\}$  ;
3 Set the depth of all equivalent sources ;
4 Set a  $D \times 1$  residuals vector  $\mathbf{r} = \mathbf{d}$  ;
5 Set a  $P \times 1$  vector  $\tilde{\mathbf{p}} = \mathbf{0}$  ;
6  $m = 1$  ;
7 while  $m < M$  do
8   | Set the matrix  $\mathbf{W}_d^m$  ;
9   | Compute the matrix  $\mathbf{G}^m$  ;
10  | Compute  $\tilde{\mathbf{p}}^m$  (equation 36) ;
11  |  $\tilde{\mathbf{p}}[\mathbf{j}^m] \leftarrow \tilde{\mathbf{p}}[\mathbf{j}^m] + \tilde{\mathbf{p}}^m$  ;
12  |  $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{G}[:, \mathbf{j}^m] \tilde{\mathbf{p}}^m$  ;
13  |  $m \leftarrow m + 1$  ;
14 end

```

Algorithm 4: Generic pseudo-code for the method proposed by Cordell (1992).

Initialization :

```

1 Compute a  $D \times 1$  vector  $\Delta\mathbf{z}$  whose  $i$ -th element  $\Delta z_i$  is a vertical distance controlling the depth of
  the  $i$ -th equivalent source,  $i \in \{1 : D\}$  ;
2 Set a tolerance  $\epsilon$  ;
3 Set a maximum number of iterations ITMAX ;
4 Set a  $D \times 1$  residuals vector  $\mathbf{r} = \mathbf{d}$  ;
5 Set a  $D \times 1$  vector  $\tilde{\mathbf{p}} = \mathbf{0}$  ;
6 Define the maximum absolute value  $r_{\max}$  in  $\mathbf{r}$  ;
7  $m = 1$  ;
8 while ( $r_{\max} > \epsilon$ ) and ( $m < \text{ITMAX}$ ) do
9   | Define the coordinates  $(x_{\max}, y_{\max}, z_{\max})$  and index  $i_{\max}$  of the observation point associated with
      $r_{\max}$  ;
10  |  $\tilde{\mathbf{p}}[i_{\max}] \leftarrow \tilde{\mathbf{p}}[i_{\max}] + (r_{\max} \Delta\mathbf{z}[i_{\max}])$  ;
11  |  $\mathbf{r} \leftarrow \mathbf{r} - (\mathbf{G}[:, i_{\max}] \tilde{\mathbf{p}}[i_{\max}])$  ;
12  | Define the new  $r_{\max}$  in  $\mathbf{r}$  ;
13  |  $m \leftarrow m + 1$  ;
14 end

```

Algorithm 5: Generic pseudo-code for the method proposed by Mendonça and Silva (1994).

Initialization :

- 1 Set a regular grid of P equivalent sources at a horizontal plane z_0 ;
- 2 Set a tolerance ϵ ;
- 3 Set a $D \times 1$ residuals vector $\mathbf{r} = \mathbf{d}$;
- 4 Define the maximum absolute value r_{\max} in \mathbf{r} ;
- 5 Define the index i_{\max} of r_{\max} ;
- 6 Define the list of indices \mathbf{i}_r of the remaining data in \mathbf{r} ;
- 7 Define $\mathbf{d}_e = \mathbf{d}[i_{\max}]$;
- 8 Compute $(\mathbf{F} + \mu \mathbf{I}_{D_e})$ and \mathbf{G}_e ;
- 9 Compute $\tilde{\mathbf{p}}$ (equation 40) ;
- 10 Compute $\mathbf{r} = \mathbf{d}[\mathbf{i}_r] - \mathbf{G}[\mathbf{i}_r, :] \tilde{\mathbf{p}}$;
- 11 Define the maximum absolute value r_{\max} in \mathbf{r} ;
- 12 **while** ($r_{\max} > \epsilon$) **do**
- 13 Define the index i_{\max} of r_{\max} ;
- 14 Define the list of indices \mathbf{i}_r of the remaining elements in \mathbf{r} ;
- 15 $\mathbf{d}_e \leftarrow \begin{bmatrix} \mathbf{d}_e \\ \mathbf{d}[i_{\max}] \end{bmatrix}$;
- 16 Update $(\mathbf{F} + \mu \mathbf{I}_{D_e})$ and \mathbf{G}_e ;
- 17 Update $\tilde{\mathbf{p}}$ (equation 40) ;
- 18 Update $\mathbf{r} = \mathbf{d}[\mathbf{i}_r] - \mathbf{G}[\mathbf{i}_r, :] \tilde{\mathbf{p}}$;
- 19 Define the maximum absolute value r_{\max} in \mathbf{r} ;
- 20 **end**

Algorithm 6: Generic pseudo-code for the iterative method proposed by Siqueira et al. (2017). The symbol “ \circ ” denotes the entrywise or Hadamard product (e.g., Horn and Johnson, 1991, p. 298) and σ is a $P \times 1$ vector whose j -th element is the ratio of a predefined element of area centered at the j -th equivalent source and the term $2\pi\gamma$, where γ is the gravitational constant.

Initialization :

- 1 Set P equivalent sources on a horizontal plane z_0 ;
- 2 Set a tolerance ϵ ;
- 3 Set a maximum number of iterations ITMAX ;
- 4 Set an auxiliary vector σ ;
- 5 Compute $\tilde{\mathbf{p}} = \mathbf{0}$;
- 6 Compute \mathbf{G} (equation 3) ;
- 7 Compute $\mathbf{r} = \mathbf{d}$;
- 8 Compute $\delta = \|\mathbf{r}\|/D$;
- 9 $m = 1$;
- 10 **while** ($\delta > \epsilon$) **and** ($m < \text{ITMAX}$) **do**
- 11 Compute $\Delta\mathbf{p} = \sigma \circ \mathbf{r}$;
- 12 Update $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}} + \Delta\mathbf{p}$;
- 13 Compute $\nu = \mathbf{G} \Delta\mathbf{p}$;
- 14 Update $\mathbf{r} \leftarrow \mathbf{r} - \nu$;
- 15 Compute $\delta = \|\nu\|/D$;
- 16 $m \leftarrow m + 1$;
- 17 **end**

Algorithm 7: Generic pseudo-code for the convolutional equivalent-layer method proposed by Takahashi et al. (2020, 2022).

Initialization :

- 1 Set the regular grid of P equivalent sources on a horizontal plane z_0 ;
 - 2 Set a tolerance ϵ and a maximum number of iterations ITMAX ;
 - 3 Compute the first column $\mathbf{G}[:, 1]$ and row $\mathbf{G}[1, :]$ of the sensitivity matrix \mathbf{G} (equation 3) for the particular case in which it has a BTTB structure (equation 51);
 - 4 Rearrange the elements of $\mathbf{G}[:, 1]$ into matrix \mathcal{C} , compute its 2D-DFT via 2D-FFT and multiply by $\sqrt{4D}$ to obtain a matrix \mathcal{L}' (equation 66);
 - 5 Rearrange the elements of $\mathbf{G}[1, :]$ into matrix \mathcal{C} , compute its 2D-DFT via 2D-FFT and multiply by $\sqrt{4D}$ to obtain a matrix \mathcal{L}'' (equation 66);
 - 6 Set $\tilde{\mathbf{p}} = \mathbf{0}$;
 - 7 Set $\mathbf{r} = \mathbf{d}$ and compute $\delta = \|\mathbf{r}\|/D$;
 - 8 Compute $\vartheta = \mathbf{G}^\top \mathbf{r}$ (Algorithm 8) and $\rho_0 = \vartheta^\top \vartheta$;
 - 9 Set $\tau = 0$ and $\eta = \mathbf{0}$;
 - 10 $m = 1$;
 - 11 **while** ($\delta > \epsilon$) **and** ($m < \text{ITMAX}$) **do**
 - 12 Update $\eta \leftarrow \vartheta + \tau \eta$;
 - 13 Compute $\nu = \mathbf{G} \eta$ (Algorithm 8);
 - 14 Compute $v = \rho_0 / (\nu^\top \nu)$;
 - 15 Update $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}} + v \eta$;
 - 16 Update $\mathbf{r} \leftarrow \mathbf{r} - v \nu$ and $\delta \leftarrow \|v \nu\|/D$;
 - 17 Compute $\vartheta = \mathbf{G}^\top \mathbf{r}$ (Algorithm 8) and $\rho = \vartheta^\top \vartheta$;
 - 18 Compute $\tau = \rho / \rho_0$;
 - 19 Update $\rho_0 \leftarrow \rho$;
 - 20 $m \leftarrow m + 1$;
 - 21 **end**
-

Algorithm 8: Pseudo-code for computing the generic matrix-vector products given by equations 52 and 53 via fast 2D discrete convolution for a given vector \mathbf{v} (equation 54) and matrix \mathcal{L} (equation 66).

- 1 Rearrange the elements of \mathbf{v} (equations 52 and 54) into the matrix \mathcal{V}_c (equation 65);
 - 2 Compute $\mathcal{F}_{2N_B} \mathcal{V}_c \mathcal{F}_{2N_b}$ via 2D-FFT;
 - 3 Compute the Hadamard product with matrix \mathcal{L} (equation 66);
 - 4 Compute 2D-IDFT via 2D-FFT to obtain matrix \mathcal{W}_c (65);
 - 5 Retrieve \mathbf{w} (equations 52 and 54) from \mathbf{w}_c (equations 55–57);
-

12 TABLES

Reference	Term	flops
eq. 10	$\mathbf{G} \mathbf{H}$	$2DQP$
eq. 15	$\mathbf{H} \tilde{\mathbf{q}}$	$2PQ$
eq. 22	$(\mathbf{G} \mathbf{H})^\top (\mathbf{G} \mathbf{H})$	$2Q^2D$
eq. 22	$(\mathbf{G} \mathbf{H})^\top \delta_d$	$2QD$
eq. 23	$(\mathbf{G} \mathbf{H}) (\mathbf{G} \mathbf{H})^\top$	$2D^2Q$
eq. 23	$(\mathbf{G} \mathbf{H})^\top \mathbf{u}$	$2QD$
eq. 25	lower triangle of \mathcal{G}	$D^3/3$ or $Q^3/3$
eq. 26	solve triangular systems	$2D^2$ or $2Q^2$
Alg. 1	$\boldsymbol{\eta} \leftarrow \boldsymbol{\vartheta} + \tau \boldsymbol{\eta}$	$2Q$
Alg. 1	$\boldsymbol{\vartheta}^\top \boldsymbol{\vartheta}$	$2Q$
Alg. 6	$\boldsymbol{\sigma} \circ \mathbf{d}$	D

Table 1. Total number of flops associated with some useful terms according to Golub and Van Loan (2013, p. 12). The number of flops associated with equations 25 and 26 depends if the problem is over or underdetermined. Note that $P = Q$ for the case in which $\mathbf{H} = \mathbf{I}_P$ (subsection 3.2). The term associated with Algorithm 1 is a vector update called *saxpy* (Golub and Van Loan, 2013, p. 4). The terms defined here are references to compute the total number of flops throughout the manuscript.

13 FIGURES

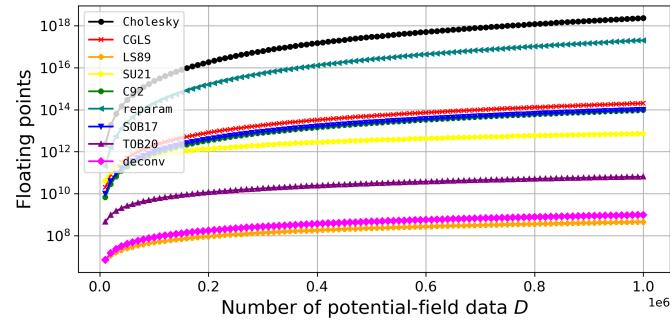


Figure 1. Total number of flops for different equivalent-layer methods (equations 27, 28, 35, 37, 38, 43, 50, 67, and 70). The number of potential-field data D varies from 10,000 to 1,000,000.

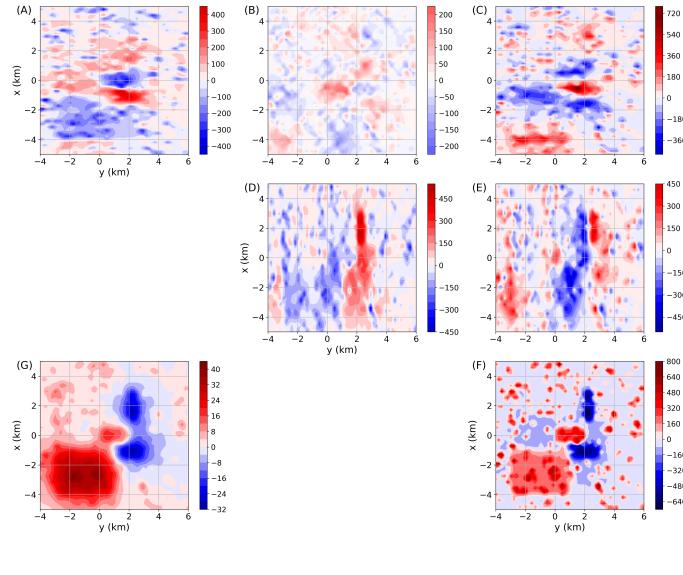


Figure 2. Noise-free gravity data produced by an ensemble of rectangular prisms (not shown). The data are located on a regular grid of 50×50 points. Panels (A)–(F) show, respectively, the xx , xy , xz , yy , yz and zz component of the gravity-gradient tensor in Eötvös (E). Panel (G) shows the gravity disturbance in milligals (mGal).

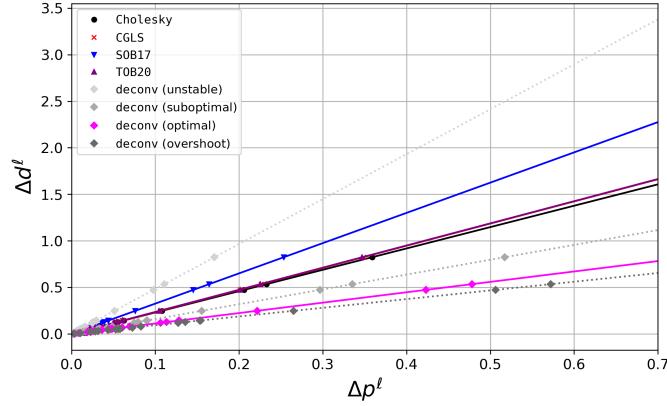


Figure 3. Numerical stability curves obtained for the 21 synthetic gravity data sets by using the Cholesky factorization with $\mu \approx 2 \times 10^{-2}$, CGLS, iterative method ((SOB17)) and iterative deconvolution (T0B20) with 50 iterations each (Algorithms 1, 6 and 7) and the direct deconvolution (deconv.) computed with four different values for ζ (equation 69): 0, 10^{-18} (overshoot), 10^{-22} (optimal) and 10^{-28} (suboptimal). The stability parameter κ (equation 29) obtained for the eight curves described above are 2.29 (Cholesky), 2.38 (CGLS), 3.25 (SOB17), 2.38 (T0B20), 4.83, 1.59, 1.11 and 0.93 (deconv. with null, suboptimal, optimal and overshoot ζ).

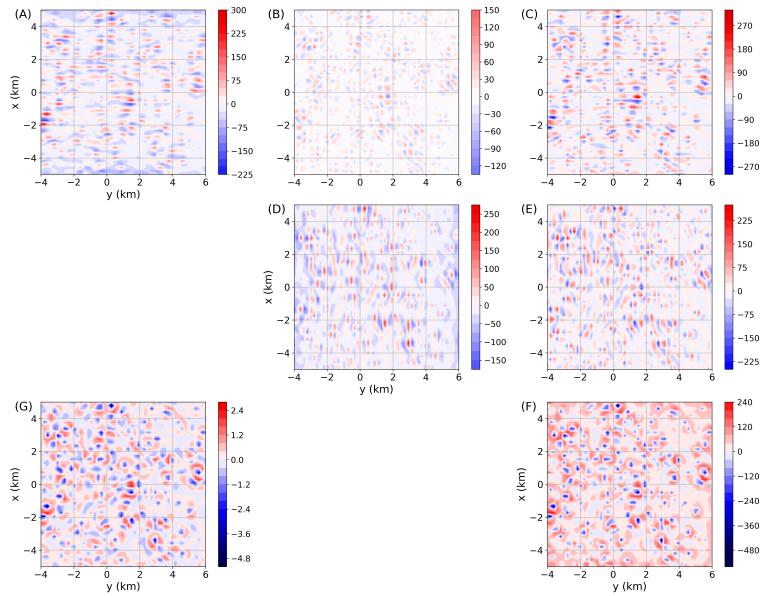


Figure 4. Residuals between the gravity data predicted by the equivalent layer estimated with the iterative deconvolution (T0B20) (Algorithm 7). The inverse problems was solved by using the noise-corrupted gravity disturbance having the maximum noise level (not shown). Panels (A)–(F) show the residuals between the predicted and noise-free gravity gradient data (Figure 2) associated with the xx , xy , xz , yy , yz and zz components of the gravity-gradient tensor, respectively. The values are in Eötvös. (G) Shows the residuals between the predicted and noise-corrupted gravity disturbances. The values are in milligals (mGal).

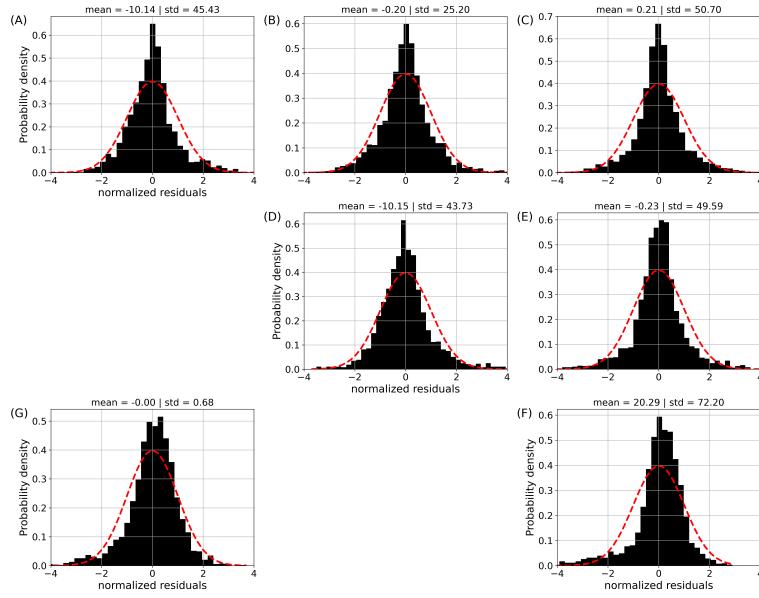


Figure 5. Histograms of the residuals shown in Figure 4. The residuals were normalized by removing the mean and dividing the difference by the standard deviation. Panels (A)–(F) show the histograms associated with the xx , xy , xz , yy , yz and zz components of the gravity-gradient tensor, respectively. (G) Shows the histogram of the residuals between the predicted and noise-corrupted gravity disturbances.

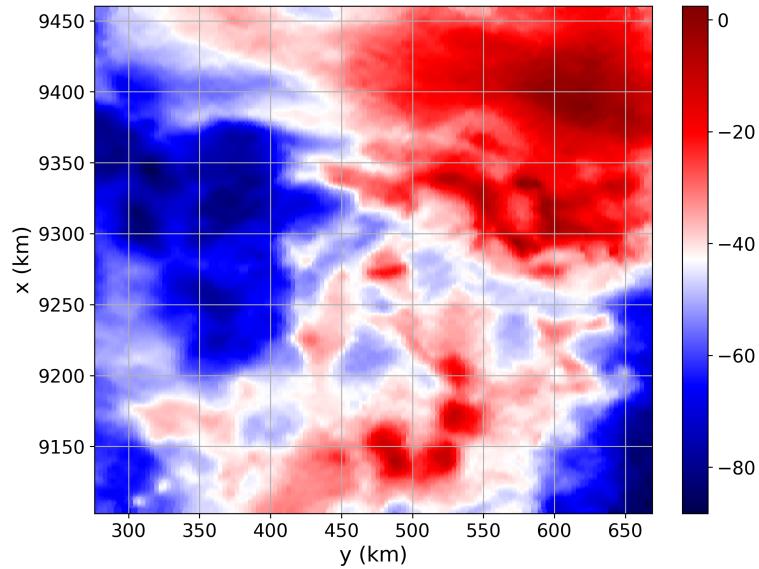


Figure 6. Field aerogravimetric data over Carajás, Brazil. There are $D = 500,000$ observations located on regular grid of $1,000 \times 500$ points.

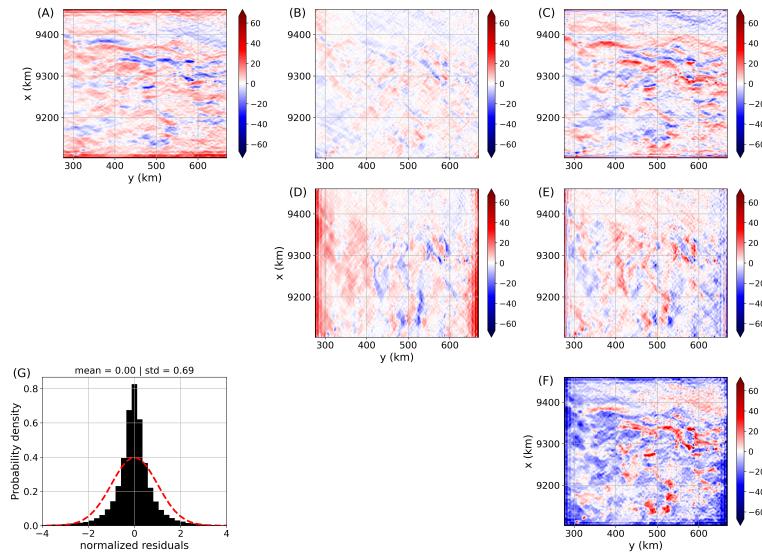


Figure 7. Estimated gravity-gradient tensor components over Carajás, Brazil. Panels (A)–(F) show, respectively, the xx , xy , xz , yy , yz and zz components of the gravity-gradient tensor in Eötvös. Panel (G) shows the histogram of the residuals between predicted data (not shown) and field data (Figure 6). The residuals were normalized by removing the mean and dividing the difference by the standard deviation. The results were generated by applying the iterative deconvolution (TOB20) (Algorithm 7) with 50 iterations.