ABSTRACT

We have developed an efficient and very fast equivalent layer technique for gravity data processing by modifying an iterative method grounded on excess mass constraint that does not require the solution of linear systems. Taking advantage of the property of being symmetric Block-Toeplitz Toeplitz-block (BTTB) matrix, that raises when regular grids of observation points and equivalent sources (point masses) are used to set up a fictitious equivalent layer, we have developed an algorithm which greatly reduces the number of flops and memory RAM necessary to estimate of a 2D mass distribution over the equivalent layer. The algorithm is based on the structure of symmetric BTTB matrix, where all its elements consist of the elements of the first column, which in turn can be embedded into a symmetric Block-Circulant Circulant-Block (BCCB) matrix. Likewise, only the first column of the BCCB matrix is needed to reconstruct the full matrix completely. From the first column of BCCB matrix, its eigenvalues can be calculated using the fast Fourier transform, which can be used to readily compute the matrix-vector product of the forward modeling in the fast equivalent-layer technique. As a result, our method is efficient to process very large datasets using either fine- or mid-grid meshes. The larger the dataset, the faster and more efficient our method becomes compared to the available equivalent-layer techniques. Synthetic tests demonstrate the ability of our method to satisfactorily upward- and downward-continuing the gravity data. Test with real data from Carajás, Brazil, shows its applicability to process very large dataset at low computational cost.

INTRODUCTION

The equivalent layer is a well-known technique for processing potential-field data in applied geophysics since 1960. It comes from potential theory as a mathematical solution of the Laplace's equation, in the region above the sources, by using the Dirichlet boundary condition (Kellogg, 1929). This theory states that any potential-field data produced by an arbitrary 3D physical property distribution can be exactly reproduced by a fictitious layer located at any depth and having a continuous 2D physical property distribution. In practical situations, the layer is approximated by a finite set of sources (e.g., point masses or dipoles) and their physical properties are estimated by solving a linear system of equations that yield an acceptable potential-field data fit. These fictitious sources are called equivalent sources.

Many previous works have used the equivalent layer as a processing technique of potential-field data. Dampney (1969) used the equivalent-layer technique for gridding and for computing the upward continuation of the potential-field data. Cordell (1992) and Mendonça and Silva (1994) used it for interpolating and gridding potential-field data. Emilia (1973), Hansen and Miyazaki (1984) and Li and Oldenburg (2010) used it for upward continuation of the potential-field data. Silva (1986), Leão and Silva (1989), Guspí and Novara (2009), and Oliveira Jr. et al. (2013) used it for reducing the magnetic data to the pole. Boggs and Dransfield (2004) used it for combining multiple data sets and Barnes and Lumley (2011) for gradient-data processing.

The classic equivalent layer formulation consists in estimating the physical-property distribution within a layer, composed by a set of equivalent sources, by solving a linear system of equations formed by harmonic functions (e.g.,the inverse of the distance between the ob-

servation point and the equivalent source). When these observation points and equivalent sources are regularly spaced, a Toeplitz system arises. Toeplitz systems are well-known in many branches of science as in (1) mathematics, for solving partial and ordinary diferential equations (e.g., Lin et al., 2003); (2) image processing (e.g., Chan et al., 1999) and; (3) computational neuroscience (e.g., Wray and Green, 1994). (Jin, 2003) and (Chan and Jin, 2007) give many examples of applications for Toeplitz systems.

In potential-field methods, the properties of Toeplitz system were used for downward continuation (Zhang et al., 2016) and for 3D gravity-data inversion using a 2D multilayer model (Zhang and Wong, 2015). In the particular case of gravity data, the kernel generates a linear system with a matrix known as symmetric Block-Toeplitz Toeplitz-Block (BTTB).

A wide variety of applications in mathematics and engineers that fall into Toeplitz systems propelled the development of a large variety of methods for solving them. Direct methods were conceived by (Levinson, 1946) and by (Trench, 1964). Currently the conjugate gradient is used in most cases. In Grenander and Szegö (1958), Szegö noticed that a circulant matrix can be diagonalized by taking the fast Fourier trasform of its first column, making it possible to calculate the matrix-vector product and solve the system with low computational cost (Strang and Aarikka, 1986; Olkin, 1986). Chan and Jin (2007) show some preconditioners to embedd the Toeplitz and BTTB matrices into, respectively, circulant matrices and Block-Circulant Circulant-Block (BCCB) by solving the system applying the conjugate gradient method.

Although the use of the equivalent-layer technique increased over the last decades, one of the biggest problem is still its high computational cost for processing large-data sets. Siqueira et al. (2017) developed a computationally efficient scheme for processing gravity

data. This scheme does not solve a linear system, instead uses an iterative process that corrects the physical-property distribution over the equivalent layer by adding mass corrections that are proportional to the gravity residual. Although efficient, the method presented by Siqueira et al. (2017) requires, at each iteration, the full computation of the forward problem to guarantee the convergence of the algorithm. The time spent on forward modeling accounts for most of the total computation time of their method.

We propose the use of BTTB and BCCB matrices properties to efficiently solve the forward modeling in method of Siqueira et al. (2017), resulting in faster parameter estimation and the possibility to use very large datasets. Here, we show how the system memory (RAM) usage can be drastically decreased by calculating only the first row of the BTTB matrix and embedding into a BCCB matrix. Using the Szegö theorema combined with Strang and Aarikka (1986), the matrix-vector product can be accomplished with very low cost, reducing in some orders of magnitude the number of operations required to complete the process. We present synthetic tests to validate our proposal and real field data from Carajás, Brazil to demonstrate its applicability.

METHODOLOGY

Equivalent layer technique for gravity data processing

Let d_i^o be the observed gravity data at the point (x_i, y_i, z_i), $i = 1, ..., N$, of a local Cartesian system with x axis pointing to North, the y axis pointing to East and the z axis pointing downward. We consider that the gravity data are properly processed so that they represent the difference between the observed gravity (corrected from non-gravitational effects due to the vehicle motion) and the normal gravity, at the same point. This quantity is called

gravity disturbance (Hofmann-Wellenhof and Moritz, 2005). Several authors have discussed the differences between the gravity anomaly and the gravity disturbance, as well as proposed that the second is more appropriated for geophysical applications. A detailed discussion about this theoretical topic is beyond the scope of our work and we refer the reader to Li and Götze (2001); Fairhead et al. (2003); Hackney and Featherstone (2003); Hinze et al. (2005) and Vajda et al. (2006, 2007, 2008), for example.

In a local coordinate system, the gravity disturbance can be considered as the z - component (or vertical component) of the gravitational attraction exerted by gravity sources. As a consequence, it can be approximated by a linear combination of ...

$$\delta g(x_i, y_i, z_i) = \sum_{j=1}^{N} p_j a_{ij} \,, \tag{1}$$

$$a_{ij} = \frac{G\,(z_0 - z_i)\,10^{-5}}{[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_0)^2]^{\frac{3}{2}}} \,. \tag{2}$$

$$\mathbf{d}(\mathbf{p}) = \mathbf{A}\mathbf{p} \,, \tag{3}$$

The elements a_{ij} (equation 2) forming the matrix $\mathbf{A}$ (equation 3) are defined in terms of the coordinates x_i, y_i, x_j and y_j, which are associated with the observed data and the equivalent sources. For convenience, we designate these coordinates as *matrix coordinates* and the indices i and j as *matrix indices*.

$$\Psi(\mathbf{p}) = \|\mathbf{d}^o - \mathbf{d}(\mathbf{p})\|_2^2 + \mu \|\mathbf{p}\|_2^2 \,, \tag{4}$$

$$\hat{\mathbf{p}} = \left(\mathbf{A}^\top \mathbf{A} + \mu\, \mathbf{I}\right)^{-1} \mathbf{A}^\top \mathbf{d}^o \,. \tag{5}$$

Structure of matrix A for regular grids

Consider that the observed data are located on an $N_x \times N_y$ regular grid of points regularly spaced from Δx and Δy along the x and y directions, respectively, on a horizontal plane defined by the constant vertical coordinate $z_1 < z_0$. As a consequence, a given pair of matrix coordinates (x_i, y_i), defined by the matrix index i, $i = 1, \ldots, N = N_x N_y$, is equivalent to a pair of coordinates (x_k, y_l) given by:

$$x_i \equiv x_k = x_1 + [k(i) - 1]\ \Delta x \,, \tag{6}$$

and

$$y_i \equiv y_l = y_1 + [l(i) - 1]\ \Delta y \,, \tag{7}$$

where $k(i)$ and $l(i)$ are integer functions of the matrix index i. These equations can also be used to define the matrix coordinates x_j and y_j associated with the j-th equivalent source, $j = 1, \ldots, N = N_x N_y$. In this case, the integer functions are evaluated by using the index j instead of i. For convenience, we designate x_k and y_l as *grid coordinates* and the indices k and l as *grid indices*, which are computed with the integer functions.

The integer functions assume different forms depending on the orientation of the regular grid of data. Consider the case in which the grid is oriented along the x-axis (Figure 1a). For convenience, we designate these grids as *x-oriented grids*. For them, we have the following integer functions:

$$i(k, l) = (l - 1)\, N_x + k \quad , \tag{8}$$

$$l(i) = \left\lceil \frac{i}{N_x} \right\rceil \tag{9}$$

and

$$k(i) = i - \left\lceil \frac{i}{N_x} \right\rceil N_x + N_x \quad , \tag{10}$$

where $\lceil \cdot \rceil$ denotes the ceiling function (Graham et al., 1994, p. 67). These integer functions are defined in terms of the matrix index i, but they can be defined in the same way by using the index j. Figure 1a illustrates an x-oriented grid defined by $N_x = 4$ and $N_y = 3$. In this example, the matrix coordinates x_7 and y_7, defined by the matrix index $i = 7$ (or $j = 7$), are equivalent to the grid coordinates x_3 and y_2, which are defined by the grid indices $k = 3$ and $l = 2$, respectively. These indices are computed with equations 9 and 10, by using the matrix index $i = 7$ (or $j = 7$).

Now, consider the case in which the regular grid of data is oriented along the y-axis (Figure 1b). For convenience, we call them *y-oriented grids*. Similarly to x-oriented grids, we have the following integer functions associated with y-oriented grids:

$$i(k, l) = (k - 1) N_y + l \quad , \tag{11}$$

$$k(i) = \left\lceil \frac{i}{N_y} \right\rceil \tag{12}$$

and

$$l(i) = i - \left\lceil \frac{i}{N_y} \right\rceil N_y + N_y \quad . \tag{13}$$

Figure 1b illustrates an y-oriented grid defined by $N_x = 4$ and $N_y = 3$. In this example, the matrix coordinates x_7 and y_7, defined by the matrix index $i = 7$ (or $j = 7$), are equivalent to the grid coordinates x_3 and y_1, which are defined by the grid indices $k = 3$ and $l = 1$, respectively. Differently from the example shown in Figure 1a, the grid indices of the present example are computed with equations 12 and 13, by using the matrix index $i = 7$ (or $j = 7$).

The element a_{ij} (equation 2) can be rewritten by using equations 6 and 7, giving rise to:

$$a_{ij} = \frac{G\,\Delta z\,10^{-5}}{\left[\left(\Delta k_{ij}\,\Delta x\right)^2 + \left(\Delta l_{ij}\,\Delta y\right)^2 + \left(\Delta z\right)^2\right]^{\frac{3}{2}}}\,, \tag{14}$$

where $\Delta z = z_0 - z_1$, $\Delta k_{ij} = k(i) - k(j)$ (equations 10 and 12) and $\Delta l_{ij} = l(i) - l(j)$ (equations 9 and 13). Notice that the structure of matrix $\mathbf{A}$ (equation 3) for the case in which its elements are given by a_{ij} (equation 14) is defined by the coefficients Δk_{ij} and Δl_{ij}.

For x-oriented grids, the coefficients Δk_{ij} and Δl_{ij} are computed by using equations 10 and 9, respectively. In this case, $\mathbf{A}$ (equation 3) is a symmetric Block-Toeplitz Toeplitz-Block (BTTB) matrix (Chan and Jin, 2007, p. 67) composed of $N_y \times N_y$ blocks, where each block is a symmetric Toeplitz matrix formed by $N_x \times N_x$ elements. For y-oriented grids, the coefficients Δk_{ij} and Δl_{ij} are computed by using equations 12 and 13, respectively. In this case, $\mathbf{A}$ (equation 3) is a symmetric BTTB matrix composed of $N_x \times N_x$ blocks, where each block is a symmetric Toeplitz matrix formed by $N_y \times N_y$ elements. In both cases, the blocks lying at the same diagonal are equal to each other and those located above the main diagonal are equal to those located below. These symmetries come from the fact that the coefficients Δk_{ij} and Δl_{ij} are squared at the denominator of a_{ij} (equation 14).

We follow the common notation found in the literature and represent the blocks of $\mathbf{A}$ as $\mathbf{A}_m$. For x-oriented grids, the index m varies from 0, at the main diagonal, to $N_y - 1$, at the corners of $\mathbf{A}$. In this case, the index m is defined as an integer function of the matrix indices i and j as follows:

$$m(i,j) = \mid l(i) - l(j) \mid \quad , \tag{15}$$

where $l(i)$ and $l(j)$ are defined by equation 9. The elements forming the first column of $\mathbf{A}_m$

are conveniently represented as a_n^m, $n = 0, \ldots, N_x - 1$, where $m \equiv m(i,j)$ (equation 15) and the subscript index n is defined by the following integer function:

$$n(i,j) = \mid k(i) - k(j) \mid \quad , \tag{16}$$

where $k(i)$ and $k(j)$ are defined by equation 10. For y-oriented grids, the index m varies from 0, at the main diagonal, to $N_x - 1$, at the corners of $\mathbf{A}$. The index m vary from 0, at the main diagonal, to $N_x - 1$, at the corners of $\mathbf{A}$. In such grids, m is given by:

$$m(i,j) = \mid k(i) - k(j) \mid \quad , \tag{17}$$

where the integer functions $k(i)$ and $k(j)$ are defined by equation 12. In this case, the elements forming the first column of $\mathbf{A}_m$ are represented as a_n^m, $n = 0, \ldots, N_y - 1$, where $m \equiv m(i,j)$ (equation 17) and the subscript index n is defined by:

$$n(i,j) = \mid l(i) - l(j) \mid \quad , \tag{18}$$

where $l(i)$ and $l(j)$ are defined by equation 13. For convenience, we designate the indices m and n as *block indices*.

It is important noting that different matrix indices i or j produce the same absolute values for the grid indices k (equations 10 and 12) and l (equations 9 and 13). As a consequence, different pairs of matrix indices i and j generate the same absolute values for the coefficients Δk_{ij} and Δl_{ij} that compose the denominator of a_{ij} (equation 14). It means that elements a_{ij} defined by different matrix indices i and j have the same value. The key point for understanding the structure of matrix $\mathbf{A}$ is then, given a single element a_{ij} defined by matrix indices i and j, compute the grid indices k (equations 10 and 12) and l (equations 9 and 13). These grid indices are used to (1) compute the coefficients Δk_{ij} and Δl_{ij} and determine the value of a_{ij} with equation 14 and (2) compute the block indices m

(equations 15, and 17) and n (equations 16 and and 18) and determine the positions where a_{ij} appears in matrix $\mathbf{A}$.

Consider the x-oriented grid of $N_x \times N_y$ points shown in Figure 1a, with $N_x = 4$, $N_y = 3$ and $N = N_x\, N_y = 12$. This grid results in a matrix $\mathbf{A}$ given by:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{A}_1 & \mathbf{A}_0 & \mathbf{A}_1 \\ \mathbf{A}_2 & \mathbf{A}_1 & \mathbf{A}_2 \end{bmatrix} \quad , \tag{19}$$

where $\mathbf{A}_m$, $m = 0, 1, 2$, are symmetric Toeplitz matrices given by:

$$\mathbf{A}_m = \begin{bmatrix} a_0^m & a_1^m & a_2^m & a_3^m \\ a_1^m & a_0^m & a_1^m & a_2^m \\ a_2^m & a_1^m & a_0^m & a_1^m \\ a_3^m & a_2^m & a_1^m & a_0^m \end{bmatrix} \quad . \tag{20}$$

To illustrate the relationship between the block indices (m and n) and the matrix indices (i and j), consider the element a_{ij} defined by $i = 2$ and $j = 10$, which is located at the upper right corner of $\mathbf{A}$, in the 2nd line and 10th column. By using equations 9 and 10, we obtain the grid indices $l(i) = 1$, $l(j) = 3$, $k(i) = 2$ and $k(j) = 2$. These grid indices result in the coefficients $\Delta k_{ij} = 0$ and $\Delta l_{ij} = -2$, which are used to compute the element a_{ij} (equation 14), as well as in the block indices $m = 2$ (equation 15) and $n = 0$ (equation 16). These block indices indicate that this element a_{ij} appears in the main diagonal of the blocks $\mathbf{A}_2$, which are located at the corners of $\mathbf{A}$. To verify this, let us take the matrix indices associated with these elements. They are $(i, j) = (1, 9)$, $(2, 10)$, $(3, 11)$, $(4, 12)$, $(9, 1)$, $(10, 2)$, $(11, 3)$ and $(12, 4)$. By using these matrix indices, it is easy to verify that all of them produce the same grid indices $l(i)$, $l(j)$, $k(i)$ and $k(j)$ (equations 9 and 10) as those associated with the element defined by $i = 2$ and $j = 10$. Consequently, all of them produce elements a_{ij}

(equation 14) having the same value. Besides, it is also easy to verify that all these matrix indices produce the same block indices $m = 2$ (equation 15) and $n = 0$ (equation 16). By repeating this procedure for all elements a_{ij}, $i = 0, 1, 2, 3$, $j = 0, 1, 2$, forming the matrix $\mathbf{A}$ obtained from our x-oriented grid (Figure 1a), we can verify the well-defined pattern represented by equations 19 and 20. This procedure can also be used to verify that the matrix $\mathbf{A}$ obtained from the y-oriented grid illustrated in Figure 1b is given by

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_0 & \mathbf{A}_1 & \mathbf{A}_2 & \mathbf{A}_3 \\ \mathbf{A}_1 & \mathbf{A}_0 & \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{A}_2 & \mathbf{A}_1 & \mathbf{A}_2 & \mathbf{A}_1 \\ \mathbf{A}_3 & \mathbf{A}_2 & \mathbf{A}_1 & \mathbf{A}_0 \end{bmatrix} \quad , \tag{21}$$

where $\mathbf{A}_m$, $m = 0, 1, 2, 3$, are symmetric Toeplitz matrices given by:

$$\mathbf{A}_m = \begin{bmatrix} a_0^m & a_1^m & a_2^m \\ a_1^m & a_0^m & a_1^m \\ a_2^m & a_1^m & a_0^m \end{bmatrix} \quad . \tag{22}$$

BTTB matrix-vector product

As previous discussed in the fast equivalent layer technique section, the matrix-vector product accounts for most of the computational cost.

When large data sets are used, this operation can take some time and even be prohibited by memory RAM shortage.

In order to lessen this problem we transform the BTTB matrix $\mathbf{A}$ into a Block-Circulating Circulating-Block (BCCB) matrix $\mathbf{C}$ and use its eigenvalues to carry the product of $\mathbf{A}$ and an arbitrary vector $\mathbf{p}$.

This strategy has been successfully applied by Zhang and Wong (2015) and Zhang et al. (2016) for optimizing the computational cost of 3D gravity inversion and downward continuation of potential field, respectively. Here, we use this strategy for improving the computational efficiency of the fast equivalent layer method proposed by Siqueira et al. (2017).

Following the example of our BTTB matrix $\mathbf{A}$ (equation **??**), its transformation into a BCCB matrix $\mathbf{C}$ is given by:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} \\ \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} \\ \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} \\ \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} \\ \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} \\ \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} \\ \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} \\ \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} \end{bmatrix}, \tag{23}$$

where each block $\mathbf{C}_l$ is:

$$\mathbf{C}_l = \begin{bmatrix} \mathbf{A}_l & \times \\ \times & \mathbf{A}_l \end{bmatrix} = \begin{bmatrix} a_0^\ell & a_1^\ell & a_2^\ell & 0^\ell & a_2^\ell & a_1^\ell \\ a_1^\ell & a_0^\ell & a_1^\ell & a_2^\ell & 0^\ell & a_2^\ell \\ a_2^\ell & a_1^\ell & a_0^\ell & a_1^\ell & a_2^\ell & 0^\ell \\ 0^\ell & a_2^\ell & a_1^\ell & a_0^\ell & a_1^\ell & a_2^\ell \\ a_2^\ell & 0^\ell & a_2^\ell & a_0^\ell & a_0^\ell & a_1^\ell \\ a_1^\ell & a_2^\ell & 0^\ell & a_2^\ell & a_1^\ell & a_0^\ell \end{bmatrix}. \tag{24}$$

The matrix $\mathbf{C}$ is a $4N_xN_y \times 4N_xN_y$ Block-Circulant matrix formed by Circulant-Blocks matrices. This matrix is formed by a grid of $2N_x \times 2N_x$ blocks, where each block is a $2N_y \times 2N_y$ matrix.

Instead of calculating the product $\mathbf{d} = \mathbf{Ap}$ (equation 3), we now carry the following multiplication:

$$\mathbf{Cv} = \mathbf{q}\,, \tag{25}$$

where $\mathbf{v}$ and $\mathbf{q}$ are $4N_xN_y \times 1$ vectors given, respectively, by:

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{N_x-1} \\ \mathbf{0}_{(2N_xN_y)} \end{bmatrix} \tag{26}$$

and

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_0 \\ \mathbf{q}_1 \\ \vdots \\ \mathbf{q}_{N_x-1} \\ \dagger_{(2N_xN_y)} \end{bmatrix}, \tag{27}$$

where $\mathbf{0}_{(2N_xN_y)}$ is a $2N_xN_y \times 1$ vector of zeros, $\dagger_{(2N_xN_y)}$ is $2N_xN_y \times 1$ vector to be drop out and $\mathbf{v}_\ell$ and $\mathbf{q}_\ell$, $\ell = 0, \ldots, N_x - 1$ are:

$$\mathbf{v}_\ell = \begin{bmatrix} \mathbf{p}_\ell \\ \\ \mathbf{0}_{(N_y)} \end{bmatrix} \tag{28}$$

and

$$\mathbf{q}_\ell = \begin{bmatrix} \mathbf{d}_\ell \\ \\ \dagger_{(N_y)} \end{bmatrix} , \tag{29}$$

where $\mathbf{0}_{(N_y)}$ is a $N_y \times 1$ vector of zeros, $\dagger_{(N_y)}$ is a $N_y \times 1$ vector to be drop out and $\mathbf{d}_\ell$ and $\mathbf{p}_\ell$ are $N_y \times 1$ vectors partitioned from the original $\mathbf{d}$ and $\mathbf{p}$ vectors (equation 3).

As demonstrated by **?**, circulant matrices can be diagonalized by taking the discrete Fourier transform (DFT), i.e., its eigenvalues can be easily calculated by a fast algorithm of DFT. For BCCB matrices **?** demonstrate that they satisfy:

$$\mathbf{C} = \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right)^* \mathbf{\Lambda} \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) , \tag{30}$$

where $\mathbf{F}_{(2N_y)}$ and $\mathbf{F}_{(2N_x)}$ are the matrices of the discrete Fourier transform, "$\otimes$" represents the Kronecker product, "$*$" the conjugate matrix and $\mathbf{\Lambda}$ is the diagonal matrix $4N_xN_y \times 4N_xN_y$ containing the eigenvalues of $\mathbf{C}$. By proper manipulating equation 30 it is possible to calculate the eigenvalues of the BCCB matrix using only its first column:

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{C} = \mathbf{\Lambda} \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \tag{31}$$

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{C}\,\mathbf{t}_1 = \mathbf{\Lambda} \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{t}_1 \tag{32}$$

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{c}_0 = \mathbf{\Lambda} \frac{1}{\sqrt{4N_xN_y}} \mathbf{1}_{(4N_xN_y)} \tag{33}$$

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{c}_0 = \frac{1}{\sqrt{4N_xN_y}} \lambda \quad , \tag{34}$$

where $\mathbf{t}_1$ is a $4N_xN_y \times 1$ vector with first element equal to 1 and all the remaining elements equal to 0, the vector $\mathbf{c}_0$ $4N_xN_y \times 1$ is the first column of $\mathbf{C}$, $\mathbf{1}_{(4N_xN_y)}$ is a $4N_xN_y \times 1$ with all elements equal to 1 and λ is a vector representing the diagonal of $\mathbf{\Lambda}$ (the eigenvalues of $\mathbf{C}$). Note that using one of the Kronecker product properties equation 34 can be calculated using the 2D-DFT (**?**):

$$\mathbf{F}_{(2N_x)}\mathbf{G}\,\mathbf{F}_{(2N_y)} \quad = \frac{1}{\sqrt{4N_xN_y}}\mathbf{L} \quad , \tag{35}$$

where $\mathbf{G}$ is a $2N_x \times 2N_y$ row-oriented matrix containing the elements of the first column $\mathbf{c}_0$ of $\mathbf{C}$ and $\mathbf{L}$ is a $2N_x \times 2N_y$ row-oriented matrix containing the eigenvalues λ.

By substituting equation 30 in equation 25, using the fast calculation of the eigenvalues of BCCB matrices (equation 35) and using the same previous Kronecker product property, the auxiliary matrix-vetor product $\mathbf{Cv} = \mathbf{q}$ can be rewritten as follows:

$$\mathbf{F}^*_{(2N_x)}\left[\mathbf{L} \circ \left(\mathbf{F}_{(2N_x)}\mathbf{V}\,\mathbf{F}_{(2N_y)}\right)\right]\mathbf{F}^*_{(2N_y)} = \mathbf{Q}\,, \tag{36}$$

where "$\circ$" denotes the Hadamard product, $\mathbf{L}$ is a $2N_x \times 2N_y$ row-oriented matrix containing the eigenvalues of $\mathbf{C}$, and $\mathbf{V}$ and $\mathbf{Q}$ are $2N_x \times 2N_y$ row-oriented matrices obtained from the vectors $\mathbf{v}$ and $\mathbf{q}$.

Note that in general, the first column of blocks forming a BCCB matrix $\mathbf{C}_\ell$ (equation 23) is given by:

$$[\mathbf{C}]_{(0)} = \begin{bmatrix} \mathbf{C}_0 \\ \mathbf{C}_1 \\ \vdots \\ \mathbf{C}_{N_x-2} \\ \mathbf{C}_{N_x-1} \\ \mathbf{0} \\ \mathbf{C}_{N_x-1} \\ \mathbf{C}_{N_x-2} \\ \vdots \\ \mathbf{C}_1 \end{bmatrix}, \tag{37}$$

where each block $\mathbf{C}_\ell$, $\ell = 0, \ldots, N_x - 1$, is a $2N_y \times 2N_y$ circulant matrix and $\mathbf{0}$ is a $2N_y \times 2N_y$ matrix with all elements equal to zero. Thus, the first column $\mathbf{c}_0$ of a circulant matrix $\mathbf{C}$ is given by:

$$\mathbf{c}_0 = \begin{bmatrix} a_{00} \\ a_{10} \\ \vdots \\ a_{(N_y-2)0} \\ a_{N_y-1)0} \\ 0 \\ a_{N_y-1)0} \\ a_{(N_y-2)0} \\ \vdots \\ a_{10} \end{bmatrix} . \tag{38}$$

To complete the process, after calculating the inverse to obtain $\mathbf{Q}$ it is necessary to rearrange its rows to obtain the vector $\mathbf{q}$ and also rearrange the elements of $\mathbf{q}$ to obtain the wanted vector $\mathbf{d}(\mathbf{p})$.

Computational performance

In a normal procedure of the fast equivalent layer proposed by Siqueira et al. (2017), at each iteration a full matrix $\mathbf{A}$ (equation 3) is multiplied by the estimated mass distribution parameter vector $\hat{\mathbf{p}}^k$ producing the predicted gravity data $\mathbf{d}(\mathbf{p})$ iteratively. As pointed in Siqueira et al. (2017) the number of flops (floating-point operations) necessary to estimate the N-dimensional parameter vector inside the iteration loop is

$$f_0 = N^{it}(3N + 2N^2) \, , \tag{39}$$

where N^{it} is the number of iterations. From equation 39 it is clear that the matrix-vector product ($2N^2$) accounts for most of the computational complexity in this method.

It is well known that FFT takes $2N\log_2(N)$ flops (Chu and George (1999)). Computing the eigenvalues of the BCCB matrix ($4N \times 4N$) and applying 2D-FFT on the parameter row-oriented matrix $\mathbf{V}$ (equation 36), takes $8N\log_2(4N)$ each. However, the sensitivity matrix does not change during the process thus, the eigenvalues of BCCB must be calculated only once, outside of the iteration. The multiplication of two complex numbers takes four real multiplications and two additions, which brings 6 times $4N$ flops to carry the Hadamard product. As it is necessary to compute the inverse FFT, another $8N\log_2(4N)$ must be taken in account. This lead us to a flops count in our method of

$$f_1 = 8N\log_2(4N) + N^{it}(27N + 16N\log_2(4N)). \tag{40}$$

Another major improvement of this methodology is the exoneration of calculating the full sensibility matrix $\mathbf{A}$ (equation 3). Each element needs 12 flops (equation 2), totalizing $12N^2$ flops for the full matrix. Calculating a single column of the BTTB matrix requires $12N$ flops. Thus, the full flops count of the method presented by Siqueira et al. (2017) is

$$f_s = 12N^2 + N^{it}(3N + 2N^2), \tag{41}$$

and it is decreased in our method to

$$f_f = 12N + 8N\log_2(4N) + N^{it}(27N + 16N\log(4N)). \tag{42}$$

Figure 1 shows the floating points to estimate the parameter vector using the fast

equivalent layer by using the method of Siqueira et al. (2017) (equation 39) and our approach (equation 40) versus the number of observation points varyig from $N = 5000$ to $N = 1\,000\,000$ with 50 iterations. The number of operations is drastically decreased.

Table 1 shows the system memory RAM usage needed to store the full sensibility matrix, the single column of the sensitivity matrix to form the BTTB matrix and the BCCB eigenvalues (8 times greater than that required by the BTTB first column). The quantities were computed for different numbers of data (N) with the same corresponding number of equivalent sources (N). Table 1 considers that each element of the matrix is a double-precision number, which requires 8 bytes of storage, except for the BCCB complex eigenvalues, which requires 16 bytes per element. Notice that 1 000 000 observation points requires nearly 7.6 Terabytes of memory RAM to store the whole sensibility matrix of the equivalent layer.

Using a PC with a Intel Core i7 4790@3.6GHz processor and 16 Gb of memory RAM, Figure 2 compares the running time of the Siqueira et al. (2017) method with the one of our work, considering a constant number of iterations equal to 50. Clearly, the major advantage of our approach is its computational efficiency that allows a rapid calculation of the gravity forward modeling with number of observations greater than 10 000. Because of the RAM available in this system, we could not perform this comparison with more observations. Therefore, the number of observation is limited to 22 500. Disregarding the limitation of 16 Gb of RAM, Figure 3 shows the running time of our method with 50 iterations and with the number of observations up to 25 millions. Our method requires 26:8 seconds to run one million of observations, whereas Siqueira et al. (2017) method took 48:3 seconds to run 22 500 observations

SYNTHETIC TESTS

In this section, we investigate the effectiveness of using the properties of BTTB and BCCB matrices (equation 25) to solve, at each iteration, the forward modeling (the matrix-vector product $\mathbf{A}\hat{\mathbf{p}}^k$) required in the fast equivalent-layer method proposed by Siqueira et al. (2017). We simulated three sources whose horizontal projections are shown in Figure 4 as black lines. These sources are two vertical prisms with density contrasts of $0.35\,\mathrm{g/cm^3}$ (upper-left prism) and $0.4\,\mathrm{g/cm^3}$ (upper-right prism) and a sphere with radius of 1 000 m with density contrast of $-0.5\,\mathrm{g/cm^3}$. Figure 4 shows the vertical component of gravity field generated by these sources contaminated with additive pseudorandom Gaussian noise with zero mean and standard deviation of 0.015 mGal.

The advantage of using the structures of BTTB and BCCB matrices to compute forward modeling in the fast-equivalent layer method (Siqueira et al., 2017) is grounded on the use of regular grids of data and equivalent sources. Hence, we created 10 000 observation points regularly spaced in a grid of 100×100 at 100 m height. We also set a grid of equivalent point masses, each one directly beneath each observation points, located at 300 m deep. Figures 5a and 6a show the fitted gravity data obtained, respectively, by the fast equivalent layer method and by our modified form of this method that computes the forward modeling using equation 25. The corresponding residuals (Figures 5b and 6b), defined as the difference between the observed (Figure 4) and fitted gravity data (Figures 5a and 6a), show means close to zero and standard deviations of 0.0144 mGal. Therefore, Figures 5 and 6 show that Siqueira et al. (2017) method and our modified version of this method produced virtually the same results. This excellent agreement is confirmed in Figures 7 and 8 which shows that there are virtually no differences, respectively, in the fitted data presented in Figures

5b and 6b and in the estimated mass distributions within the equivalent layers (not shown) yielded by both Siqueira et al. (2017) method and our modification of this method. These results (Figures 7 and 8) show that computing the matrix-vector product ($\mathbf{A}\hat{\mathbf{p}}^k$), required in the forward modeling, by means of embedding the BTTB matrix into a BCCB matrix (equation 25) yields practically the same result as the one produced by computing this matrix-vector product with a full matrix $\mathbf{A}$ as used in Siqueira et al. (2017).

We perform two forms of processing the gravity data (Figure 4) through the equivalent layer technique: the upward (Figure 9) and the downward (Figure 10) continuations. The upward height is 300 m and the downward is at 50 m. Either in the upward continuation (Figure 9) or in the downward continuation (Figure 10), the continued gravity data using the fast equivalent layer proposed by Siqueira et al. (2017) (Figures 9a and 10a) are in close agreement with those produced by our modification of Siqueira et al. (2017) method (Figures 9b and 10b). The residuals (Figures 9c and 10c) quantify this agreement since their means and standard deviations are close to zero in both continued gravity data using both methods. All the continued gravity data shown here (Figures 9 and 10) agree with the true ones (not shown). The most striking feature of these upward or the downward continuations concerns the total computation time. The computation time spent by our method is approximately 1 500 times faster than Siqueira et al. (2017) method.

REAL DATA TEST

Test with real data are conducted with the gravity data from Carajás, north of Brazil, were provided by the Geological Survey of Brazil (CPRM). The real aerogravimetric data were collected in 113 flight lines along northsouth direction with flight line spacing of 3 km and tie lines along eastwest direction at 12 km.

This airborne gravity survey was divided in two different areas, collected in different times, having samples spacing of 7.65 m and 15.21 m, totalizing 4,353,428 observation points. The height of the flight was fixed at 900 m. The gravity data (Figure 11) were gridded into a regularly spaced dataset of 250 000 observation points (500×500) with a grid spacing of 716.9311 km north-south and 781.7387 km east-west.

To apply our modification of the fast equivalent-layer method (Siqueira et al., 2017) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25), we set an equivalent layer at 300 m deep. Figure 12a shows the fitted gravity data after 50 iterations by applying our method. The residuals (Figure 12b), defined as the difference between the observed (Figure 11) and the predicted (Figure 12a) data, show an acceptable data fitting because they have a mean close to zero (0.0003 mGal) and a small standard deviation of 0.105 mGal which corresponds to approximately 0.1 % of the amplitude of the gravity data.

These small residuals indicate that our method yielded an estimated mass distribution (not shown) that can be used in the data processing. We perform upward-continuation of the real gravity data (Figure 11) at a constant height of 5 000 m over the real data. The upward-continued gravity data (Figure 13) seem a reasonable processing because of the attenuation of the short wavelenghts. By using our approach, the processing of the 250 000 observations was extremely fast and took 0.216 seconds.

CONCLUSIONS

By exploring the properties related to Block-Toeplitz Toeplitz-block (BTTB) of the sensitivity matrix in the gravity data processing, we have proposed a new efficient approach for

calculating the gravity-data forward modeling required in the iterative fast equivalent-layer technique grounded on excess mass constraint that does not demand the solution of linear systems. Its efficiency requires the use of regular grids of observations and equivalent sources (point masses). Our algorithm greatly reduces the number of flops necessary to estimate a 2D mass distribution within the equivalent layer that fits the observed gravity data. For example, when processing one million observations the number of flops is reduced in 104 times. Traditionally, such amount of data impractically requires 7.6 Terabytes of RAM memory to handle the full sensibility matrix. Rather, in our method, this matrix takes 61.035 Megabytes of RAM memory only.

Our method takes advantage of the symmetric BTTB system that arises when processing a harmonic function and considering that either the observations or the sources of the interpretative model (point of masses over the equivalent layer) are distributed on regular grids. Symmetric BTTB matrices can be stored by using only a single column and can be embedded into a symmetric BCCB matrix, which in turn also only needs a single column. This means that only a single column of the sensitivity matrix needs to be calculated. Once the jth column of the sensitivity matrix represents the influence of the jth source (jth point mass) has on the predicted data, our method only requires a single point mass that set up the equivalent layer to compute the gravity-data forward modeling.

Using the fast Fourier transform it is possible to calculate the eigenvalues of BCCB matrices which can be used to compute a matrix-vector product (gravity-data forward modeling) in a very low computational cost. We have successfully applied the proposed method to upward (or downward) synthetic gravity data. Testing on field data from the Carajás Province, north of Brazil, confirms the potential of our approach in upward-continuing gravity data with 250 000 observations in about 0.2 seconds. Our method allows, in future

research, applying the equivalent layer-technique for processing and interpreting massive data set such as collected in continental and global scales studies.

Figures

Figure 1

Figure 2

Figure 3

Figure 4

Figure 5

Figure 6

Figure 7

Figure 8

Figure 9

Figure 10

Figure 11

Figure 12

Figure 13

Tables

$N \times N$	**Full RAM (Mb)**	**BTTB RAM (Mb)**	**BCCB RAM (Mb)**
100×100	0.0763	0.0000763	0.0006104
400×400	1.22	0.0031	0.0248
$2\,500 \times 2\,500$	48	0.0191	0.1528
$10\,000 \times 10\,000$	763	0.00763	0.6104
$40\,000 \times 40\,000$	12 207	0.305	2.4416
$250\,000 \times 250\,000$	476 837	1.907	15.3
$500\,000 \times 500\,000$	1 907 349	3.815	30.518
$1\,000\,000 \times 1\,000\,000$	7 629 395	7.629	61.035

Table 1: Comparison between the system memory RAM usage needed to store the full matrix, the BTTB single column of the sensitivity matrix and the BCCB eigenvalues (eight times greater than the BTTB singel column). The quantities were computed for different numbers of data (N) with the same corresponding number of equivalent sources (N). This table considers that each element of the matrix is a double-precision number, which requires 8 bytes of storage, except for the BCCB complex eigenvalues, which requires 16 bytes per element.

REFERENCES

Boggs, D., and M. Dransfield, 2004, Analysis of errors in gravity derived from the falcon airborne gravity gradiometer: ASEG-PESA Airborne Gravity 2004 Workshop, Geoscience Australia Record, 135–141.

Chan, R. H., T. F. Chan, and C.-K. Wong, 1999, Cosine transform based preconditioners for total variation deblurring: IEEE transactions on Image Processing, **8**, 1472–1478.

Chan, R. H.-F., and X.-Q. Jin, 2007, An introduction to iterative toeplitz solvers: SIAM, **5**.

Chu, E., and A. George, 1999, Inside the fft black box: serial and parallel fast fourier transform algorithms: CRC Press.

Cordell, L., 1992, A scattered equivalent-source method for interpolation and gridding of potential-field data in three dimensions: GEOPHYSICS, **57**, 629–636.

Dampney, C. N. G., 1969, The equivalent source technique: GEOPHYSICS, **34**, 39–53.

Emilia, D. A., 1973, Equivalent sources used as an analytic base for processing total magnetic field profiles: GEOPHYSICS, **38**, 339–348.

Fairhead, J. D., C. M. Green, and D. Blitzkow, 2003, The use of gps in gravity surveys: The Leading Edge, **22**, 954–959.

Graham, L., D. E. Knuth, and O. Patashnik, 1994, Concrete mathematics: a foundation for computer science, 2 ed.: Addison-Wesley Publishing Company.

Grenander, U., and G. Szegö, 1958, Toeplitz forms and their applications: California Monographs in Mathematical Sciences. University of California Press, Berkeley, CA.

Guspí, F., and I. Novara, 2009, Reduction to the pole and transformations of scattered magnetic data using newtonian equivalent sources: GEOPHYSICS, **74**, L67–L73.

Hackney, R. I., and W. E. Featherstone, 2003, Geodetic versus geophysical perspectives of

the gravity anomaly: Geophysical Journal International, **154**, 35–43.

Hansen, R. O., and Y. Miyazaki, 1984, Continuation of potential fields between arbitrary surfaces: GEOPHYSICS, **49**, 787–795.

Hinze, W. J., C. Aiken, J. Brozena, B. Coakley, D. Dater, G. Flanagan, R. Forsberg, T. Hildenbrand, G. R. Keller, J. Kellogg, R. Kucks, X. Li, A. Mainville, R. Morin, M. Pilkington, D. Plouff, D. Ravat, D. Roman, J. Urrutia-Fucugauchi, M. Véronneau, M. Webring, and D. Winester, 2005, New standards for reducing gravity data: The north american gravity database: GEOPHYSICS, **70**, J25–J32.

Hofmann-Wellenhof, B., and H. Moritz, 2005, Physical geodesy: Springer.

Jin, X.-Q., 2003, Developments and applications of block toeplitz iterative solvers: Springer Science & Business Media, **2**.

Kellogg, O. D., 1929, Foundations of potential theory: Frederick Ungar Publishing Company.

Leão, J. W. D., and J. B. C. Silva, 1989, Discrete linear transformations of potential field data: GEOPHYSICS, **54**, 497–507.

Levinson, N., 1946, The wiener (root mean square) error criterion in filter design and prediction: Journal of Mathematics and Physics, **25**, 261–278.

Li, X., and H.-J. Götze, 2001, Ellipsoid, geoid, gravity, geodesy, and geophysics: Geophysics, **66**, 1660–1668.

Li, Y., and D. W. Oldenburg, 2010, Rapid construction of equivalent sources using wavelets: GEOPHYSICS, **75**, L51–L59.

Lin, F., X. Jin, and S. Lei, 2003, Strang-type preconditioners for solving linear systems from delay differential equations: BIT Numerical Mathematics, **43**, 139–152.

Mendonça, C. A., and J. B. C. Silva, 1994, The equivalent data concept applied to the

interpolation of potential field data: GEOPHYSICS, **59**, 722–732.

Oliveira Jr., V. C., V. C. F. Barbosa, and L. Uieda, 2013, Polynomial equivalent layer: GEOPHYSICS, **78**, G1–G13.

Olkin, J. A., 1986, Linear and nonlinear deconvolution problems: PhD thesis, Rice University.

Silva, J. B. C., 1986, Reduction to the pole as an inverse problem and its application to low-latitude anomalies: GEOPHYSICS, **51**, 369–382.

Siqueira, F. C., V. C. Oliveira Jr, and V. C. Barbosa, 2017, Fast iterative equivalent-layer technique for gravity data processing: A method grounded on excess mass constraint: Geophysics, **82**, G57–G69.

Strang, G., and K. Aarikka, 1986, Introduction to applied mathematics: Wellesley-Cambridge Press Wellesley, MA, **16**.

Trench, W. F., 1964, An algorithm for the inversion of finite toeplitz matrices: Journal of the Society for Industrial and Applied Mathematics, **12**, 515–522.

Vajda, P., A. Ellmann, B. Meurers, P. Vaníček, P. Novák, and R. Tenzer, 2008, Global ellipsoid-referenced topographic, bathymetric and stripping corrections to gravity disturbance: Studia Geophysica et Geodaetica, **52**, 19.

Vajda, P., P. Vaníček, and B. Meurers, 2006, A new physical foundation for anomalous gravity: Studia Geophysica et Geodaetica, **50**, 189–216.

Vajda, P., P. Vaníček, P. Novák, R. Tenzer, and A. Ellmann, 2007, Secondary indirect effects in gravity anomaly data inversion or interpretation: Journal of Geophysical Research: Solid Earth, **112**.

Wray, J., and G. G. Green, 1994, Calculation of the volterra kernels of non-linear dynamic systems using an artificial neural network: Biological Cybernetics, **71**, 187–195.

Zhang, Y., and Y. S. Wong, 2015, Bttb-based numerical schemes for three-dimensional gravity field inversion: Geophysical Journal International, **203**, 243–256.

Zhang, Y., Y. S. Wong, and Y. Lin, 2016, Bttb–rrcg method for downward continuation of potential field data: Journal of applied Geophysics, **126**, 74–86.

LIST OF FIGURES

and standard deviation of 0.0144 mGal.

7 Difference between the fitted gravity data produced by Siqueira et al. (2017) method (Figure 5a) and by our modified form of this method that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25).

8 Difference between the estimated mass distribution within the equivalent layer produced by Siqueira et al. (2017) method (Figure 5a) and by our modified form of this method that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25).

9 The upward-continued gravity data using: (a) the fast equivalent layer proposed by Siqueira et al. (2017) and (b) our modified form of Siqueira et al. (2017) method by using the properties of BTTB and BCCB matrices (equation 25) to calculate the forward modeling. (c) Residuals, defined as the difference between panels a and b with their mean of $-5.938e-18$ and standard deviation of $8.701e-18$. The total computation times in the Siqueira et al. (2017) method and in our approach are 7.62026 and 0.00834 seconds, respectively.

10 The downward-continued gravity data using: (a) the fast equivalent layer proposed by Siqueira et al. (2017) and (b) our modified form of Siqueira et al. (2017) method by using the properties of BTTB and BCCB matrices (equation 25) to calculate the forward modeling. (c) Residuals, defined as the difference between panels a and b with their mean of $5.914e-18$ and standard deviation of $9.014e-18$. The total computation times in the Siqueira et al. (2017) method and in our approach are 7.59654 and 0.00547 seconds, respectively.

11 Carajás Province, Brazil. Gravity data on a regular grid of 500×500 points, totaling $250,000$ observations. The inset shows the study area (blue rectangle) which covers

the southeast part of the state of Pará, north of Brazil.

12 Carajás Province, Brazil. (a) Predicted gravity data produced by our modification of the fast equivalent-layer method (Siqueira et al., 2017) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25). (b) Gravity residuals, defined as the difference between the observed data in Figure 11 and the predicted data in panel a, with their mean of 0.000292 mGal and standard deviation of 0.105 mGal.

13 Carajás Province, Brazil. The upward-continued gravity data using our modification of the fast equivalent layer method (Siqueira et al., 2017) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25). The total computation time is 0.216 seconds for processing of the $250,000$ observations.

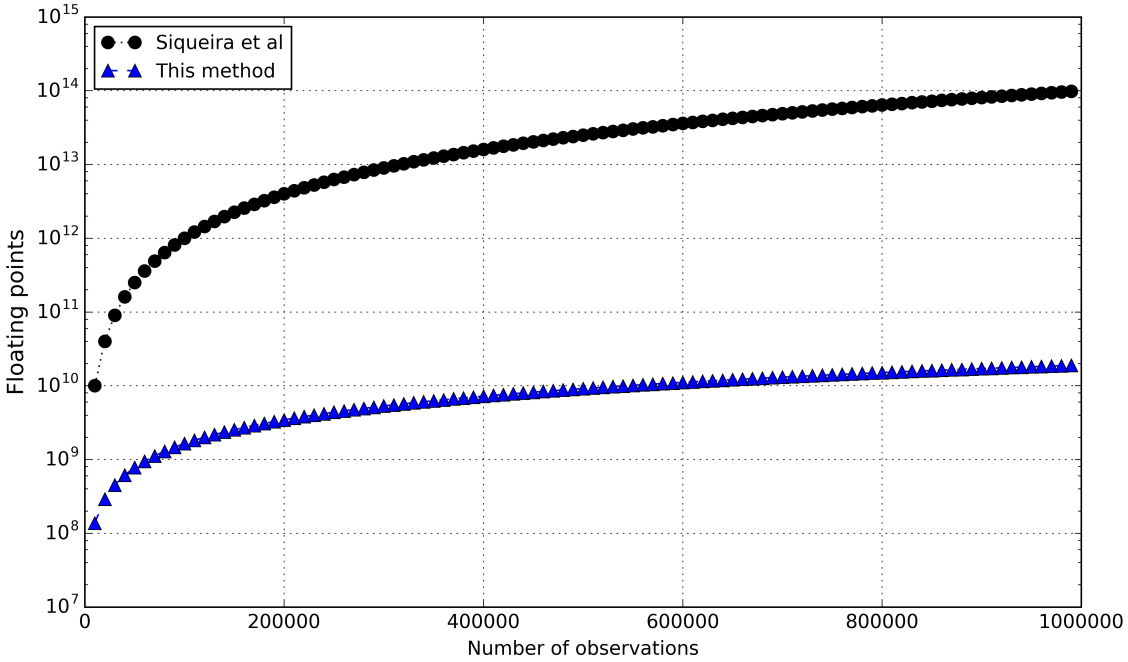

Figure 1: floating points to estimate the parameter vector using the fast equivalent layer using Siqueira et al. (2017) method (equation 39) and our approach (equation 40) versus the numbers of observation points varyig from $N = 5\,000$ to $N = 1\,000\,000$ with 50 iterations. The number of operations is drastically decreased.

– **GEO-XXXX**

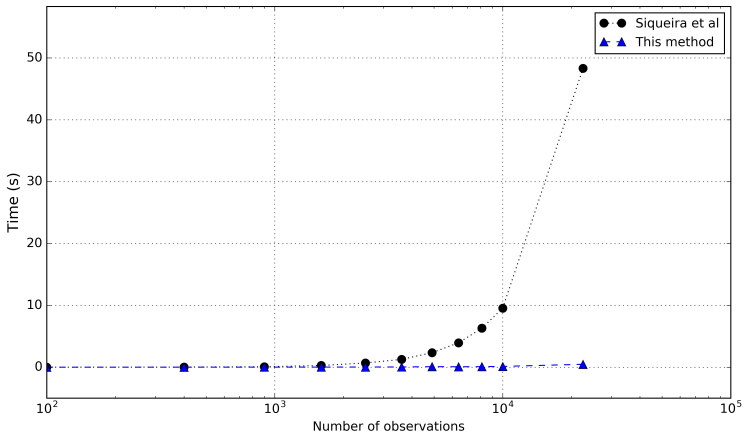

Figure 2: time necessary to run 50 iterations of the Siqueira et al. (2017) method and the one presented in this work. With the limitation of 16 Gb of memory RAM in our system, we could test only up to 22 500 obervation points.

– **GEO-XXXX**

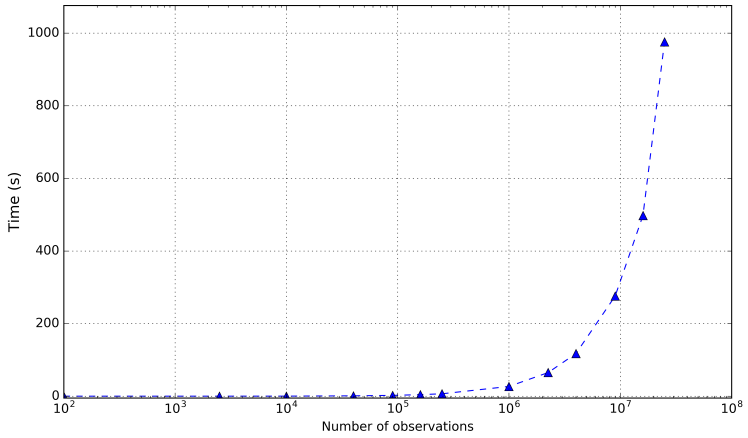

Figure 3: time necessary to run the equivalent layer technique with 50 iterations using our approach, where the RAM is not a limitation factor. We could run up to 25 million observation points. In comparison, 1 million observation points took 26.8 seconds to run, where the maximun 22 500 observation points in Figure 2, with Siqueira et al. (2017) method, took 48:3 seconds.

– **GEO-XXXX**

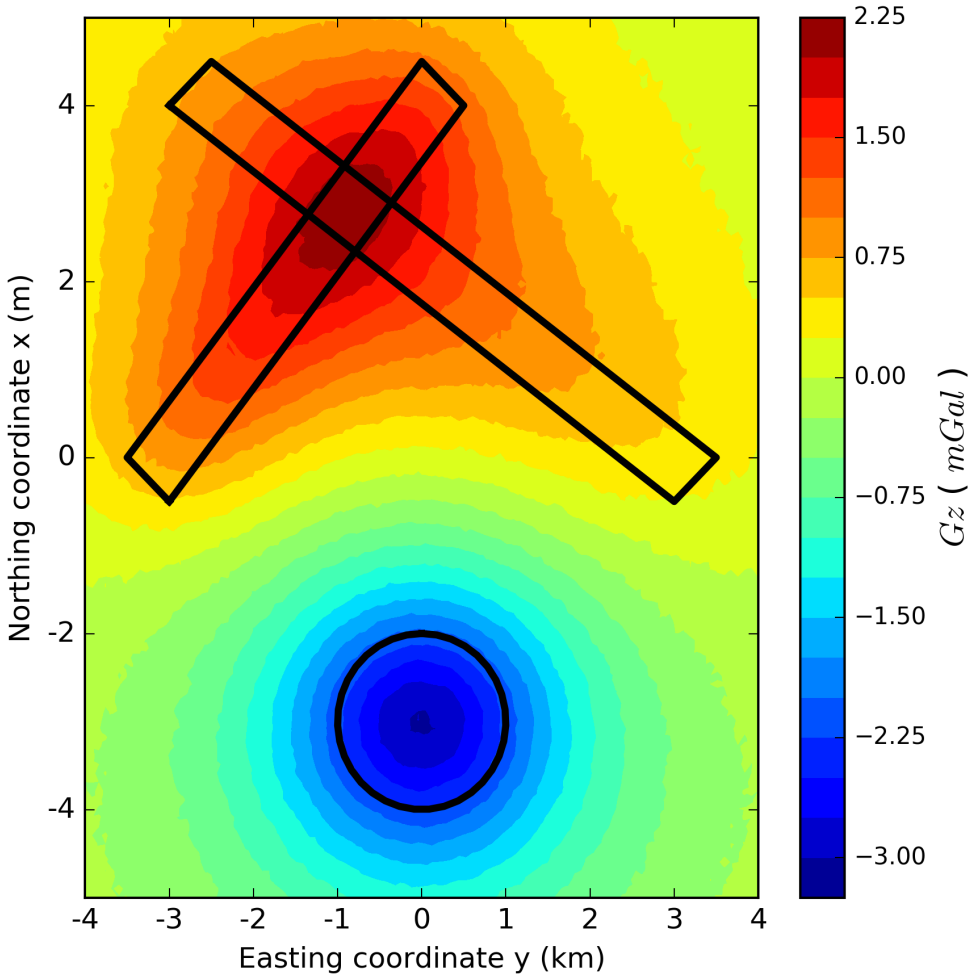

Figure 4: Noise-corrupted gravity data (in color map) produced by three simulated sources whose horizontal projections are shown in black lines. The simulated sources are: two polygonal prisms, with density contrast of $0.35\,\mathrm{g/cm^3}$ (upper-left body) and $0.4\,\mathrm{g/cm^3}$ (upper-right body), and a sphere with radius of 1 000 m with density contrast of $-0.5\,\mathrm{g/cm^3}$.

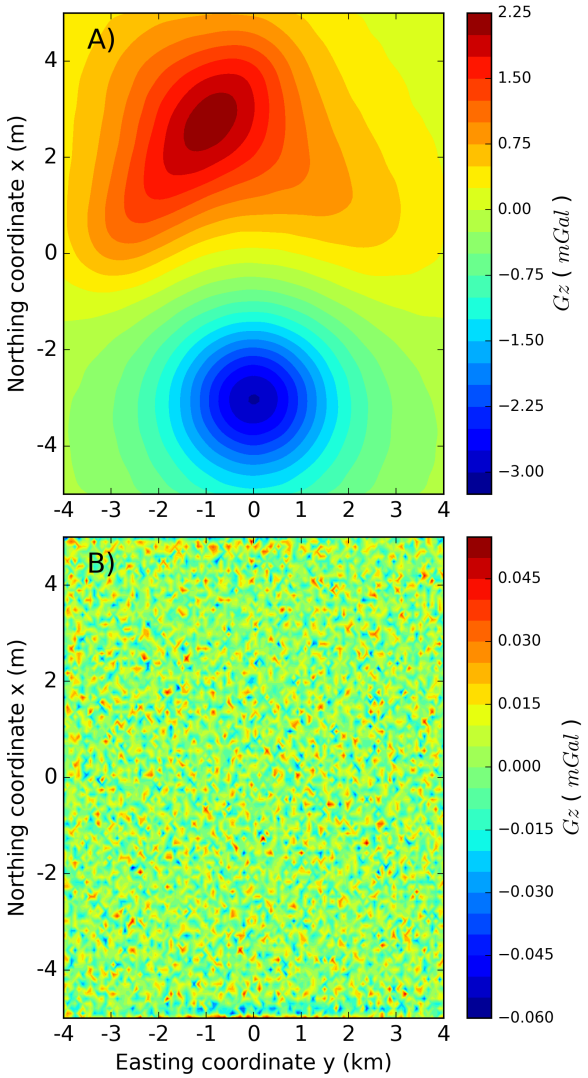

Figure 5: (a) Fitted gravity data produced by the fast equivalent-layer technique proposed by Siqueira et al. (2017). (b) Gravity residuals, defined as the difference between the observed data in Figure 4 and the predicted data in panel a, with their mean of $8.264e-7$ and standard deviation of 0.0144 mGal.

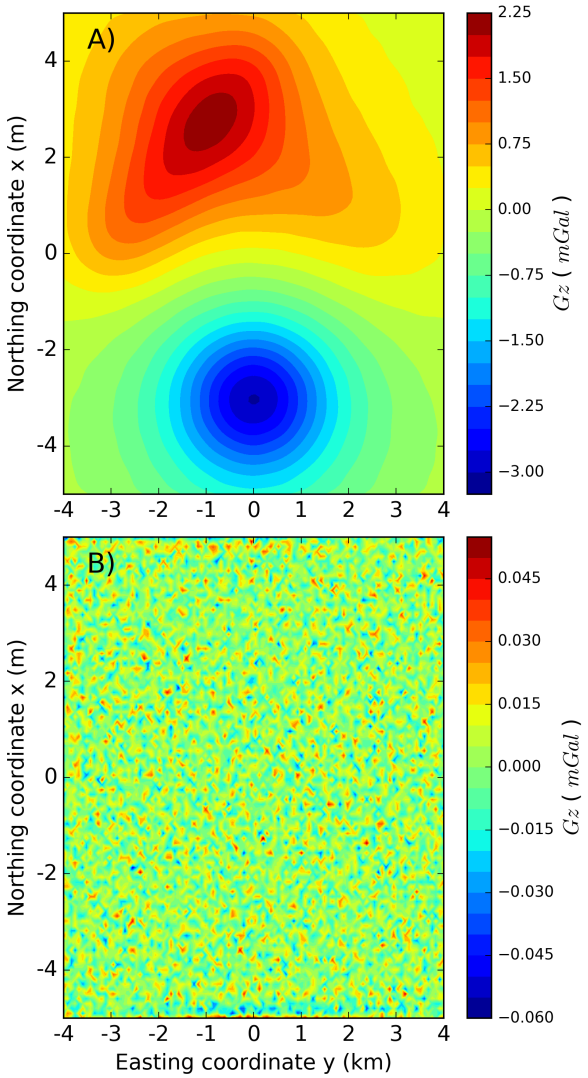

Figure 6: (a) Fitted gravity data produced by our modification of the fast equivalent layer proposed by Siqueira et al. (2017). (b) Gravity residuals, defined as the difference between the observed data in Figure 4 and the predicted data in panel a, with their mean of $8.264e{-}7$ and standard deviation of 0.0144 mGal.

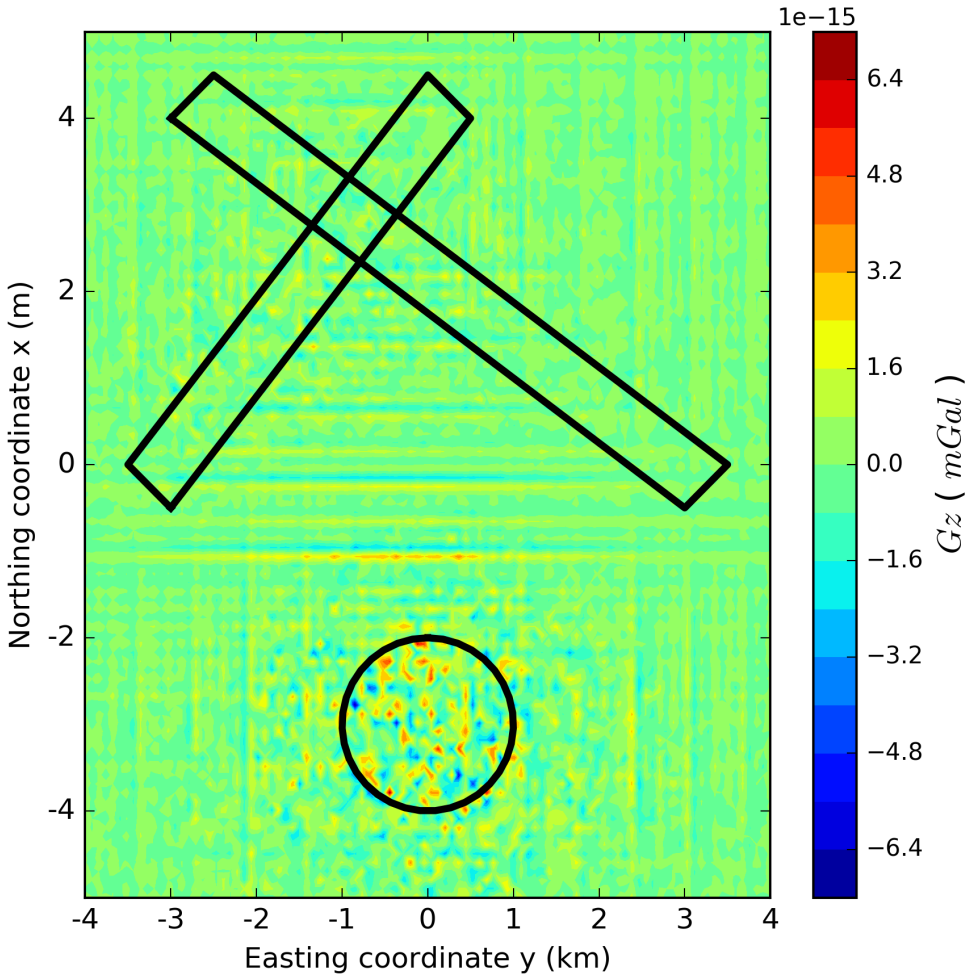

Figure 7: Difference between the fitted gravity data produced by Siqueira et al. (2017) method (Figure 5a) and by our modified form of this method that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25).

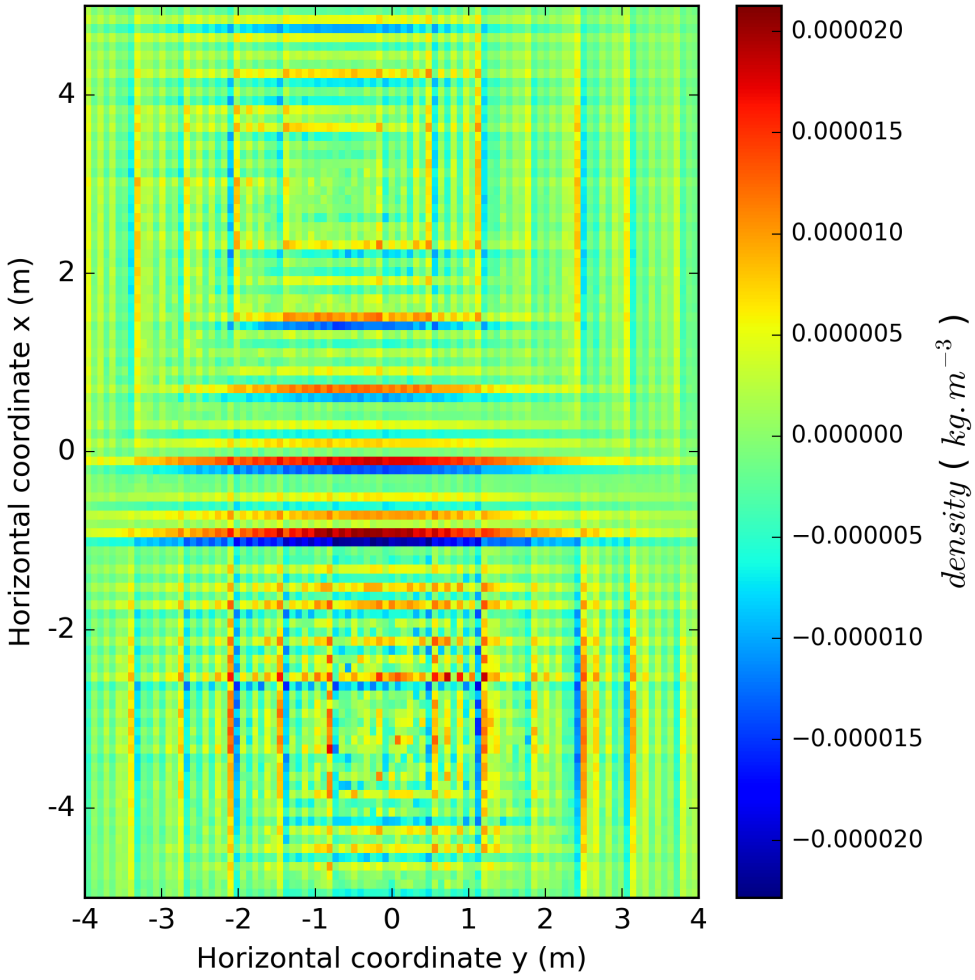

Figure 8: Difference between the estimated mass distribution within the equivalent layer produced by Siqueira et al. (2017) method (Figure 5a) and by our modified form of this method that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25).

– **GEO-XXXX**

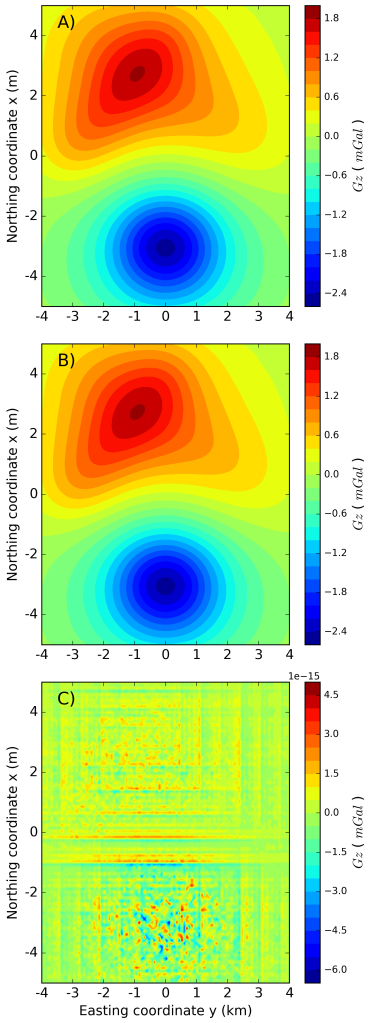

Figure 9: The upward-continued gravity data using: (a) the fast equivalent layer proposed by Siqueira et al. (2017) and (b) our modified form of Siqueira et al. (2017) method by using the properties of BTTB and BCCB matrices (equation 25) to calculate the forward modeling. (c) Residuals, defined as the difference between panels a and b with their mean of $-5.938e-18$ and standard deviation of $8.701e-18$. The total computation times in the Siqueira et al. (2017) method and in our approach are 7.62026 and 0.00834 seconds, respectively.

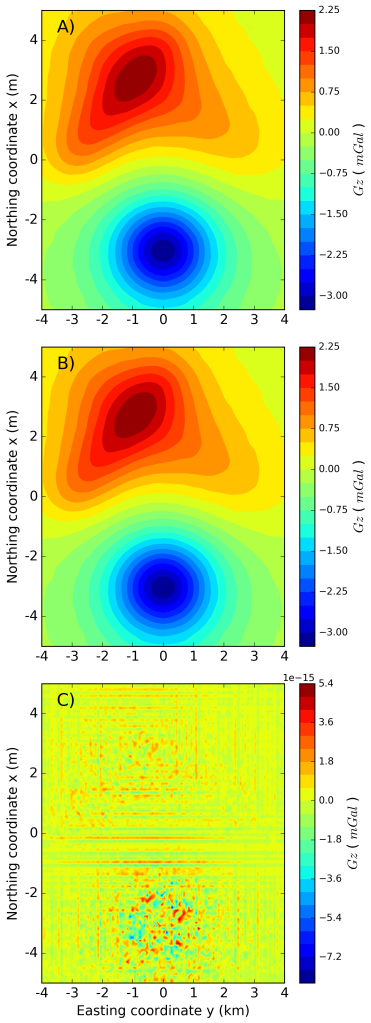

Figure 10: The downward-continued gravity data using: (a) the fast equivalent layer proposed by Siqueira et al. (2017) and (b) our modified form of Siqueira et al. (2017) method by using the properties of BTTB and BCCB matrices (equation 25) to calculate the forward modeling. (c) Residuals, defined as the difference between panels a and b with their mean of $5.914e-18$ and standard deviation of $9.014e-18$. The total computation times in the Siqueira et al. (2017) method and in our approach are 7.59654 and 0.00547 seconds, respectively.

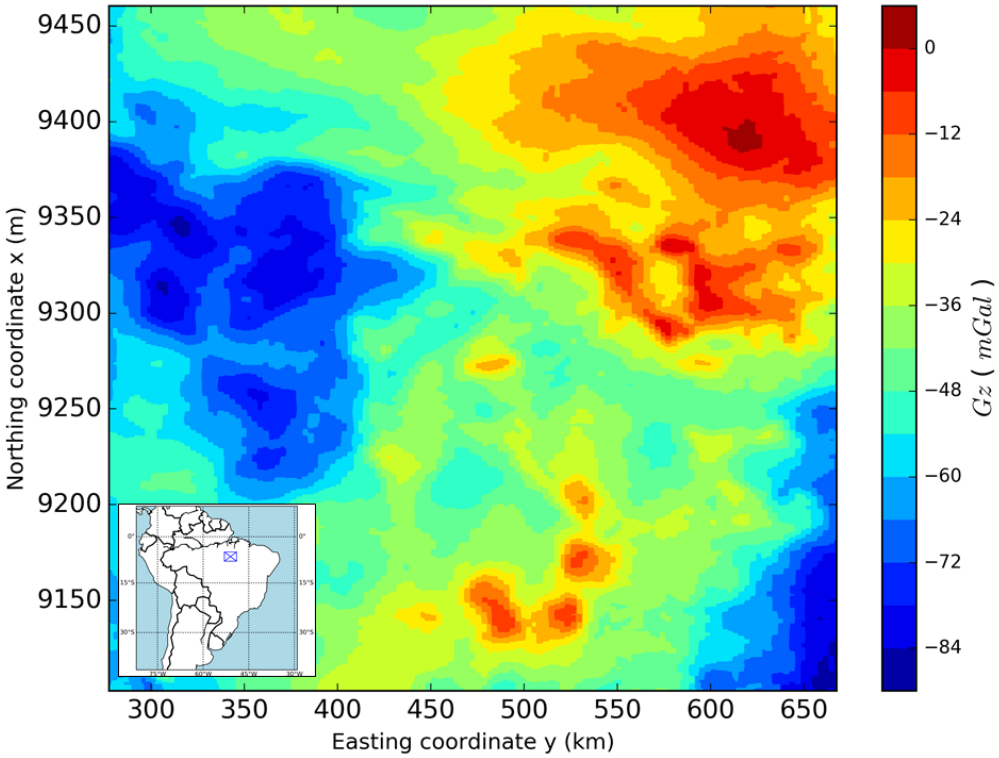

Figure 11: Carajás Province, Brazil. Gravity data on a regular grid of 500×500 points, totaling $250,000$ observations. The inset shows the study area (blue rectangle) which covers the southeast part of the state of Pará, north of Brazil.

– **GEO-XXXX**

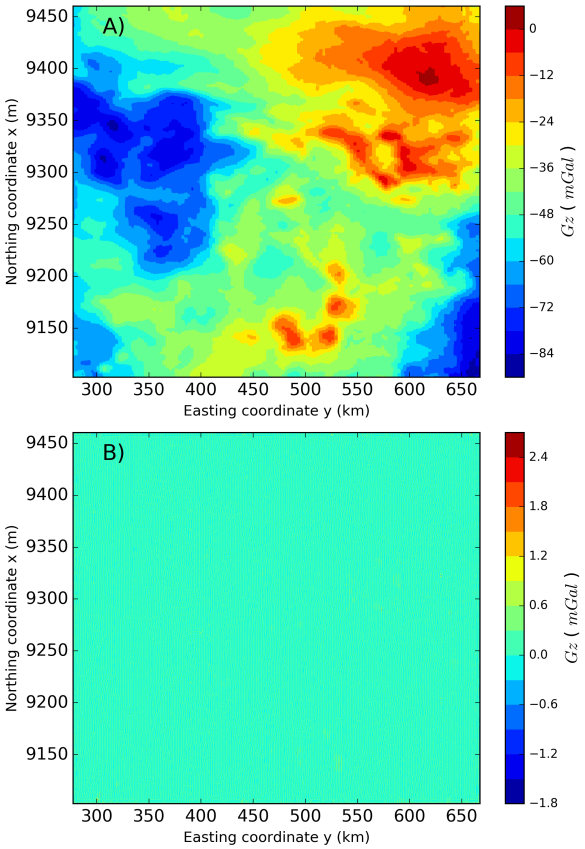

Figure 12: Carajás Province, Brazil. (a) Predicted gravity data produced by our modification of the fast equivalent-layer method (Siqueira et al., 2017) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25). (b) Gravity residuals, defined as the difference between the observed data in Figure 11 and the predicted data in panel a, with their mean of 0.000292 mGal and standard deviation of 0.105 mGal.

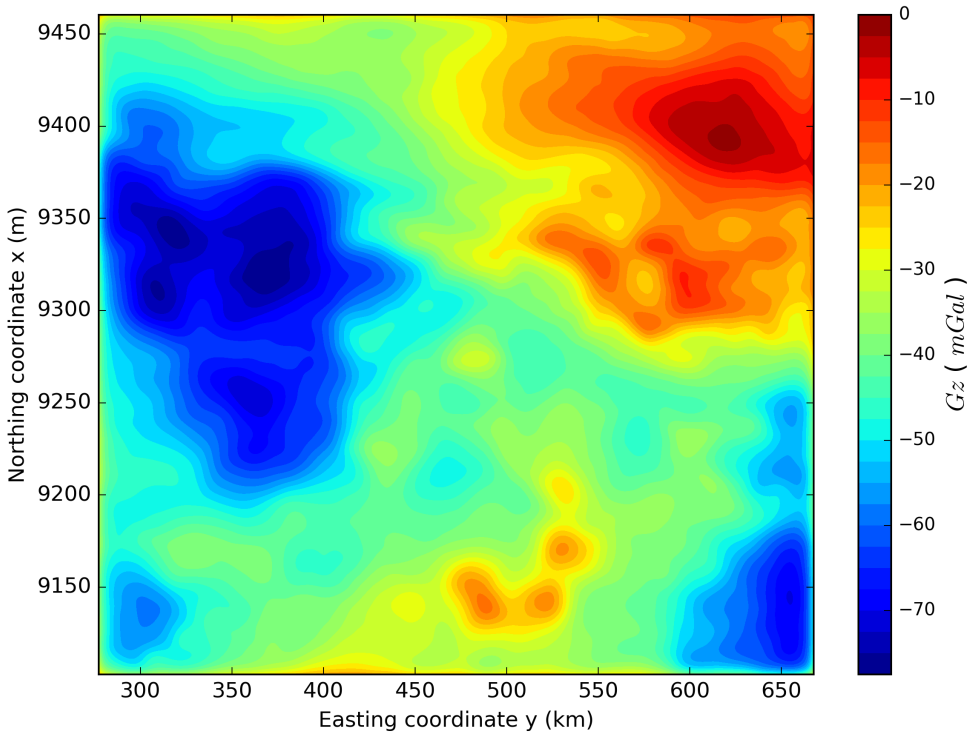

Figure 13: Carajás Province, Brazil. The upward-continued gravity data using our modification of the fast equivalent layer method (Siqueira et al., 2017) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 25). The total computation time is 0.216 seconds for processing of the $250,000$ observations.

– **GEO-XXXX**