Fast equivalent layer technique for gravity data processing with Block-Toeplitz Toeplitz-Block matrix systems

Diego Takahashi[*], *Vanderlei C. Oliveira Jr.*[*] *and Valéria C. F. Barbosa*[*]

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 properties related to the symmetric Block-Toeplitz Toeplitz-block (BTTB) and Block-Circulant Circulant-Block (BCCB) matrices, 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 row, which in turn can be embedded into a symmetric BCCB matrix. Likewise, only the first row 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. 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 pratical 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 observation 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 Siqueira et al.'s (2017) method 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 Siqueira et al.'s (2017) method.

We propose the use of BTTB and BCCB matrices properties to efficiently solve the forward modeling in Siqueira et al.'s (2017) method, 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 theory for gravity data

In applied geophysics, the observed gravity disturbance $\delta g(x, y, z)$ (Heiskanen and Moritz, 1967) is commomly approximated as the vertical component z of the gravitational attraction produced by gravity sources, as follows:

$$\delta g(x, y, z) = c_g \, G \int\int_{v}\int \rho(x', y', z') \frac{(z' - z) \quad dx'dy'dz'}{[(x - x')^2 + (y - y')^2 + (z - z')^2]^{\frac{3}{2}}} \, , \tag{1}$$

where $c_g = 10^5$ is a constant transforming from m/s^2 to $mGal$, G is the Newton's gravitational constant $(m^3/kg\, s^2)$ and $\rho(x', y', z')$ is the density at the point (x', y', z') inside the volume V of the source. This integral is defined in a Cartesian coordinate system with axis x pointing to north, y pointing to east and z pointing downward.

According to the classic upward continuation integral (Henderson, 1960, 1970), it is possible to compute the gravity disturbance $\delta g(x_i, y_i, z_h)$, at a point (x_i, y_i, z_h), from the gravity disturbance $\delta g(x_j, y_j, z_0)$ at the constant plane z_0:

$$\delta g(x_i, y_i, z_h) = \frac{1}{2\pi} \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \frac{\delta g(x_j, y_j, z_0)(z_0 - z_h)}{[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_h - z_0)^2]^{\frac{3}{2}}} \, dx \, dy, \tag{2}$$

where, $z_0 > z_h$. Equation 2 shows that the gravity disturbance $\delta g(x_i, y_i, z_h)$ is a convolution between $\delta g(x_j, y_j, z_0)$ and another harmonic function on the horizontal plane $z = z_0$. Multiplying and dividing equation 2 by G and discretizing the integral, we obtain:

$$\delta g(x_i, y_i, z_h) = \sum_{j=1}^{N} p_j a_{ij} \, , \quad i, j = 1, \ldots, N, \tag{3}$$

where p_j is the coefficient representing the physical property of the jth equivalent source:

$$p_j = \frac{\Delta S_j \delta g(x_j, y_j, z_0)}{2\pi c_g G} \, , \tag{4}$$

with Δs being an element of area located at the vertical coordinate z_h and centered at the horizontal coordinates (x_i, y_i), and a_{ij} is given by:

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

In this case, the harmonic function a_{ij} represents the vertical component of the gravitational attraction exerted, at the point (x_i, y_i, z_h), by a point mass located at the point (x_j, y_j, z_0), with mass p_j.

Equation 3 can be expressed in matrix form as:

$$\mathbf{d}(\mathbf{p}) = \mathbf{A}\mathbf{p} \, , \tag{6}$$

where $\mathbf{d}(\mathbf{p})$ is an $N \times 1$ vector, whose ith element is the predicted gravity disturbance $\delta g(x_i, y_i, z_h)$, $\mathbf{p}$ is an $N \times 1$ parameter vector, whose jth element is the coefficient p_j (equation 4) representing the physical property of the jth equivalent source and $\mathbf{A}$ is an $N \times N$ sensibility matrix, where each element a_{ij} is given by equation 5 and defines the vertical component of gravity field produced by the jth equivalent source with unitary density at the observation point (x_i, y_i, z_h). The solution of the parameters $\mathbf{p}$ can be solved by a linear inversion minimizing the function:

$$\Psi(\mathbf{p}) = \theta_g(\mathbf{p}) + \mu \, \theta_m(\mathbf{p}) \, , \tag{7}$$

where $\theta_g(\mathbf{p})$ is the misfit data function given by:

$$\theta_g(\mathbf{p}) = ||\mathbf{d}^o - \mathbf{d}(\mathbf{p})||_2^2 \, , \tag{8}$$

which is the Euclidian norm of the residual defined as the difference between the observed data $\mathbf{d}^o$ and the predicted data $\mathbf{d}(\mathbf{p})$.

The function $\theta_m(\mathbf{p})$ is the regularization function, for example, the zeroth-order Tikhonov (Tikhonov and Arsenin, 1977):

$$\theta_m(\mathbf{p}) = ||\mathbf{p}||_2^2 \, , \tag{9}$$

which is the Euclidian norm of the parameters $\mathbf{p}$. The variable μ in equation 7 is a real positive number called regularizing parameter.

Derivating equation 7 in relation to $\mathbf{p}$ and making the result equal to zero, it is possible to estimate the parameters as:

$$\hat{\mathbf{p}} = \left(\mathbf{A}^\top \mathbf{A} + \mu \, \mathbf{I} \right)^{-1} \mathbf{A}^\top \mathbf{d}^o \, . \tag{10}$$

Fast equivalent layer technique

In Siqueira et al. (2017), the authors developed an iterative least-squares method to estimate the mass distribution over the equivalent layer based on the excess of mass and the positive correlation between the observed gravity data and the masses on equivalent sources. This scheme is proven to have a better computational efficiency than the classical equivalent layer approach with data sets of at least 200 observation points, even using a large number of iterations. This work also proves that the excess of mass of a body is proportional to the surface integration of the gravity data on the horizontal plane z_0 where the equivalent sources are distributed. Considering each equivalent source directly beneath the observation points, a initial approximation of mass distibution is given by:

$$\mathbf{p}^0 = \tilde{\mathbf{A}}^{-1} \mathbf{d}^o \, , \tag{11}$$

where $\mathbf{d}^o$ is the gravity data and $\tilde{\mathbf{A}}^{-1}$ is an $N \times N$ diagonal matrix whose elements are:

$$\tilde{\mathbf{A}}^{-1} = \frac{\Delta s}{(2\pi \, G \, c_g)} \, . \tag{12}$$

At each kth iteration, a mass correction vector $\mathbf{\Delta p^k}$ for every source is calculated by minimizing the function:

$$\phi(\mathbf{\Delta p^k}) = ||\mathbf{d}^o - \mathbf{A}\hat{\mathbf{p}}^k - \tilde{\mathbf{A}}\widehat{\mathbf{\Delta p}^k}||_2^2 \tag{13}$$

and the mass distribution of the equivalent sources is updated as:

$$\mathbf{\Delta}\hat{\mathbf{p}}^{k+1} = \hat{\mathbf{p}}^k + \mathbf{\Delta}\hat{\mathbf{p}}^k \, . \tag{14}$$

At the kth iteration of Siqueira et al.'s (2017) method, the matrix-vector product $\mathbf{A}\hat{\mathbf{p}}^k$ must be calculated to get a new residual $\mathbf{d^0} - \mathbf{A}\hat{\mathbf{p}}^k$. While it is true that for a small number of observation points this represents no computational effort, for very large data sets it is costful and can be overwhelming in terms of memory RAM to maintain such operation.

Structure of sensibility matrix A

If a gravity disturbance data $\delta g(x_i, y_i, z_h)$ (equation 3) $i = 1, \ldots, N$, is placed on a regular grid at the constant vertical coordinate $z_h = z_1$ and there is a set of equivalent sources each one located directly below each point of this grid at a constant depth z_0, the elements a_{ij} (equation 5) can be rewritten as follows:

$$a_{ij} = c_g G \frac{\Delta z}{\left[(r_{ij})^2 + (\Delta z)^2\right]^{\frac{3}{2}}} \, , \tag{15}$$

where $\Delta z = z_0 - z_1$ and $r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ represents the relative horizontal distance between the ith observation point and the jth equivalent source. It is worth noting that: (i) $r_{ij} = r_{ji}$ for any pair ij and (ii) r_{ij} computed for different pairs ij may assume the same value. As a consequence of these two properties, the matrix $\mathbf{A}$ (equation 6) has the structure of a symmetric Toeplitz by blocks, where each block is also a symmetric Toeplitz matrix (BTTB) (Chan and Jin, 2007; Golub and Loan, 2013). For example, consider a regular grid of $N_x \times N_y$ points, where $N_x = 4$ and $N_y = 3$, comprising a total number of $N = N_x \times N_y = 12$ points. In this case, matrix $\mathbf{A}$ is a partitioned matrix by $N_x \times N_x$ blocks:

$$\mathbf{A} = \begin{pmatrix} \mathbf{A_0} & \mathbf{A_1} & \mathbf{A_2} & \mathbf{A_3} \\ \mathbf{A_1} & \mathbf{A_0} & \mathbf{A_1} & \mathbf{A_2} \\ \mathbf{A_2} & \mathbf{A_1} & \mathbf{A_0} & \mathbf{A_1} \\ \mathbf{A_3} & \mathbf{A_2} & \mathbf{A_1} & \mathbf{A_0} \end{pmatrix} , \tag{16}$$

where each block is a $N_y \times N_y$ matrix given by:

$$\mathbf{A}_0 = \begin{pmatrix} a_0^\ell & a_1^\ell & a_2^\ell \\ a_1^\ell & a_0^\ell & a_1^\ell \\ a_2^\ell & a_1^\ell & a_0^\ell \end{pmatrix} , \tag{17}$$

where the elements a_k^l, $l = 0, \dots, N_x - 1$, $k = 0, \dots, N_y - 1$ are computed with equation 15. Here, the most striking feature is that the complete BTTB matrix $\mathbf{A}$ can be obtained by computing only its first row or column.

BCCB 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 optmizing 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 16), its transformation into a BCCB matrix $\mathbf{C}$ is given by:

$$\mathbf{C} = \begin{pmatrix} \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} \\ \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} \\ \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} \\ \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} \\ \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} \\ \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} & \mathbf{C_2} \\ \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} & \mathbf{C_1} \\ \mathbf{C_1} & \mathbf{C_2} & \mathbf{C_3} & \mathbf{0} & \mathbf{C_3} & \mathbf{C_2} & \mathbf{C_1} & \mathbf{C_0} \end{pmatrix}, \tag{18}$$

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

$$\mathbf{C}_l = \begin{pmatrix} \mathbf{A}_l & \times \\ \times & \mathbf{A}_l \end{pmatrix} = \begin{pmatrix} a_0^\ell & a_1^\ell & a_2^\ell & 0^\ell & a_2^\ell & a_1^\ell \\ a_1^\ell & a_0^\ell & a_1^\ell & a_2^\ell & 0^\ell & a_2^\ell \\ a_2^\ell & a_1^\ell & a_0^\ell & a_1^\ell & a_2^\ell & 0^\ell \\ 0^\ell & a_2^\ell & a_1^\ell & a_0^\ell & a_1^\ell & a_2^\ell \\ a_2^\ell & 0^\ell & a_2^\ell & a_0^\ell & a_0^\ell & a_1^\ell \\ a_1^\ell & a_2^\ell & 0^\ell & a_2^\ell & a_1^\ell & a_0^\ell \end{pmatrix}. \tag{19}$$

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 6), we now carry the following multiplication:

$$\mathbf{Cv} = \mathbf{q}\,, \tag{20}$$

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

$$\mathbf{v} = \begin{pmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{N_x-1} \\ \mathbf{0}_{(2N_xN_y)} \end{pmatrix} \tag{21}$$

and

$$\mathbf{q} = \begin{pmatrix} \mathbf{q}_0 \\ \mathbf{q}_1 \\ \vdots \\ \mathbf{q}_{N_x-1} \\ \dagger_{(2N_xN_y)} \end{pmatrix} , \tag{22}$$

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 of "garbage" (ACHAR UM JEITO MELHOR DE DESCREVER, CHAN E ZHANG DESCREVEM APENAS COMO "A PARTE A SER JOGADA FORA") and $\mathbf{v}_\ell$ and $\mathbf{q}_\ell$, $\ell = 0, \ldots, N_x - 1$ are:

$$\mathbf{v}_\ell = \begin{pmatrix} \mathbf{p}_\ell \\ \mathbf{0}_{(N_y)} \end{pmatrix} \tag{23}$$

and

$$\mathbf{q}_\ell = \begin{pmatrix} \mathbf{d}_\ell \\ \dagger_{(N_y)} \end{pmatrix} , \tag{24}$$

where $\mathbf{0}_{(N_y)}$ is a $N_y \times 1$ vector of zeros, $\dagger_{(N_y)}$ is a $N_y \times 1$ vector of "garbage" (ACHAR UM JEITO MELHOR DE DESCREVER, CHAN E ZHANG DESCREVEM APENAS COMO "A PARTE A SER JOGADA FORA") 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 6).

As demonstrated by Grenander and Szegö (1958), 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 Chan and Jin (2007) demonstrate that they satisfy:

$$\mathbf{C} = \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right)^* \mathbf{\Lambda} \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) , \tag{25}$$

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 25 it is possible to calculate the eigenvalues of the BCCB matrix using only its first column:

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{C} = \mathbf{\Lambda} \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \tag{26}$$

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{C}\, \mathbf{t}_1 = \mathbf{\Lambda} \left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \mathbf{t}_1 \tag{27}$$

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \, \mathbf{c}_0 = \mathbf{\Lambda} \, \frac{1}{\sqrt{4N_xN_y}} \mathbf{1}_{(4N_xN_y)} \tag{28}$$

$$\left(\mathbf{F}_{(2N_y)} \otimes \mathbf{F}_{(2N_x)}\right) \, \mathbf{c}_0 = \frac{1}{\sqrt{4N_xN_y}} \lambda \quad , \tag{29}$$

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 29 can be calculated using the 2D-DFT (Jain, 1989):

$$\mathbf{F}_{(2N_x)} \mathbf{G} \, \mathbf{F}_{(2N_y)} \quad = \tfrac{1}{\sqrt{4N_xN_y}} \mathbf{L} \quad , \tag{30}$$

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 25 in equation 20, using the fast calculation of the eigenvalues of BCCB matrices (equation 30) and using the same previous Kronecker product property, the auxiliary matrix-vetor product $\mathbf{Cv} = \mathbf{q}$ can be rewritten as follows:

$$\mathbf{F}^*_{(2N_x)} \left[\mathbf{L} \circ \left(\mathbf{F}_{(2N_x)} \mathbf{V} \, \mathbf{F}_{(2N_y)}\right)\right] \mathbf{F}^*_{(2N_y)} = \mathbf{Q} \, , \tag{31}$$

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 18) is given by:

$$[\mathbf{C}]_{(0)} = \begin{pmatrix} \mathbf{C}_0 \\ \mathbf{C}_1 \\ \vdots \\ \mathbf{C}_{N_x-2} \\ \mathbf{C}_{N_x-1} \\ \mathbf{0} \\ \mathbf{C}_{N_x-1} \\ \mathbf{C}_{N_x-2} \\ \vdots \\ \mathbf{C}_1 \end{pmatrix} \quad , \tag{32}$$

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:

$$\mathbf{c}_0 = \begin{pmatrix} a_{00} \\ a_{10} \\ \vdots \\ a_{(N_y-2)0} \\ a_{N_y-1)0} \\ 0 \\ a_{N_y-1)0} \\ a_{(N_y-2)0} \\ \vdots \\ a_{10} \end{pmatrix} . \tag{33}$$

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 6) 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:

$$f_0 = N^{it}(3N + 2N^2) \; , \tag{34}$$

where N^{it} is the number of iterations. From equation 34 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 $N \log_2(N)$ flops (Brigham and Brigham, 1988). Computing the eigenvalues of the BCCB matrix ($4N \times 4N$) and applying 2D-FFT on the parameter vector (equation 31), takes $4N \log(4N)$ each. The point-multiplication takes $4N$. As it is necessary to compute the inverse FFT another two $4N \log(4N)$ must be taken in account. However, the sensibility matrix does not change during the process, thus, the eigenvalues of BCCB must be calculated only once, outside of the iteration. This lead us to a flops count in our method of:

$$f_1 = 4N \log(4N) + N^{it}(7N + 8N \log(4N)). \tag{35}$$

Another major improvement of this methodology is the exoneration of calculating the full sensibility matrix $\mathbf{A}$ (equation 6). Each element needs 12 flops (equation 5), totalizing $12N^2$ flops for the full matrix. Calculating only the first row of the BTTB matrix $12N$ flops is required. Thus, the full flops count of Siqueira et al.'s (2017) method:

$$f_s = 12N^2 + N^{it}(3N + 2N^2), \tag{36}$$

it is decreased in our method to:

$$f_f = 12N + 4N\log(4N) + N^{it}(7N + 8N\log(4N)). \tag{37}$$

Figure 1 shows the floating points to estimate the parameter vector using the fast equivalent layer with Siqueira et al.'s (2017) method (equation 34) and our approach (equation 35) versus the number of observation points varyig from $N = 5000$ to $N = 1000000$ 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 BTTB first row and the BCCB eigenvalues (8 times the BTTB first row). 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.'s (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.'s (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 20) to solve, at each iteration, the forward modeling (the matrix-vector product $\mathbf{A}\hat{\mathbf{p}}^k$) required in the fast equivalent layer 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 $0.35g/cm^3$ (upper-left prism) and $0.4g/cm^3$ (upper-right prism) and a sphere with radius of 1,000 m with density contrast of $-0.5g/cm^3$. 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 0.01486 mGal.

The advantage of using the structures of BTTB and BCCB matrices to compute forward modeling in the fast equivalent layer proposed by 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 using Siqueira et al.'s (2017) method and by our modified form of this method that computes the forward modeling using equation 20. 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.'s (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.'s (2017) method and our modification of this method. These results (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 20) yields practically the same result as the one produced by computing this matrix-vector product with a full matrix $\mathbf{A}$.

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 300m and the downward is at 50m. 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.'s (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 1500 times faster than Siqueira et al.'s (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 north–south direction with flight line spacing of 3 km and tie lines along east–west 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 20), we set an equivalent layer at 300 m deep. Figure 12a shows the fitted gravity data after with 50 iterations by applying our method approach. 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 (0.000292 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 5000 m over the real data. The upward-continued gravity data (Figure 13) seem a reasonable processing because of the attenuatinon of the short-wave lenghts. 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) and Block-Circulant Circulant-Block (BCCB) matrices 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 is grounded on 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 as the vertical component of gravity, that depends on the inverse of distance between the observation and the point mass over the equivalent layer. Symmetric BTTB matrices can be stored by its only first row and can be embedded into a symmetric BCCB matrix, which in turn also only needs its first row.

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. The time needed to process medium-sized grids of observation, for example 22,500 points, is reduced in 102 times. 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

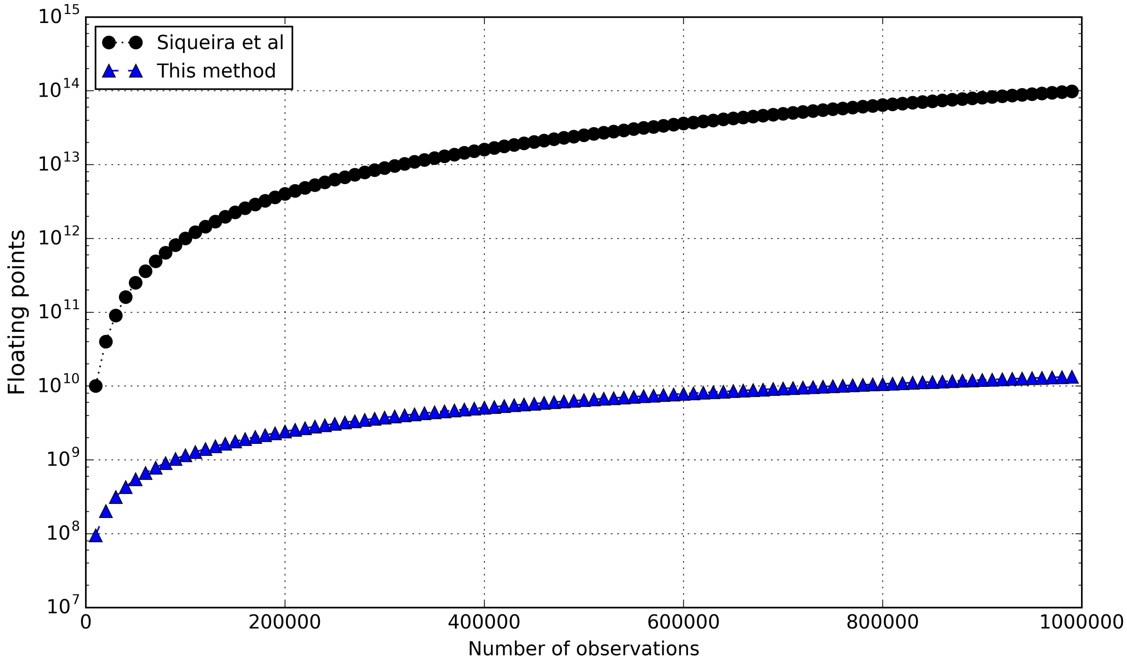

Figure 1: floating points to estimate the parameter vector using the fast equivalent layer with Siqueira et al.'s (2017) method (equation 34) and our approach (equation 35) versus the numbers of observation points varyig from $N = 5000$ to $N = 1000000$ with 50 iterations. The number of operations is drastically decreased.

Figure 2

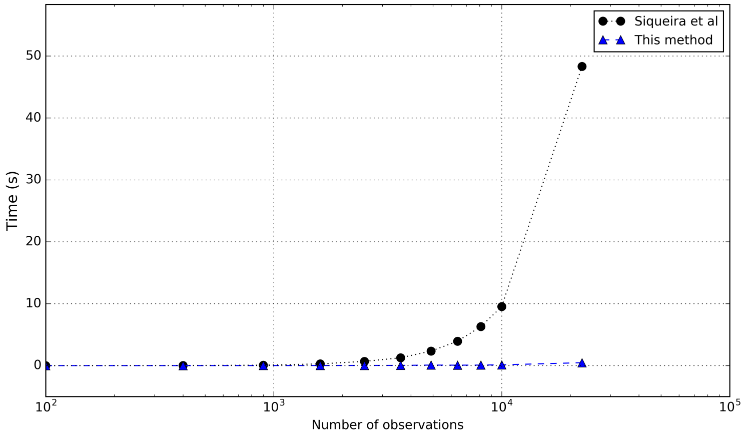

Figure 2: time necessary to run 50 iterations of the Siqueira et al.'s (2017) method and the one presented in this work. With the limitation of 16 Gb of memory RAM in our system, we could test only up to 22500 obervation points.

Figure 3

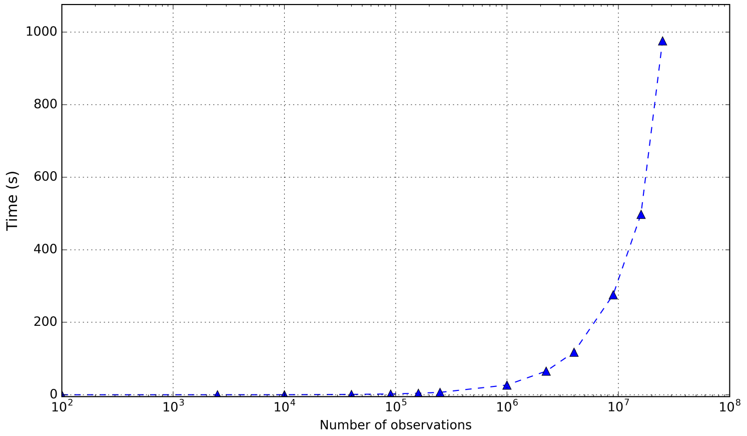

Figure 3: time necessary to run the equivalent layer technique with 50 iterations using only this new approach, where the RAM is not a limitation factor. We could run up to 25 million observation points. In comparison, 1 million observation points took 26.8 seconds to run, where the maximun 22500 observation points in figure 2, with Siqueira et al.'s (2017) method, took 48.3 seconds.

Figure 4

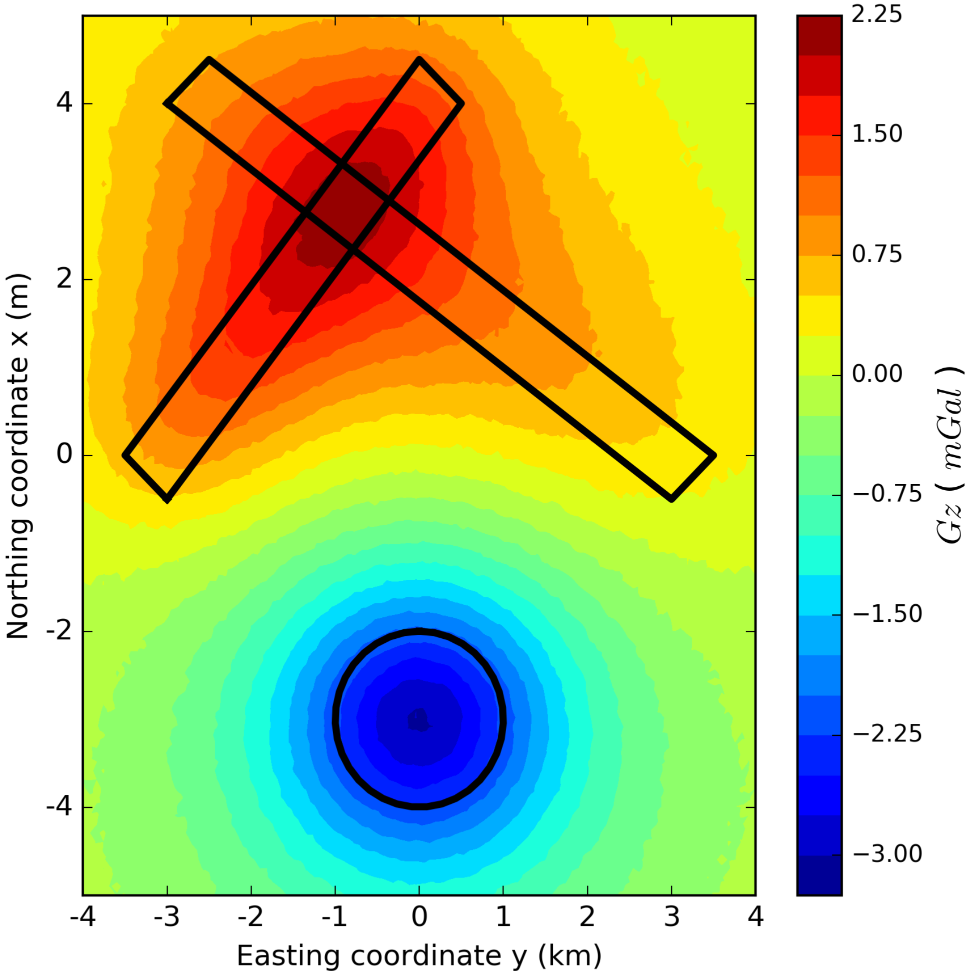

Figure 4: model with two polygonal prisms, with density contrast of 0.35 (upper-left body) and $0.4 g/cm^3$ (upper-right body), and a sphere with radius of $1000m$ with density contrast of $-0.5 g/cm^3$. The vertical component of gravity generated by this bodies were calculated and are shown together with their horizontal projections. A gaussian noise was added to the data with mean of zero and maximum value of 0.5% of the maximum of the original data. As previous said only in regular grids the BTTB matrix structures appears. We created 10000 observation points regularly spaced in a grid of 100×100, with a uniform $100m$ of height for all the observations.

Figure 5

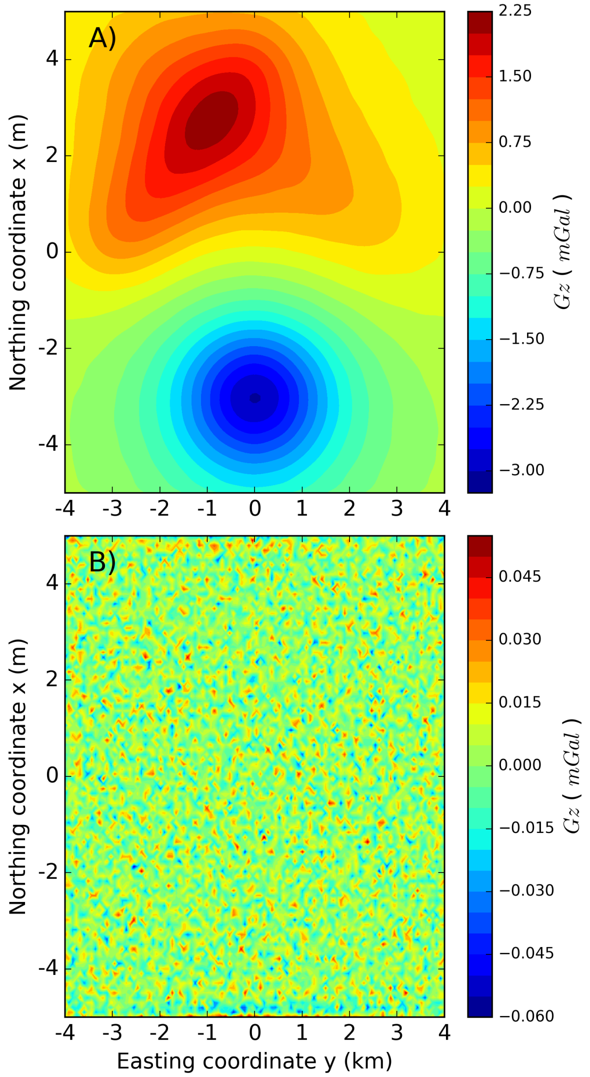

Figure 5: (a) Fitted gravity data produced by the fast equivalent layer proposed by Siqueira et al. (2017). (b) Gravity residuals, defined as the difference between the observed data in Figure 4 and the predicted data in (a), with their mean of $8.264e-7$ and standard deviation of 0.0144 mGal.

Figure 6

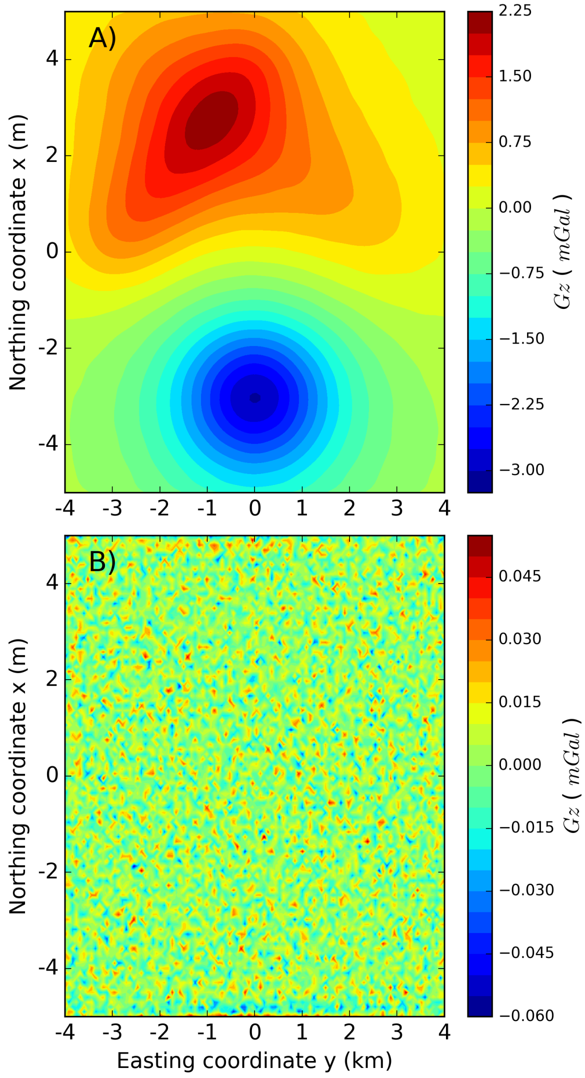

Figure 6: (a) Fitted gravity data produced by our modification of the fast equivalent layer proposed by Siqueira et al. (2017). (b) Gravity residuals, defined as the difference between the observed data in Figure 4 and the predicted data in (a), with their mean of $8.264e-7$ and standard deviation of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

Figure 7

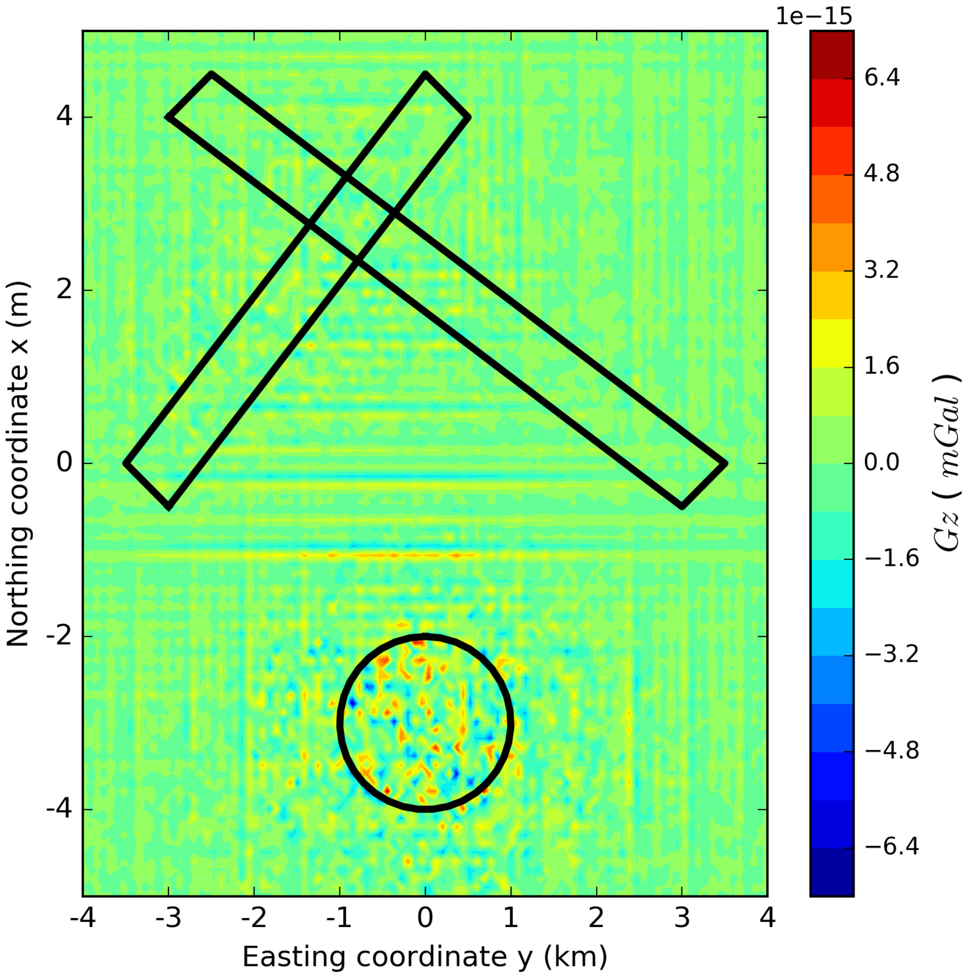

Figure 7: Difference between the fitted gravity data produced by Siqueira et al.'s (2017) method (Figure 5a) and by our modified form of this method that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 20).

Figure 8

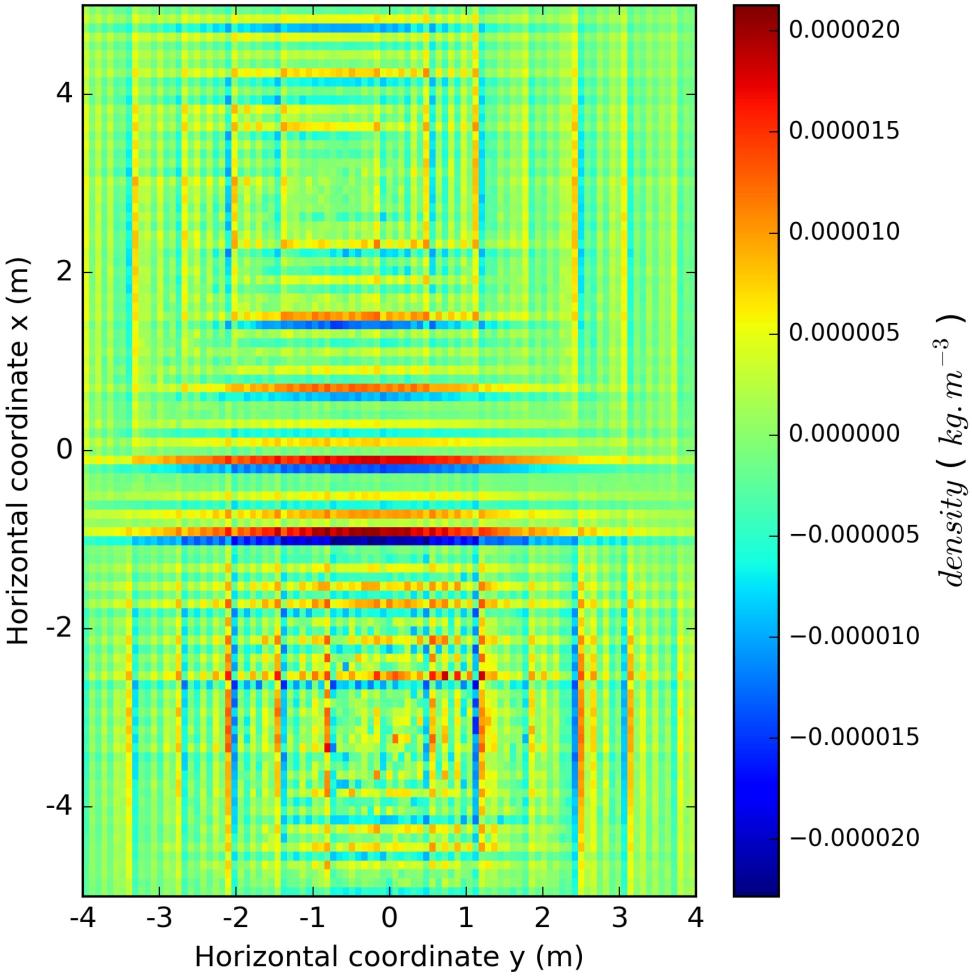

Figure 8: Difference between the estimated mass distribution within the equivalent layer produced by Siqueira et al.'s (2017) method (Figure 5a) and by our modified form of this method that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 20).

Figure 9

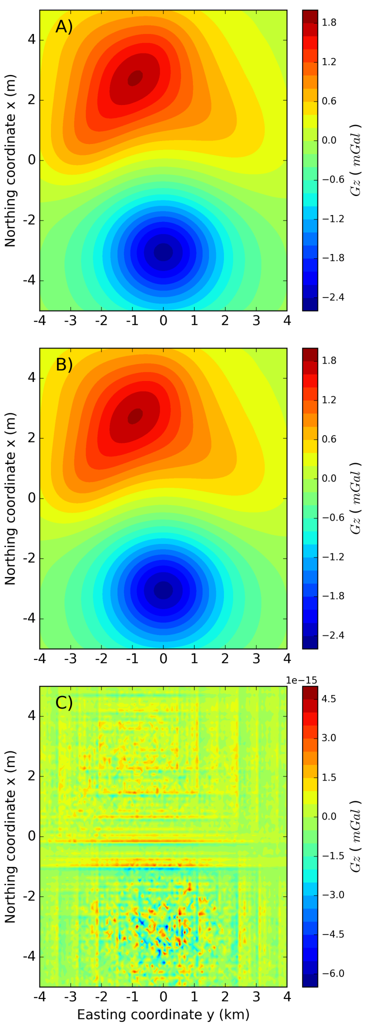

Figure 9: The upward-continued gravity data using: (a) the fast equivalent layer proposed by Siqueira et al. (2017) and (b) our modified form of Siqueira et al.'s (2017) method by using the properties of BTTB and BCCB matrices (equation 20) to calculate the forward modeling. (c) Residuals, defined as the difference between 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.'s (2017) method and in our approach are 7.62026 and 0.00834 seconds, respectively.

Figure 10

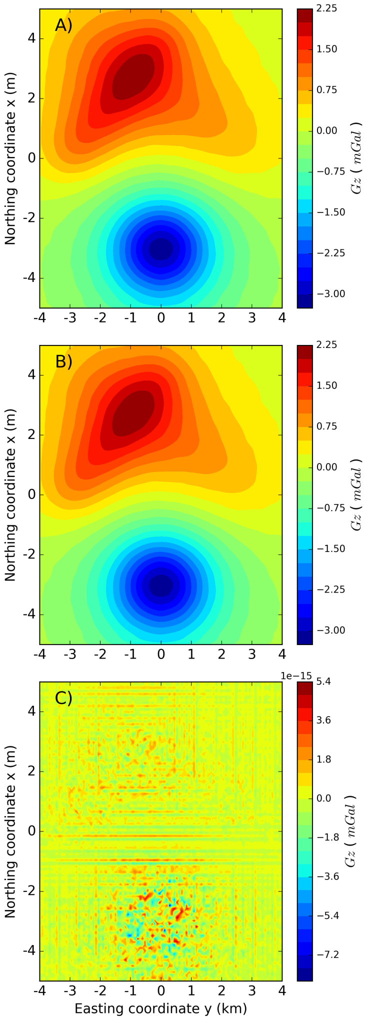

Figure 10: The downward-continued gravity data using: (a) the fast equivalent layer proposed by Siqueira et al. (2017) and (b) our modified form of Siqueira et al.'s (2017) method by using the properties of BTTB and BCCB matrices (equation 20) to calculate the forward modeling. (c) Residuals, defined as the difference between 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.'s (2017) method and in our approach are 7.59654 and 0.00547 seconds, respectively.

Figure 11

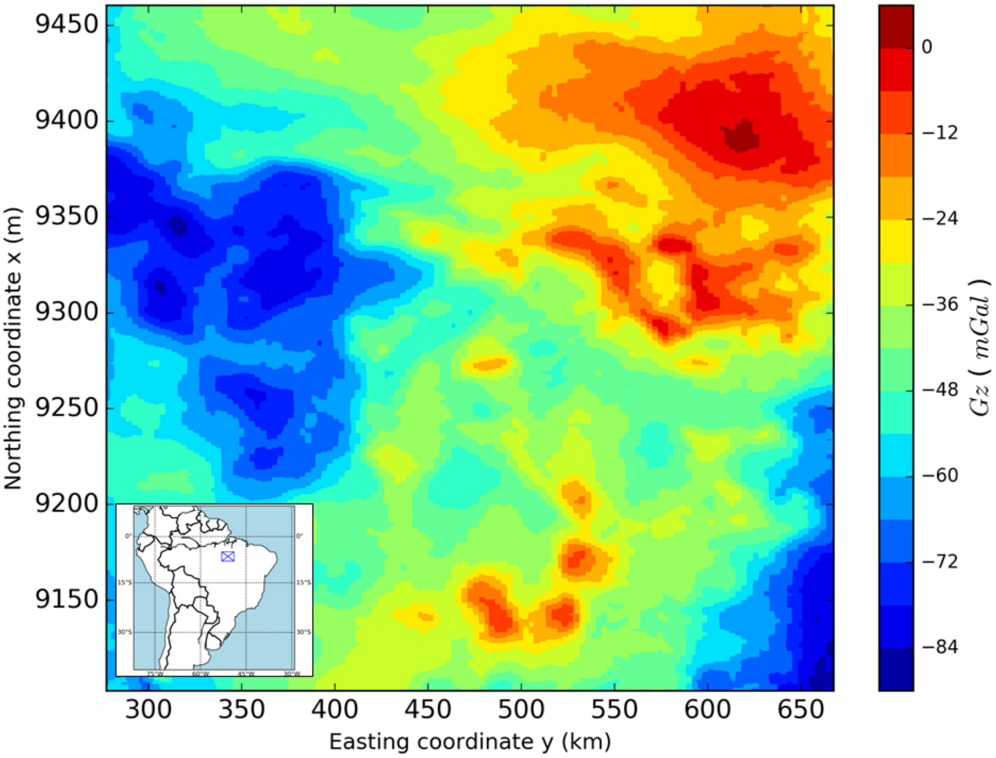

Figure 11: Carajás Province, Brazil. Gravity data on a regular grid of 500×500 points, totaling $250,000$ observations. The inset shows the study area (blue rectangle) which covers the southeast part of the state of Pará, north of Brazil.

Figure 12

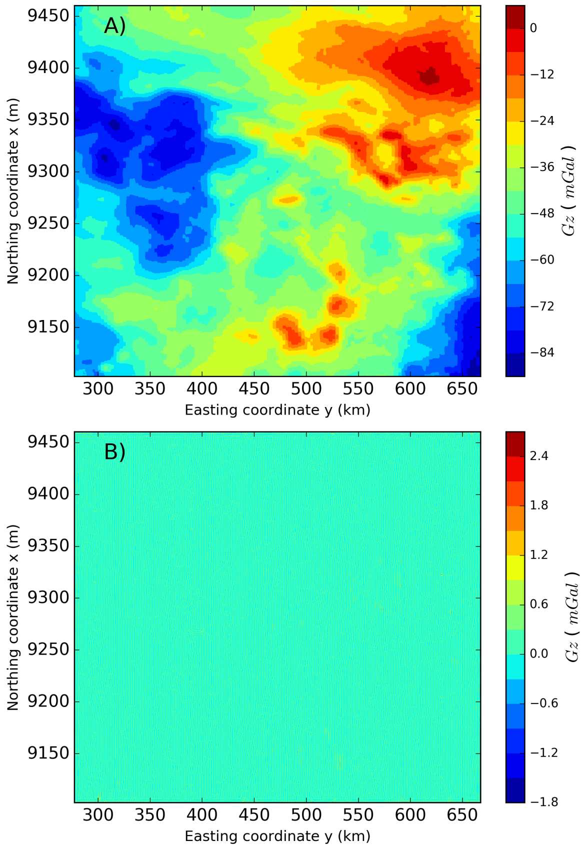

Figure 12: Carajás Province, Brazil. (a) Predicted gravity data produced by our modification of the fast equivalent layer method (Siqueira et al. (2017)) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 20). (b) Gravity residuals, defined as the difference between the observed data in Figure 11 and the predicted data in a, with their mean of 0.000292 mGal and standard deviation of 0.105 mGal.

Figure 13

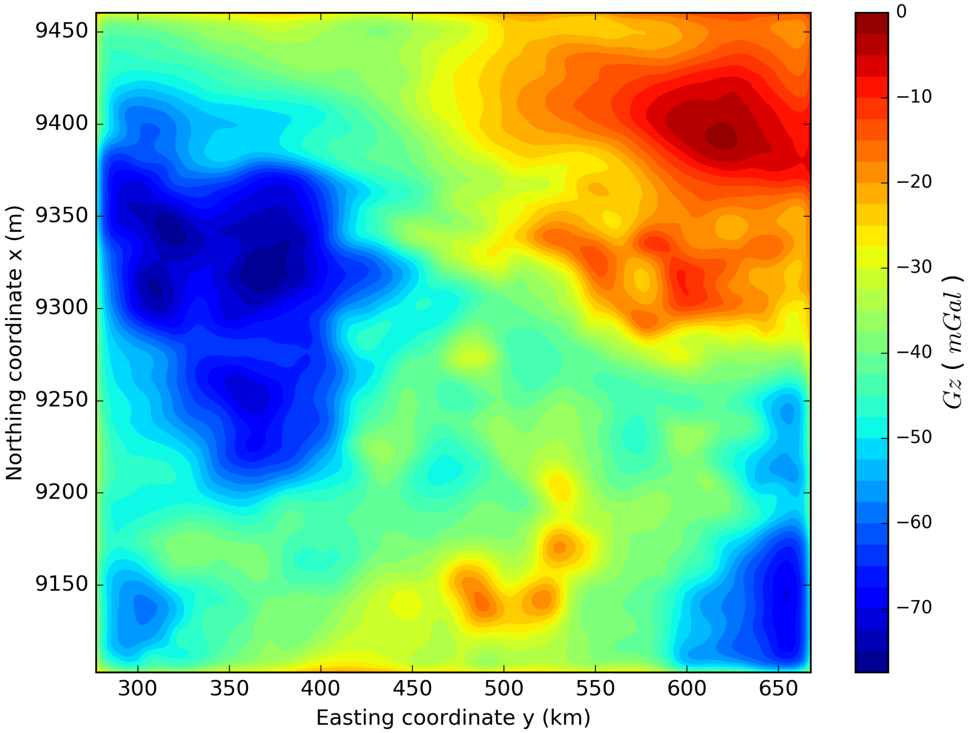

Figure 13: Carajás Province, Brazil. The upward-continued gravity data using our modification of the fast equivalent layer method (Siqueira et al. (2017)) that computes the forward modeling using the properties of BTTB and BCCB matrices (equation 20). The total computation time is 0.216 seconds for processing of the $250,000$ observations.

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
2500 × 2500	48	0.0191	0.1528
10000 × 10000	763	0.00763	0.6104
40000 × 40000	12207	0.305	2.4416
250000 × 250000	476837	1.907	15.3
500000 × 500000	1907349	3.815	30.518
1000000 × 1000000	7629395	7.629	61.035

Table 1: Comparison between the system memory RAM usage needed to store the full matrix, the BTTB first row and the BCCB eigenvalues (eight times the BTTB). 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.

ACKNOWLEDGMENTS

This study was financed by the brazilian agencies CAPES (in the form of a scholarship), FAPERJ (grant n.° E-26 202.729/2018) and CNPq (grant n.° 308945/2017-4).

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.

Brigham, E. O., and E. O. Brigham, 1988, The fast fourier transform and its applications: prentice Hall Englewood Cliffs, NJ, **448**.

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**.

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., 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.

Golub, G. H., and C. F. V. Loan, 2013, Matrix computations (johns hopkins studies in the mathematical sciences), 4 ed.: Johns Hopkins University Press.

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.

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

Heiskanen, W. A., and H. Moritz, 1967, Physical geodesy: Bulletin Géodésique (1946-1975), **86**, 491–492.

Henderson, R. G., 1960, A comprehensive system of automatic computation in magnetic and gravity interpretation: Geophysics, **25**, 569–585.

———, 1970, On the validity of the use of the upward continuation integral for total magnetic intensity data: Geophysics, **35**, 916–919.

Jain, A. K., 1989, Fundamentals of digital image processing: Englewood Cliffs, NJ: Prentice Hall,.

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, 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**.

Tikhonov, A. N., and V. Y. Arsenin, 1977, Solutions of ill-posed problems: V. H. Winston & Sons.

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.

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.