# Convolutional equivalent layer for magnetic data processing

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

[†]*Observatório Nacional, Department of Geophysics, Rio de Janeiro, Brazil*

[*] *Corresponding author: diego.takahashi@gmail.com*

(September 3, 2021)

**GEO-XXXX**

Running head: **Magnetic convolutional equivalent layer**

## ABSTRACT

A fast equivalent layer for magnetic data processing is presented. Taking advantage of the sensitivity matrix structure of the magnetic kernel, when observation and equivalent sources are aligned on a regular spaced grid with constant height, it is possible to calculate the matrix-vector product in a very fast manner. The structure is called block-Toeplitz Toeplitz-block (BTTB) and this type of matrix is well known in literature to be embbeded in a block-Circulant Circulant-block (BCCB), wich in turn can have its eigenvalues calculated using only the first column and a 2D fast Fourier transform. We show that, despite this BTTB matrix is not symmetric, by using only the first equivalent source it is possible to calculate all the first columns of the inverse of distance second derivatives that composes the magnetic kernel and reconstruct the first column of the BCCB matrix, saving computational time and system memory. The conjugate gradient iterative method is used to solve the linear system and estimate the physical properties distributed over the equivalent layer. Synthetic tests show a decrease in the order of $10^4$ in floating-point operations, 25 times in computation runtime with a mid-size $80 \times 80$ grid, and exponential decrease in memory RAM usage, allowing to perform this operation with millions of observation points on desktop computers. Synthetic tests with irregular grids also show that this method can work with directional disturbances under certain limits. A real magnetic data of Carajás Province, Brazil, with $1,310,000$ observation points in an irregular grid was used to successfully perform a data processing with this method, taking 385.56 seconds to estimate the physical property and 2.64 seconds for the upward-continuation.

# INTRODUCTION

Large-scale data processing with tens of thousands of data, is a reality in all areas of geophysics including the geophysical potential fields. The potential-field data processing includes convolution integrals which can be solved either in the space or Fourier domains. The earliest techniques of potential-field data processing were developed in the space domain. For example, Peters (1949) accomplished, in the space domain, the second and fourth derivatives of magnetic data and the upward- and downward-continuations of magnetic data by deriving coefficients that are used in a graphical convolution with the magnetic data. Howerver, the techniques for processing potential-field data in space-domain were soon substituted by the Fourier-domain techniques. Dean (1958) pointed out that the operations of second derivative, analytic continuation, smoothing, the removing of residuals or regionals, and others for processing potential-field data are similar to the electric filter circuits in Fourier domain. Dean (1958) was the first to develop the theroy of linear filter in Fourier domain for gravity and magnetic processing and to present filters in Fourier domain (Dean, 1958, see Table I, p 113) for some theoretical geophysical operations (e.g., derivatives and upward and downward continuations). Gunn (1975) presented a comprehensive analysis of processing potential-field data in Fourier domain.

An approach for processing potential-field data in space domain is the equivalent-layer technique. The equations deductions of the equivalent layer as a solution of the Laplace's equation in the region above the source was first presented by Kellogg (1929) and detailed explanations can also be found in Blakely (1996). Although the equivalent-layer technique has been known since the 1960s in geophysical literature (Danes, 1961; Bott, 1967; Dampney, 1969), its use has become feasible only recently because the advances in computational power. In magnetic data procesing, some authors explored this technique for calculating the first and second vertical derivatives fields (Emilia, 1973), reduction to the pole (Silva, 1986; Oliveira Jr. et al., 2013; Li et al., 2014), upward/downward continuations (Hansen and Miyazaki, 1984; Li and Oldenburg, 2010) and total magnetic induction vector components calculation (Sun et al., 2019).

Together with the rise in computational processing power, some works tried new implementations to increase the efficiency of the equivalent layer. In Leão and Silva (1989) the authors used a shifting window over the layer, increasing the number of linear systems to be solved, but the size of each linear system is reduced. Another approach for a fast equivalent layer was proposed by Li and Oldenburg (2010) who transformed the full sensitivity matrix into a sparse one by using the compression of the coefficient matrix using fast wavelet transforms based on orthonormal, compactly supported wavelets. Oliveira Jr. et al. (2013) divided the equivalent layer into a grid of fixed source windows. Instead of directly calculating the the physical-property distribution of a finite set of equivalent sources (e.g., dipoles, point of masses) arranged in the entire equivalent layer, Oliveira Jr. et al. (2013) estimated the coefficients of a bivariate polynomial function describing the physical-property distribution within each equivalent-source window. The estimated polynomial-coefficients are transformed into the physical-property distribution and thus any standard linear transformation of the data can be performed. Grounded on excess mass constraint, Siqueira et al. (2017) proposed an iterative method for processing large gravimetric data using the equivalent layer without requiring the solution of a linear system. In Siqueira et al. (2017), the initial mass distribution over the equivalent layer is proportional to observed gravity data and it is updated at each iteration by adding mass corrections that are proportional to the residuals of observed and estimated data.

One of the greatest obstacles to the use of the equivalent-layer technique for processing potential-field data is the solution of the associated linear system. A wide variety of applications in mathematics and engineering 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 iterative method of conjugate gradient is used in most cases, in Chan and Jin (2007) the authors presented an introduction on the topic for 1D data structures of Toeplitz matrices and also for 2D data structures, which they called block-Toeplitz Toeplitz-block matrices. In both cases, the solving strategy is to embbed the Toeplitz/BTTB matrix into a Circulant/Block-Circulant Circulant-Block matrix, calculate its eigenvalues by a 1D or 2D fast Fourier transform of its first column, respectively and carry the matrix-vector product between kernel and parameters at each iteration of the conjugate gradient method in a very fast manner.

In potential field methods, the properties of Toeplitz system have been used for downward continuation (Zhang et al., 2016) and for 3D gravity-data inversion using a 2D multilayer model (Zhang and Wong, 2015). More recently, Hogue et al. (2020) provided an overview on modeling the gravity and magnetic kernels using the BTTB structures and Renaut et al. (2020) used BTTB the structures for inversion of both gravity and magnetic data to recover sparse subsurface structures. Takahashi et al. (2020) combined the fast equivalent source technique presented by Siqueira et al. (2017) with the concept of symmetric block-Toeplitz Toeplitz-block (BTTB) matrices to introduce the convolutional equivalent layer for gravimetric data technique. Takahashi et al. (2020) showed that the BTTB structure appears when the sensitivity matrix of the linear system, required to solve the gravimetric equivalent layer, is calculated on a regular spaced grid of dataset with constant height and each equivalent source is exactly beneath each observed data point. This work showed an decrease in the order of $10^4$ in floating-point operations needed to estimate the equivalent sources; thus, the Takahashi et al. (2020) method was able to efficiently process very large gravity data sets. Moreover, Takahashi et al. (2020) method yielded neither significant boundary effects nor noise amplification.

In this work, the convolutional equivalent layer using the block-Toeplitz Toeplitz-block idea, presented in Takahashi et al. (2020), will be used to solve the linear system required to estimate the physical property that produces a magnetic field on regular grids. Here, we achive very fast solutions using a conjugate gradient algorithm combined with the fast Fourier transform. We present a novel method of exploring the symmetric structures of the second derivatives of the inverse of the distance contained in the magnetic kernel, to keep the memory RAM usage to the minimal by using only one equivalent source to carry the calculations of the forward problem. We also show tests of the magnetic convolutional equivalent layer when irregular grids are used. The convergence of the conjugate gradient maintains in an acceptable level even using irregular grids. Our results show the good performance of our method in producing fast and robust solutions for processing large amounts of magnetic data using the equivalent layer technique.

## METHODOLOGY

### Classical equivalent layer for magnetic data

Let $\mathbf{d}^o$ be the $N \times 1$ observed data vector, whose $i$th element is the total-field anomaly $d_i^o$ produced by arbitrarily magnetized sources at the position $(x_i, y_i, z_i)$, $i = 1, \ldots, N$, of a right-handed Cartesian coordinate system with $x$-, $y$- and $z$-axis pointing to north, east and down, respectively. We consider that the total-field anomaly data $d_i^o$ represent the discrete values of a harmonic function. Besides, we consider that the main geomagnetic field direction at the study area can be defined by the unit vector

$$\hat{\mathbf{F}} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} \cos(I_0)\,\cos(D_0) \\ \cos(I_0)\,\sin(D_0) \\ \sin(I_0) \end{bmatrix} , \tag{1}$$

with constant inclination $I_0$ and declination $D_0$. In this case, $d_i^o$ can be approximated by the predicted total-field anomaly

$$\Delta T_i = \sum_{j=1}^{M} p_j a_{ij} \, , \tag{2}$$

which describes the magnetic induction exerted, at the observation point $(x_i, y_i, z_i)$, by a discrete layer of $M$ dipoles (equivalent sources) defined on the horizontal plane $z = z_c$, where $p_j$ is the magnetic moment intensity (in A m $^2$) of the $j$th dipole, that has unit volume and is located at the point $(x_j, y_j, z_c)$. In equation 2, $a_{ij}$ is the harmonic function

$$a_{ij} = c_m \frac{\mu_0}{4\pi} \hat{\mathbf{F}}^\top \mathbf{H}_{ij} \, \hat{\mathbf{u}} \, , \tag{3}$$

the unit vector

$$\hat{\mathbf{u}} = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} \cos(I)\,\cos(D) \\ \cos(I)\,\sin(D) \\ \sin(I) \end{bmatrix} , \tag{4}$$

defines the magnetization direction of all dipoles, with constant inclination $I$ and declination $D$, $\mu_0 = 4\pi\, 10^{-7}$ H/m is the magnetic constant, $c_m = 10^9$ is a factor that transforms the magnetic induction from Tesla (T) to nanotesla (nT) and $\mathbf{H}_{ij}$ is a $3 \times 3$ matrix

$$\mathbf{H}_{ij} = \begin{bmatrix} h_{ij}^{xx} & h_{ij}^{xy} & h_{ij}^{xz} \\ h_{ij}^{xy} & h_{ij}^{yy} & h_{ij}^{yz} \\ h_{ij}^{xz} & h_{ij}^{yz} & h_{ij}^{zz} \end{bmatrix} , \tag{5}$$

where

$$h_{ij}^{\alpha\beta} = \begin{cases} \frac{3(\alpha_i - \alpha_j)^2}{r_{ij}^5} - \frac{1}{r_{ij}^3} , & \alpha = \beta \\ \frac{3(\alpha_i - \alpha_j)(\beta_i - \beta_j)}{r_{ij}^5} , & \alpha \neq \beta \end{cases} , \quad \alpha, \beta = x, y, z , \tag{6}$$

are the second derivatives of the inverse distance function

$$\frac{1}{r_{ij}} = \frac{1}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_c)^2}} \tag{7}$$

with respect to the coordinates of the observation point $(x_i, y_i, z_i)$.

Equation 2 can be rewritten in matrix notation as follows:

$$\mathbf{d}(\mathbf{p}) = \mathbf{A}\mathbf{p} , \tag{8}$$

where $\mathbf{d}(\mathbf{p})$ is the $N \times 1$ predicted data vector with $i$th element defined be the predicted total-field anomaly $\Delta T_i$ (equation 2), $\mathbf{p}$ is the $M \times 1$ parameter vector whose $j$th element is the magnetic moment intensity $p_j$ of the $j$th dipole and $\mathbf{A}$ is the $N \times M$ sensitivity matrix with element $ij$ defined by the harmonic function $a_{ij}$ (equation 3). In the classical equivalent-layer technique, the common approach for estimating the parameter vector $\mathbf{p}$ from the observed total-field anomaly data $\mathbf{d}^o$ is solving the least-squares normal equations

$$\mathbf{A}^\top \mathbf{A}\, \mathbf{p} = \mathbf{A}^\top \mathbf{d}^o . \tag{9}$$

This equation is usually solved by first computing the Cholesky factor of matrix $\mathbf{A}^\top\mathbf{A}$ and then using it to solve the linear system (Golub and Loan, 2013, p. 262). The estimated parameter vector obtained by following this approach will be referenced throughout this work as the *classical solution*.

The computational cost required for estimating the classical solution can be very high when dealing with large datasets. In the following subsections, we will show how to explore the structure of the sensitivity matrix $\mathbf{A}$ and efficiently solve the least-squares normal equations (equation 9).

## Matrix A in terms of matrix components $\mathbf{A}_{\boldsymbol{\alpha\beta}}$

To access the structure of the sensitivity matrix $\mathbf{A}$ (equation 8), let us first rewrite its elements $a_{ij}$ (equation 3) in the following way:

$$a_{ij} = a_{ij}^{xx} + a_{ij}^{xy} + a_{ij}^{xz} + a_{ij}^{yy} + a_{ij}^{yz} + a_{ij}^{zz} \ , \tag{10}$$

where

$$a_{ij}^{\alpha\beta} = \begin{cases} c_m \, \frac{\mu_0}{4\pi} \, (F_\alpha u_\beta) \, h_{ij}^{\alpha\beta} & , \quad \alpha = \beta \\ c_m \, \frac{\mu_0}{4\pi} \, (F_\alpha u_\beta + F_\beta u_\alpha) \, h_{ij}^{\alpha\beta} & , \quad \alpha \neq \beta \end{cases} \quad , \quad \alpha, \beta = x, y, z \ , \tag{11}$$

are defined by the elements of $\hat{\mathbf{F}}$ (equation 1), $\hat{\mathbf{u}}$ (equation 4) and $\mathbf{H}_{ij}$ (equations 5 and 6). Then, we can rewrite the sensitivity matrix $\mathbf{A}$ (equation 8) according to:

$$\mathbf{A} = \mathbf{A_{xx}} + \mathbf{A_{xy}} + \mathbf{A_{xz}} + \mathbf{A_{yy}} + \mathbf{A_{yz}} + \mathbf{A_{zz}} \ , \tag{12}$$

where $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ are $N \times M$ matrices with elements $ij$ defined by $a_{ij}^{\alpha\beta}$ (equation 11).

Now we can define the structure of $\mathbf{A}$ in terms of its components $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ (equation 12). To do this, let us consider that the observed total-field anomaly is located on an $N_x \times N_y$ regular grid of points spaced by $\Delta_x$ and $\Delta_y$ along the $x$- and $y$-directions, respectively, on a constant vertical coordinate $z_0$. We also consider that the equivalent layer is formed by one dipole right below each observation point, at the constant coordinate $z_c$. In this case, the number of equivalent sources $M$ is equal to the number of data $N$ and, consequently, matrices $\mathbf{A}$ and $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ become square ($N \times N$). Besides, the horizontal coordinates $x_i$ and $y_i$ of the observation points can be defined by

$$x_i = x_1 + [k(i) - 1] \, \Delta_x \tag{13}$$

and

$$y_i = y_1 + [l(i) - 1] \, \Delta_y \, , \tag{14}$$

where $x_1$ and $y_1$ are the lower limits for $x_i$ and $y_i$, respectively, and $k(i)$ and $l(i)$ are integer functions defined according to the orientation of the data grid (For details, please refer to Takahashi et al. (2020)). For *$x$-oriented grids*, the integer functions are given by

$$k(i) = i - \left\lceil \frac{i}{N_x} \right\rceil N_x + N_x \tag{15}$$

and

$$l(i) = \left\lceil \frac{i}{N_x} \right\rceil . \tag{16}$$

For *$y$-oriented grids*, the integer functions are given by

$$k(i) = \left\lceil \frac{i}{N_y} \right\rceil \tag{17}$$

and

$$l(i) = i - \left\lceil \frac{i}{N_y} \right\rceil N_y + N_y \, . \tag{18}$$

In equations 15–18, $\lceil \cdot \rceil$ denotes the ceiling function (e.g., Graham et al., 1994, p. 67-68). Equations 13–18 can also be used to define the coordinates $x_j$ and $y_j$ of the equivalent sources, but with index $j$ instead of $i$.

By using equations 13–18 to define the coordinates $x_i$ and $y_i$ of the observation points and $x_j$ and $y_j$ of the equivalent sources, we can rewrite the elements $h_{ij}^{\alpha\beta}$ (equation 6) of matrix $\mathbf{H}_{ij}$ (equation 5) as follows:

$$h_{ij}^{xx} = \frac{3\left(\Delta k_{ij}\,\Delta_x\right)^2}{r_{ij}^5} - \frac{1}{r_{ij}^3}\ , \tag{19}$$

$$h_{ij}^{yy} = \frac{3\left(\Delta l_{ij}\,\Delta_y\right)^2}{r_{ij}^5} - \frac{1}{r_{ij}^3}\ , \tag{20}$$

$$h_{ij}^{zz} = \frac{3\Delta_z^2}{r_{ij}^5} - \frac{1}{r_{ij}^3}\ , \tag{21}$$

$$h_{ij}^{xy} = \frac{3\left(\Delta k_{ij}\,\Delta_x\right)\left(\Delta l_{ij}\,\Delta_y\right)}{r_{ij}^5}\ , \tag{22}$$

$$h_{ij}^{xz} = \frac{3\left(\Delta k_{ij}\,\Delta_x\right)\Delta_z}{r_{ij}^5} \tag{23}$$

and

$$h_{ij}^{yz} = \frac{3\left(\Delta l_{ij}\,\Delta_y\right)\Delta_z}{r_{ij}^5}\ , \tag{24}$$

where $\Delta_z = z_c - z_0$,

$$\Delta k_{ij} = \frac{x_i - x_j}{\Delta_x} = k(i) - k(j)\ , \tag{25}$$

$$\Delta l_{ij} = \frac{y_i - y_j}{\Delta_y} = l(i) - l(j) \tag{26}$$

and

$$\frac{1}{r_{ij}} = \frac{1}{\sqrt{\left(\Delta k_{ij}\,\Delta_x\right)^2 + \left(\Delta l_{ij}\,\Delta_y\right)^2 + \Delta_z^2}}\ . \tag{27}$$

Note that the integer functions $k(i)$, $k(j)$, $l(i)$ and $l(j)$ (equations 15–18) defining $\Delta k_{ij}$ (equation 25), $\Delta l_{ij}$ (equation 26) and $\frac{1}{r_{ij}}$ (equation 27) assume different forms depending on the grid orientation. Despite of that, it can be shown that

$$\Delta k_{ij} = -\Delta k_{ji}\ , \tag{28}$$

$$\Delta l_{ij} = -\Delta l_{ji} \tag{29}$$

and

$$\frac{1}{r_{ij}} = \frac{1}{r_{ji}} \tag{30}$$

for any grid orientation.

## Matrix components $\mathbf{A}_{\alpha\beta}$ in terms of block indices $q$ and $p$

By using equations 19–27 to compute $a_{ij}^{\alpha\beta}$ (equation 11), we can show that matrices $\mathbf{A}_{\alpha\beta}$ (equation 12) assume well-defined structures that can be conveniently represented by using *block indices* $q$ and $p$ (Takahashi et al., 2020). These indices are defined by the integer functions $\Delta k_{ij}$ and $\Delta l_{ij}$ (equations 25 and 26), in terms of the indices $i$ of the observation points $(x_i, y_i, z_0)$ and $j$ of the equivalent sources $(x_j, y_j, z_c)$. For *x-oriented grids*, $Q = N_y$, $P = N_x$ and the block indices $q$ and $p$ are given by:

$$q \equiv q(i,j) = \Delta l_{ij} \tag{31}$$

and

$$p \equiv p(i,j) = \Delta k_{ij} \ , \tag{32}$$

where $\Delta k_{ij}$ and $\Delta l_{ij}$ (equations 25 and 26) are defined by integer functions $k(i)$, $k(j)$, $l(i)$ and $l(j)$ given by equations 15 and 16. For *y-oriented grids*, $Q = N_x$, $P = N_y$ and the block indices $q$ and $p$ are given by:

$$q \equiv q(i,j) = \Delta k_{ij} \tag{33}$$

and

$$p \equiv p(i,j) = \Delta l_{ij} \ , \tag{34}$$

where $\Delta k_{ij}$ and $\Delta l_{ij}$ (equations 25 and 26) are defined by integer functions $k(i)$, $k(j)$, $l(i)$ and $l(j)$ given by equations 17 and 18. Equations 31–34 differ from those presented by Takahashi et al. (2020) in the absence of the module.

In order to describe matrix components $\mathbf{A}_{\alpha\beta}$ (equation 12) in terms of block indices, we have to verify that: (i) each pair of indices $i$ and $j$ produce a pair of block indices $q$ and $p$; (ii) each element $a_{ij}^{\alpha\beta}$, defined by indices $i$ and $j$, can be represented by the alternative notation

$a_{qp}^{\alpha\beta}$, defined in terms of the associated block indices $q$ and $p$; (iii) different combinations of indices $i$ and $j$ result in the same block indices $q$ and $p$; (iv) elements $a_{ij}^{\alpha\beta}$ having the same associated block indices $q$ and $p$ also have the same numerical value; (v) matrix $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ is formed by a grid of $Q \times Q$ blocks $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$, where each block has $P \times P$ elements $a_{qp}^{\alpha\beta}$ with the same block index $q$; (vi) blocks $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$ disposed along the same block diagonal of $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ have the same block index $q$ and are equal to each other; (vii) in each block $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$, the elements $a_{qp}^{\alpha\beta}$ having the same block index $p$ are arranged along the same diagonal and are equal to each other.

Item (i) is a direct consequence of evaluating equations 31–34 for a given pair of indices $i$ and $j$. Item (ii) can be verified by substituting the block indices $q$ and $p$ (equations 31–34) into the second derivatives $h_{ij}^{\alpha\beta}$ (equations 19–24) and then use them to define the elements $a_{ij}^{\alpha\beta}$ (equation 11). The result will be the elements $a_{qp}^{\alpha\beta}$. Item (iii) can be shown by first verifying that different combinations of indices $i$ and $j$ result in the same integer functions $\Delta k_{ij}$ and $\Delta l_{ij}$ (equations 25 and 26). Then, by using equations 31–34, we conclude that different combinations of $i$ and $j$ result in the same indices $q$ and $p$. Item (iv) can be shown by noting that, according to equations 31–34, elements $a_{ij}^{\alpha\beta}$ having the same associated block indices $q$ and $p$ also have the same associated integer functions $\Delta k_{ij}$ and $\Delta l_{ij}$ (equations 25 and 26). As a consequence, the associated second derivatives $h_{ij}^{\alpha\beta}$ (equations 19–24) have the same numerical values and elements $a_{ij}^{\alpha\beta}$ as well. Items (v)-(vii) can be shown by computing the block indices $q$ and $p$ (equations 31–34) for a small data grid formed by, say, $N_x = 3$, $N_y = 2$, and generalizing the result for larger grids.

From equations 31–34, we can show that the block index $q$ assume integer values varying from $-Q+1$ to $Q-1$, while $p$ varies from $-P+1$ to $P-1$. Positive (negative) indices $q$ are associated with blocks $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$ located below (above) the main block diagonal of $\mathbf{A}_{\boldsymbol{\alpha\beta}}$. Similarly, positive (negative) indices $p$ are associated with elements $a_{qp}^{\alpha\beta}$ located below (above) the main diagonal of $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$. According to this notation, $q$ defines the block $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$ and $p$ the diagonal where the element $a_{qp}^{\alpha\beta}$ is located within this block, so that matrices $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ can be written, in a general form, as a grid of $Q \times Q$ blocks

$$\mathbf{A}_{\boldsymbol{\alpha\beta}} = \begin{bmatrix} \mathbf{A}_{\boldsymbol{\alpha\beta}}^{0} & \mathbf{A}_{\boldsymbol{\alpha\beta}}^{-1} & \cdots & \mathbf{A}_{\boldsymbol{\alpha\beta}}^{-Q+1} \\ \mathbf{A}_{\boldsymbol{\alpha\beta}}^{1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{A}_{\boldsymbol{\alpha\beta}}^{-1} \\ \mathbf{A}_{\boldsymbol{\alpha\beta}}^{Q-1} & \cdots & \mathbf{A}_{\boldsymbol{\alpha\beta}}^{1} & \mathbf{A}_{\boldsymbol{\alpha\beta}}^{0} \end{bmatrix}_{N \times N} , \tag{35}$$

with blocks $\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q}$ given by

$$\mathbf{A}_{\boldsymbol{\alpha\beta}}^{q} = \begin{bmatrix} a_{q0}^{\alpha\beta} & a_{q(-1)}^{\alpha\beta} & \cdots & a_{q(-P+1)}^{\alpha\beta} \\ a_{q1}^{\alpha\beta} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{q(-1)}^{\alpha\beta} \\ a_{q(P-1)}^{\alpha\beta} & \cdots & a_{q1}^{\alpha\beta} & a_{q0}^{\alpha\beta} \end{bmatrix}_{P \times P} . \tag{36}$$

This well-defined structure (equations 35 and 36) assumed by matrix components $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ (equation 12) is called Block-Toeplitz Toeplitz-Block (BTTB) (e.g., Chan and Jin, 2007, p. 67).

## Structure of matrix components $\mathbf{A}_{\boldsymbol{\alpha\beta}}$

Equations 35 and 36 define the general BTTB structure of all matrix components $\mathbf{A}_{\boldsymbol{\alpha\beta}}$, but there are some differences between them. The integer functions $\Delta k_{ij}$ and $\Delta l_{ij}$ (equations 25 and 26) play a crucial role in defining the individual structures of matrix components $\mathbf{A}_{\boldsymbol{\alpha\beta}}$ (equation 35). These integer functions may assume positive and negative values depending on the relative position of the observation point $(x_i, y_i, z_0)$ and the equivalent source $(x_j, y_j, z_c)$.

Let us consider the matrix component $\mathbf{A}_{\boldsymbol{xx}}$, with elements $a^{xx}_{ij}$ (equation 11) defined by the second derivative $h^{xx}_{ij}$ (equation 19). It can be easily verified from equations 28 and 30 that $h^{xx}_{ij} = h^{xx}_{ji}$. As a consequence, $a^{xx}_{ij} = a^{xx}_{ji}$ and $\mathbf{A}_{\boldsymbol{xx}}$ is symmetric. Now, let us investigate the elements $a^{xx}_{qp}$ forming the blocks $\mathbf{A}^{q}_{\boldsymbol{xx}}$. For $x$-oriented grids, the block indices $q$ and $p$ are defined by equations 31 and 32 and $a^{xx}_{qp}$ can be written as follows:

$$a^{xx}_{qp} = c_m \, \frac{\mu_0}{4\pi} \, (F_x u_x) \, \frac{3 \, (p \, \Delta_x)^2}{r^5_{qp}} - \frac{1}{r^3_{qp}} \; , \tag{37}$$

where

$$\frac{1}{r_{qp}} = \frac{1}{\sqrt{(p \, \Delta_x)^2 + (q \, \Delta_y)^2 + \Delta_z^2}} \; . \tag{38}$$

For $y$-oriented grids, the block indices $q$ and $p$ are defined by equations 33 and 34 and $a^{xx}_{qp}$ can be written as follows:

$$a^{xx}_{qp} = c_m \, \frac{\mu_0}{4\pi} \, (F_x u_x) \, \frac{3 \, (q \, \Delta_x)^2}{r^5_{qp}} - \frac{1}{r^3_{qp}} \; , \tag{39}$$

where

$$\frac{1}{r_{qp}} = \frac{1}{\sqrt{(q \, \Delta_x)^2 + (p \, \Delta_y)^2 + \Delta_z^2}} \; . \tag{40}$$

From equations 37–40, we can easily verify that $a^{xx}_{qp} = a^{xx}_{(-q)p}$ and that $a^{xx}_{qp} = a^{xx}_{q(-p)}$ for any grid orientation. It means that the blocks $\mathbf{A}^{q}_{\boldsymbol{xx}} = \mathbf{A}^{(-q)}_{\boldsymbol{xx}}$ and that they are symmetric. From this results we conclude the matrix component $\mathbf{A}_{\boldsymbol{xx}}$ is symmetric-Block-Toeplitz symmetric-Toeplitz-Block for any grid orientation. The same reasoning can be used to show that matrices $\mathbf{A}_{\boldsymbol{yy}}$ and $\mathbf{A}_{\boldsymbol{zz}}$ also have this structure.

PAREI AQUI

By substituting equations 13 and 14 in equation **??**, we can also describe the elements of $\mathbf{H_{xy}}$, as:

$$h^{xy}_{ij} = \frac{3(\Delta k_{ij} \, \Delta x)(\Delta l_{ij} \, \Delta y)}{\left[(\Delta k_{ij} \, \Delta x)^2 + (\Delta l_{ij} \, \Delta y)^2 + (\Delta z)^2\right]^{\frac{5}{2}}} \; , \tag{41}$$

The component $\mathbf{H_{xy}}$ (equation 41) of matrix $\mathbf{H}$ (equation **??**) are skew symmetric-Block-Toeplitz skew symmetric-Toeplitz-Block matrices. This means that $\mathbf{H_{xy}}$ is Toeplitz and skew symmetric by its blocks and each of the blocks are skew symmetric Toeplitz matrices. This way, matrix $\mathbf{H_{xy}}$ can be described by the *block index* $q$ that represents the block diagonals of this matrix as a grid of $Q \times Q$ blocks $\mathbf{H}^q_{\mathbf{xy}}$, $q = -Q+1, \ldots, 0, \ldots, Q-1$:

$$\mathbf{H_{xy}} = \begin{bmatrix} \mathbf{H}^0_{\mathbf{xy}} & \mathbf{H}^{-1}_{\mathbf{xy}} & \cdots & \mathbf{H}^{-Q+1}_{\mathbf{xy}} \\ \mathbf{H}^1_{\mathbf{xy}} & \mathbf{H}^0_{\mathbf{xy}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{H}^{-1}_{\mathbf{xy}} \\ \mathbf{H}^{Q-1}_{\mathbf{xy}} & \cdots & \mathbf{H}^1_{\mathbf{xy}} & \mathbf{H}^0_{\mathbf{xy}} \end{bmatrix}_{N \times N} . \tag{42}$$

And each diagonal of the blocks are represented by $P \times P$ elements $h^q_p$, $p = -P + 1, \ldots, 0, \ldots, P-1$:

$$\mathbf{H}^q_{\mathbf{xy}} = \{h^q_p\} = \begin{bmatrix} h^q_0 & h^q_{-1} & \cdots & h^q_{-P+1} \\ h^q_1 & h^q_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & h^q_{-1} \\ h^q_{P-1} & \cdots & h^q_1 & h^q_0 \end{bmatrix}_{P \times P} . \tag{43}$$

In a *x-oriented grid* $Q = N_y$, $P = N_x$ and $q$ and $p$ can be defined by the functions:

$$q(i,j) = l(i) - l(j) \tag{44}$$

and

$$p(i,j) = k(i) - k(j) \quad , \tag{45}$$

where $l(i)$ and $l(j)$ are defined by equation 16 and $k(i)$ and $k(j)$ are defined by equation 15. For $y$-oriented grids, $Q = N_x$, $P = N_y$ and the block indices $q$ and $p$ are defined, respectively, by the following integer functions of the matrix indices $i$ and $j$:

$$q(i,j) = k(i) - k(j) \tag{46}$$

and

$$p(i,j) = \ l(i) - l(j) \quad , \tag{47}$$

Important to clarify that in this case, as a skew symmetric matrix, the values of oposing diagonals have oposing signals, e.g., $\mathbf{H}_{\mathbf{xy}}^{-1} = -\mathbf{H}_{\mathbf{xy}}^{1}$ and $h_{-1}^{q} = -h_{1}^{q}$.

By substituting equations 13 and 14 in equation **??**, the elements of $\mathbf{H}_{\mathbf{xz}}$, are given by:

$$h_{ij}^{xz} = \frac{3(\Delta k_{ij}\,\Delta x)(\Delta z)}{\left[(\Delta k_{ij}\,\Delta x)^2 + (\Delta l_{ij}\,\Delta y)^2 + (\Delta z)^2\right]^{\frac{5}{2}}} \ , \tag{48}$$

The component $\mathbf{H}_{\mathbf{xz}}$ (equation 48) of matrix $\mathbf{H}$ (equation **??**) are skew symmetric-Block-Toeplitz symmetric-Toeplitz-Block matrices. This means that $\mathbf{H}_{\mathbf{xz}}$ is Toeplitz and skew symmetric by its blocks and each of the blocks are symmetric Toeplitz matrices. Thus, matrix $\mathbf{H}_{\mathbf{xz}}$ can be described by the *block index* $q$ that represents the block diagonals of this matrix as a grid of $Q \times Q$ blocks $\mathbf{H}_{\mathbf{xz}}^{q}$, $q = -Q+1, \ldots, 0, \ldots, Q-1$:

$$\mathbf{H}_{\mathbf{xz}} = \begin{bmatrix} \mathbf{H}_{\mathbf{xz}}^{0} & \mathbf{H}_{\mathbf{xz}}^{-1} & \cdots & \mathbf{H}_{\mathbf{xz}}^{-Q+1} \\ \mathbf{H}_{\mathbf{xz}}^{1} & \mathbf{H}_{\mathbf{xz}}^{0} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{H}_{\mathbf{xz}}^{-1} \\ \mathbf{H}_{\mathbf{xz}}^{Q-1} & \cdots & \mathbf{H}_{\mathbf{xz}}^{1} & \mathbf{H}_{\mathbf{xz}}^{0} \end{bmatrix}_{N \times N} . \tag{49}$$

And each diagonal of the blocks are represented by $P \times P$ elements $h_{p}^{q}$, $p = 0, \ldots, P-1$:

$$\mathbf{H}_{\mathbf{xz}}^{q} = \{h_{p}^{q}\} = \begin{bmatrix} h_{0}^{q} & h_{1}^{q} & \cdots & h_{P-1}^{q} \\ h_{1}^{q} & h_{0}^{q} & \ddots & \vdots \\ \vdots & \ddots & \ddots & h_{1}^{q} \\ h_{P-1}^{q} & \cdots & h_{1}^{q} & h_{0}^{q} \end{bmatrix}_{P \times P} . \tag{50}$$

In a *x-oriented grid* $Q = N_y$, $P = N_x$ and $q$ and $p$ can be defined by the functions:

$$q(i,j) = \ l(i) - l(j) \tag{51}$$

and

$$p(i,j) = \mid k(i) - k(j) \mid \quad , \tag{52}$$

where $l(i)$ and $l(j)$ are defined by equation 16 and $k(i)$ and $k(j)$ are defined by equation 15. For $y$-oriented grids, $Q = N_x$, $P = N_y$ and the block indices $q$ and $p$ are defined, respectively, by the following integer functions of the matrix indices $i$ and $j$:

$$q(i,j) = \; k(i) - k(j) \tag{53}$$

and

$$p(i,j) = \mid l(i) - l(j) \mid \quad , \tag{54}$$

In this case as a skew symmetric matrix by blocks, the values of oposing diagonals blocks have oposing signals, e.g., $\mathbf{H_{xz}^{-1}} = -\mathbf{H_{xz}^{1}}$ but each block is a symmetric matrix.

Finally, by substituting equations 13 and 14 in equation **??**, we can also describe the elements of $\mathbf{H_{yz}}$, as:

$$h_{ij}^{yz} = \frac{3(\Delta l_{ij}\,\Delta y)(\Delta z)}{\left[(\Delta k_{ij}\,\Delta x)^2 + (\Delta l_{ij}\,\Delta y)^2 + (\Delta z)^2\right]^{\frac{5}{2}}} \,, \tag{55}$$

The component $\mathbf{H_{yz}}$ (equation 55) of matrix $\mathbf{H}$ (equation **??**) are symmetric-Block-Toeplitz skew symmetric-Toeplitz-Block matrices. This means that $\mathbf{H_{yz}}$ is Toeplitz and symmetric by its blocks and each of the blocks are skew symmetric Toeplitz matrices. Thus, matrix $\mathbf{H_{yz}}$ can be described by the *block index q* that represents the block diagonals of this matrix as a grid of $Q \times Q$ blocks $\mathbf{H_{yz}^{q}}$, $q = 0, \ldots, Q-1$:

$$\mathbf{H_{yz}} = \begin{bmatrix} \mathbf{H_{yz}^{0}} & \mathbf{H_{yz}^{1}} & \cdots & \mathbf{H_{yz}^{Q-1}} \\ \mathbf{H_{yz}^{1}} & \mathbf{H_{yz}^{0}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{H_{yz}^{1}} \\ \mathbf{H_{yz}^{Q-1}} & \cdots & \mathbf{H_{yz}^{1}} & \mathbf{H_{yz}^{0}} \end{bmatrix}_{N\times N} \quad . \tag{56}$$

And each diagonal of the blocks are represented by $P \times P$ elements $h_p^q$, $p = -P + 1, \ldots, 0, \ldots, P-1$:

$$\mathbf{H}^q_{\mathbf{yz}} = \{h_p^q\} = \begin{bmatrix} h_0^q & h_{-1}^q & \cdots & h_{-P+1}^q \\ h_1^q & h_0^q & \ddots & \vdots \\ \vdots & \ddots & \ddots & h_{-1}^q \\ h_{P-1}^q & \cdots & h_1^q & h_0^q \end{bmatrix}_{P \times P} . \tag{57}$$

In a *x-oriented grid* $Q = N_y$, $P = N_x$ and $q$ and $p$ can be defined by the functions:

$$q(i,j) = \mid l(i) - l(j) \mid \tag{58}$$

and

$$p(i,j) = \ k(i) - k(j) \quad , \tag{59}$$

where $l(i)$ and $l(j)$ are defined by equation 16 and $k(i)$ and $k(j)$ are defined by equation 15. For $y$-oriented grids, $Q = N_x$, $P = N_y$ and the block indices $q$ and $p$ are defined, respectively, by the following integer functions of the matrix indices $i$ and $j$:

$$q(i,j) = \mid k(i) - k(j) \mid \tag{60}$$

and

$$p(i,j) = \ l(i) - l(j) \quad , \tag{61}$$

Being a symmetric matrix by blocks, the values of $\mathbf{H_{yz}}$ from oposing diagonals blocks are equal, but each block have skew symmetric oposing diagonals, i.e., $h_{-1}^q = -h_1^q$.

In general, matrix $\mathbf{A}$ (equation 3) is a non-symmetric BTTB, i.e., its blocks are non-symmetric ($\mathbf{A}^{-Q+1} \neq \mathbf{A}^{Q-1}$) and its elements also are non-symmetric ($a_{-1}^q \neq a_1^q$). Depending on specific values of the main field direction and the equivalent sources magnetization directions, matrix $\mathbf{A}$ can assume other structures, for example, when $\hat{\mathbf{F}} = [0,0,1]$ and

$\hat{\mathbf{u}} = [0, 0, 1]$ it becomes symmetric. In this work, we are considering the more commom situation for the matrix $\mathbf{A}$.

Also differently for the symmetric sensitivity matrix described by Takahashi et al. (2020), the non-symmetric BTTB matrix cannot be reconstructed only by its first column. The construction of the matrix $\mathbf{A}$ (equation 3) needs four columns: the first and last columns of the first column of blocks and the first and last columns of the last column of blocks. This has a physical implication in the equivalent layer which is not possible to use only one equivalent source to reprduce all elements of matrix $\mathbf{A}$, such as in the gravity case as demonstrared by Takahashi et al. (2020). Rather, in the magnetic case it takes four equivalent sources positioned at each corner of the equivalent layer. Figure **??** shows the positioning of the equivalent sources in a regular grid $N_x = 4$ and $N_y = 3$ necessary to calculate the four columns capable of recover the matrix $\mathbf{A}$.

In this work, we propose a different approach, by calculating the first column of all six different components of second derivatives matrices from $\mathbf{H}_{ij}$ (equation **??**). These matrices are, in fact, symmetrics or skew-symmetrics BTTBs, meaning that the first column has all elements of each matrix.

## Standard Conjugate Gradient Least Squares (CGLS) method

The computational cost associated with the classical method for estimating the parameter vector $\mathbf{p}$ by solving the linear system 9 can be very high or ever prohibitive when dealing with large data sets. In these cases, a well-known alternative is solving the normal equations (equation 9) iteratively by using the standard Conjugate Gradient Least Squares (CGLS) method. Below we present a pseudocode for the standard CGLS method:

**Algorithm 1** Standard CGLS pseudocode (Aster et al., 2019, p. 166).

```
Input: $\mathbf{A}$ and $\mathbf{d}^o$.

Output: Estimated parameter vector $\tilde{\mathbf{p}}$.

Set $it = 0$, $\tilde{\mathbf{p}}_{(it)} = \mathbf{0}$, $\mathbf{c}_{(it-1)} = \mathbf{0}$, $\beta_{(it)} = 0$, $\mathbf{s}_{(it)} = \mathbf{d}^o$ and $\mathbf{r}_{(it)} = \mathbf{A}^\top \mathbf{s}_{(it)}$.
1 - If $it > 0$, $\beta_{(it)} = \dfrac{\|\mathbf{r}_{(it)}\|_2^2}{\|\mathbf{r}_{(it-1)}\|_2^2}$
2 - $\mathbf{c}_{(it)} = \mathbf{r}_{(it)} + \beta_{(it)}\, \mathbf{c}_{(it-1)}$
3 - $\alpha_{(it)} = \dfrac{\|\mathbf{r}_{(it)}\|_2^2}{\|\mathbf{A}\, \mathbf{c}_{(it)}\|_2^2}$
4 - $\tilde{\mathbf{p}}_{(it+1)} = \tilde{\mathbf{p}}_{(it)} + \alpha_{(it)}\, \mathbf{c}_{(it)}$
5 - $\mathbf{s}_{(it+1)} = \mathbf{s}_{(it)} - \alpha_{(it)}\, \mathbf{A}\, \mathbf{c}_{(it)}$
6 - $\mathbf{r}_{(it+1)} = \mathbf{A}^\top\, \mathbf{s}_{(it+1)}$
7 - $it = it + 1$
8 - Repeat previous steps until convergence.
```

Setting a convergence criteria based on the minimum tolerance of the residuals is a good option to carry out this algorithm efficiently and still obtaining very good results. Another possibility is to set an invariance to the Euclidean norm of residuals between iterations, which would increase algorithm runtime, but with smaller residuals. We chose the first option, as we achieve satisfactory results. The estimated parameter vector obtained by using the standard CGLS method (Algorithm 1) will be referenced throughout this work as the *standard CGLS solution*.

Note that the standard CGLS solution (Algorithm 1) requires neither inverse matrix nor matrix-matrix product. Instead, it only requires: one matrix-vector product out of the loop and two matrix-vector products per iteration (in steps 3 and 6). In the following subsections, we will show how to compute these three matrix-vector products efficiently by exploring the structure of the sensitivity matrix $\mathbf{A}$.

PAREI AQUI

## CGLS matrix-vector substitution

As pointed earlier in this work, the main improvement inside the CGLS method (Algorithm 1) for estimating the parameter vector $\hat{\mathbf{p}}$ (equation **??**) is to substitute the matrix-vector multiplication $\mathbf{A}^\top \mathbf{s}^{(0)}$ out of the loop and the two matrix-vector multiplications inside the loop at steps 3 an 6, $\mathbf{A}\,\mathbf{c}^{(it)}$ and $\mathbf{A}^\top\,\mathbf{s}^{(it+1)}$, that is necessary at each iteration and takes most of its runtime.

Our method consists in calculating the six first columns of the second derivatives of $\mathbf{H}$ (equation **??**) and embedding them into the first six columns of the block-circulant circulant-block (BCCB) matrices related to the $\mathbf{H}$ components. Thus, it is possible to calculate the first column of the BCCB matrix embbeded from matrix $\mathbf{A}$ (equation 3) by multiplying each component with its respective constants and summing as shown in equation 10. In Takahashi et al. (2020), Appendix A, the authors demonstrated in details how to transform a symmetric BTTB matrix into a BCCB matrix $\mathbf{C}$. The process here is the same and that work can be referenced to achieve the same results.

A new auxuliary linear system is constructed to carry the matrix-vector product:

$$\mathbf{w} = \mathbf{C}\mathbf{v}\,, \tag{62}$$

where

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_0 \\ \vdots \\ \mathbf{w}_{Q-1} \\ \mathbf{0}_{2N\times 1} \end{bmatrix}_{4N\times 1} , \tag{63}$$

$$\mathbf{w}_q = \begin{bmatrix} \mathbf{d}_q(\mathbf{p}) \\ \mathbf{0}_{P\times 1} \end{bmatrix}_{2P\times 1} , \tag{64}$$

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_0 \\ \vdots \\ \mathbf{v}_{Q-1} \\ \mathbf{0}_{2N\times 1} \end{bmatrix}_{4N\times 1} , \tag{65}$$

and

$$\mathbf{v}_q = \begin{bmatrix} \mathbf{p}_q \\ \mathbf{0}_{P\times 1} \end{bmatrix}_{2P\times 1} , \tag{66}$$

where $\mathbf{C}$ (equation 62) is a $4N \times 4N$ non-symmetric (BCCB) resulted from transforming $\mathbf{A}$ (equation 3). Without having to calculate the whole BCCB matrix, its first column can be used to carry the multiplication of this new system (equation 62). Appendix A and C in Takahashi et al. (2020) shows how to use the 2D-FFT to compute the eigenvalues of matrix $\mathbf{C}$, store in a $2Q \times 2P$ matrix using the *vec*-operator and to carry the matrix-vector product. Denoting the matrix $\mathbf{L}$ as the eigenvalues matrix follows:

$$\mathbf{F}^*_{2Q} \left[ \mathbf{L} \circ (\mathbf{F}_{2Q}\, \mathbf{V}\, \mathbf{F}_{2P}) \right] \mathbf{F}^*_{2P} = \mathbf{W} \, , \tag{67}$$

where the symbol "$\circ$" references the Hadamard product, i.e., a element-wise complex multiplication between the eingenvalues and the 2D-FFT of the matrix rearranged along the rows of the parameters $\mathbf{V}$ (equation 65) using the same *vec*-operator. The resulting inverse

2D-FFT denoted by $\mathbf{F}_{2Q}^{*} \otimes \mathbf{F}_{2P}^{*}$ is also a $2Q \times 2P$ matrix ($\mathbf{W}$) that can be rearranged to the predicted data vector $\mathbf{d}(\hat{\mathbf{p}})$ size $N$.

## Computational performance

To compare the efficiency of our algorithm we will use a numerical approach and calculate the floating-point operations (*flops*), i.e., count the number of mathematical operations necessary to complete the estimative of parameter vector $\hat{\mathbf{p}}$ of the normal equations (equation **??**) and both the CGLS methods (algorithm 1) for calculating the matrix-vector product by its standart way and our approach.

The *flops* needed to solve the linear system in equation **??** using the Cholesky factorization is:

$$f_{classical} = \frac{7}{3}N^3 + 6N^2 \, , \tag{68}$$

where $N$ is the total number of observation points and also the size of estimated parameter vector $\hat{\mathbf{p}}$.

For the more efficient CGLS algorithm the estimative can be done in:

$$f_{cgls} = 2N^2 + it\,(4N^2 + 12N) \, . \tag{69}$$

However, our approach reduces further to:

$$f_{ours} = \kappa\, 16N \log_2(4N) + 24N + it\,(\kappa\, 16N \log_2(4N) + 60N) \, , \tag{70}$$

where $\kappa$ depends on the FFT algorithm. By default, in this work we will use $\kappa = 5$ for the *radix-2* algorithm (Van Loan, 1992).

Figure **??** shows a comparative between the methods varying the number of observation points up to $1,000,000$, where it is possible to observe a reduction of $10^7$ orders of magnitude to estimate parameter vector $\hat{\mathbf{p}}$ in relation to the non-iterative classical method and $10^3$ orders of magnitude in relation to the standart CGLS algorithm using 50 iterations. A more detailed, step by step, flops count of the classical and CGLS algorithm can be found in Appendix A.

In Figure **??** we show the time necessary to construct matrix $\mathbf{A}$ (equation 3) and solve the linear system up to $10,000$ points of observation. With this dataset the classical method takes more than sixty-three seconds, the CGLS more than twelve seconds, while our method takes only half a second. The cpu used for this test was a intel core i7-7700HQ@2.8GHz.

In Figure **??** a comparison between the time to complete the task to calculate the first column of the BCCB matrix embbeded from the from $\mathbf{A}$ (equation 3) by using only one equivalent source, i.e., calculating all six first column of the second derivatives matrices from $\mathbf{H}$ (equation **??**) and using four equivalent sources to calculate the four necessary columns from the non-symmetric matrix $\mathbf{A}$ (equation 3). Although, very similar in time, with one source a small advantage can be observed as the number of data $N$ increases and goes beyond $N = 200,000$. This test was done from $N = 10,000$ to $N = 700,000$ with increases of $5,625$ observation points.

In Table 1 there is comparison between how much RAM memory is adressed to store the sensitivity matrix for each of the methods. The classical approach and the CGLS have to store the whole matrix $\mathbf{A}$ (equation 3), this means that a dataset with for example $N = 10,000$ observation points, the sensitivity matrix has $N^2 = 100,000,000$ elements and takes approximately 763 Megabytes of memory (8 bytes per element). For our method, it is necessary to store the first six columns of each of the components from matrix $\mathbf{H}$ (equation **??**) embedded into the BCCB matrices. With the same dataset $N = 10,000$

it needs 1.831 Megabytes. After completing the steps to store the eigenvalues of matrix $\mathbf{C}$ (equation 62) it takes only 0.6104 Megabytes. Here, we are considering 16 bytes per element as the eigenvalues are complex numbers resulting from the 2D FFT. For a bigger dataset as $N = 1,000,000$ the amount of RAM necessary goes to $7,629,395$, 183.096 and 61.035 Megabytes, respectively, showing the necessity to find improved and efficient methods for the equivalent layer technique as the one presented in this work. We remember that throughout our work we are always considering $N = M$.

## APPLICATION TO SYNTHETIC DATA

The synthetic data application of the fast equivalent layer for magnetic data was conducted on a regular grid of $80 \times 80$ points, totaling $N = 6,\ 400$ observation points. Three bodies were modeled: two prisms and a sphere with inclination, declination and intensity of $0°$ and $45°$ and $2\sqrt{2}$ A/m, respectively. The main field has inclination and declination of $10°$ and $37°$, respectively. Figure **??** shows the synthetic data created for this test.

Using a classical linear inversion method (equation **??**) a predicted data was estimated in Figure **??**a. The data residuals, defined as the difference between the observed (Figure **??**) and the predicted data (Figure **??**a), with mean of 0.3712 nT and standart deviation of 0.2798 nT are shown in Figure **??**b. This process took 17.10 seconds.

Using the CGLS method with the fast BTTB matrix-vector product a predicted data was estimated in Figure **??**a. The data residuals, defined as the difference between the observed (Figure **??**) and the predicted data (Figure **??**a), with mean of 0.5150 nT and standart deviation of 0.4363 nT is shown in Figure **??**b. This process took 0.18 seconds.

Figure **??** shows the convergence of our method to estimate the equivalent sources parameter vector $\mathbf{d}(\hat{\mathbf{p}})$. The Euclidean norm of the data residuals decreases as expected when the convergence criterion was satisfied, close to iteration 50. This result shows that, in practice, it is not necessary to run the conjugate gradient least square method at $N$ iterations to get an exactly solution. Actually, the exactly solution would never occur due to roundoff errors. Hence, by setting the convergence to $N$ iterations besides being unnecessary it also demands large computer processing time, even in this synthetic test with a small layer ($N = 6,\ 400$ equivalent sources).

## Tests with data on irregular grids

As shown in the methodology, a regular grid of observation points is needed to arise the BTTB matrix. In this section, we show the results when our method is applied directly to irregular grids of $N = 5\,000$ observation points. First, we set up a regular grid of $100 \times 50$ observation points in the $x$- and $y$-directions with a grid spacing of $\Delta x$ of 101.01 m along the $x$-axis and $\Delta y$ of 163.265 m along the $y$-axis. Next, the $x$ and $y$ coordinates of the observations were also contaminated with pseudorandom Gaussian noise with zero mean and standard deviations of 10%, 30% and 50% of the $\Delta x$ and $\Delta y$ spacing.

Figure **??**a shows an irregular grid with standart deviations of 10% along both $x$- and $y$-directions of the observation points. By using the classical approach, Figure **??**b shows the data residuals, defined as the difference between the observed (Figure **??**b) and the predicted data (Figure **??**a), with mean of 0.3628 nT, standart deviation of 0.2727 nT. By using our method, Figure **??**b shows the data residuals, difference between the observed (Figure **??**b) and the predicted data (Figure **??**a), with mean of 0.6024 nT and standart deviation of 0.4998 nT. Figure **??** shows the convergence of our method in which the Euclidean norm of the data residuals decreases until it achieves an invariance close to iteration 50.

Figure **??**a shows an irregular grid with standart deviations of 20% along both $x$- and $y$-directions of the observation points. Figure **??**b shows the data residuals using the classical approach. The data residuals are defined as the difference between the observed (Figure **??**b) and the predicted data (Figure **??**a) having mean of 0.3630 nT, standart deviation of 0.2731 nT. Using our method, the data residuals (difference between the observed, Figure **??**b, and the predicted data Figure **??**a) have mean of 0.7147 nT, standart deviation of 0.5622 nT are shown in Figure **??**b. The Euclidean norm of the data residuals obtained by our method (Figure **??**) decreases as expected and close to iteration 50 congerves to a constant value.

Figure **??**a shows an irregular grid with standart deviations of 30% along both $x$- and $y$-directions of the observation points. Using the classical approach, the data residuals (difference between the observed, Figure **??**b, and the predicted data Figure **??**a, have mean of 0.3634 nT, standart deviation of 0.2735 nT and are shown in Figure **??**b. Using our method, the data residuals (difference between the observed, Figure **??**b, and the predicted data, Figure **??**a) have mean of 0.9788 nT, standart deviation of 0.7462 nT and are shown in Figure **??**b. Figure **??** shows the convergence analysis of our method. Similar to the previous results, in the begining of the iterations, the Euclidean norm of the data residuals obtained by our method decreases; however it starts increasing without achieving an invariance.

## Tests with data over a undulating observation surface.

Another set of tests were also accomplished with the same previous regular grid configuration, but now with deviations in the $z$-coordinates, i.e., the observation points were no longer in a plane. In the previous tests, the total-field anomaly was computed at 900 m height. The next tests, the $z$-coordinate of the observations were contaminated with pseudorandom Gaussian noise with zero mean and standard deviations of 5%, 10%, and 20% of the 900 m height.

Figure **??**a shows a regular grid of $100 \times 50$ in the $x$- and $y$-directions located on an uneven surface of observations where the $z$-coordinates were corrupted with a standard deviation of 5% of the 900 m height. Using the classical approach, the data residuals (difference between the observed Figure **??**b, and the predicted data, Figure **??**a) have mean of 0.3712 nT, standart deviation of 0.2870 nT and are shown in Figure ??b. Using our method, the data residuals (difference between the observed, Figure ??b, and the predicted data, Figure ??a) have mean of 0.9542 nT, standart deviation of 0.8943 nT and are shown in Figure ??b. In Figure ??, the Euclidean norm of the data residuals using our method decreases and congerves to a constant value at the iteration 50.

Figure ??a shows a regular grid of $100 \times 50$ in the $x$- and $y$-directions located on an uneven surface of observations where the $z$-coordinates were corrupted with a standard deviation of 10% of the 900 m height. Using the classical approach, the data residuals, defined as the difference between the observed (Figure ??b) and the predicted data (Figure??a), have mean of 0.3865 nT, standart deviation of 0.3216 nT and are shown in Figure ??b. Using our method, Figure ??b shows the data residuals (difference between the observed, Figure ??b, and the predicted data, Figure ??a) with mean of 1.6105 nT and standart deviation of 1.6231 nT. Likewise, Figure ?? shows that the Euclidean norm of the data residuals, which were obtained by using our method, decreases up to the iteration 50 and and reaches an invariance in the subsequent iterations.

Figure ??a shows a regular grid of $100 \times 50$ in the $x$- and $y$-directions located on an uneven surface of observations where the z coordinates were corrupted with a standard deviation of 20% of the 900 m height. Figure ??b shows, using the classical approach, the data residuals (difference between the observed, Figure ??b, and the predicted data, Figure??a) with mean of 0.4155 nT and standart deviation of 0.4005 nT. By using our method, the data residuals (difference between the observed, Figure ??b, and the predicted data, Figure ??a) are the worst results (Figure ??b) with mean of 6.6220 nT and standart deviation of 5.901 nT. The convergence analysis (Figure ??) reveals the inadequacy of our method in dealing with rugged surface of observations, as the Euclidean norm of the data residuals decreases slower than previous tests and starts increasing afterwards. Under this condition, the convergence is not achieved.

Although our method is formulated to deal with magnetic observations measured on a regular grid, in the $x$- and $y$-directions, and on a planar surface, the synthetic results show that our method is robust in dealing either with irregular grids in the horizontal directions or with uneven surface. However, the robustness of our method has limitations. The performance limitation of our method depends on the degree of the departure of the $x$- and $y$-coordinates of the data from there corresponding coordinates on a regular grid and from the amplitude of the undulating observation surface. High departures of the $x$- and $y$-coordinates from a regular grid and large variations in the data elevation ($z$-coordinates of the data) are associated with unacceptable data fittings (large data residuals) as shown in Figures **??**b and **??**b, respectively. However, the poor performance of our method in cases of irregular grid and uneven observation surface can be detected easily because, besides it leads to poor data fitting, it does not converge as shown in Figures **??** and **??**. Our results suggest that the sensitivity of our method to uncertainties in the $z$-coordinates of the observations is higher than its sensitivity to uncertainties in the $x$- and $y$-coordinates of the observations.

## APPLICATION TO FIELD DATA

The field data application was performed with the aeromagnetic data of Carajás, Pará, Brazil, provided by CPRM. The survey is composed of 131 flight lines N-S oriented, spacing $\Delta y = 3,000$ m. The magnetometer (Scintrex CS-3) was set to a interval between measurements of 0.1 s giving a spacing $\Delta x = 7.65$ m. The average flight heigth is $\Delta z = -900$ m. The total number of observation points is $N = 6,081,345$. Figure **??** shows the observed magnetic field data of the area.

For the actual data processing, we have made a comparison between an interpolated regular grid of $10,000 \times 131$ using a nearest neighbour algorithm and a decimated irregular grid, also of $10,000 \times 131$, totaling $N = 1,310,000$ observation points in both cases. The decimated grid was performed by using the regular grid created in the first case as a guide and by finding the nearest real observation point to this regular grid thus, ensuring that the irregular grid is the lesser deviant possible to conduct the BTTB scheme. In figure **??**a we show the result of the interpolation and in **??**b the result of the decimation. With $1,310,000$ observation points, it would be necessary 12.49 Terabytes of RAM to store the full sensitivity matrix with the classical approach. However, taking advantage that the second derivatives of equation **??** are symmetric or skew-symmetric matrices, it is possible to reconstruct the whole sensitivity matrix storing only the first column of each component of equation **??**, thus, using only 59.97 Megabytes, allowing desktop computers being able to process this amount of data.

As this area is very large, different values of the magnetic main field can be considered. For this processing, it was considered an approximated mid location of the area (latitude $-6.5°$ and longitude $-50.75°$) where the declination is $-19.86°$ for the IGRF model in

1st january, 2014. The inclination was calculated considering the Geocentric axial dipole model ($tang\, I = 2 \times tan\, \lambda$) and is equal to 12.84°. As the source magnetization is unknown, inclination and declination equals to the main field is being used.

To achieve high efficiency in property estimative of the equivalent sources, the method CGLS for inversion was used, combined with a fast matrix-vector product, only possible because of the BTTB structure of the sensitivity matrix. This fast matrix-vector product was also used for data processing (upward-continuation) in a very efficient way.

Using a equivalent layer at 300 meters above the ground the predicted data and its residual of the interpolated regular grid are shown in figure **??**. The mean of 0.07979 nT and the standart deviation of 0.5060 nT of the residual shows the good result of physical property estimative. It was used 200 iterations of the CGLS method taking 390.80 seconds with a Intel core i7 7700HQ@2.8GHz processor in single-processing and single-threading modes. Using the same equivalent layer and CGLS configuration the predicted data and its residual of the decimated irregular grid are shown in figure **??**. With a mean of 0.07348 nT, standart deviation of 0.3172 nT and lower residual amplitude compared to the regular grid, we show that the process of decimating the original data, without creating new observation points with interpolation, can be benefical to the method, even if an irregular grid is taken place. It took 385.56 seconds to complete the estimative.

The convergence analysis for the decimated irregular grid up to $2,000$ CGLS iterations is in figure **??**, showing good convergence rate and guaranteeing that the irregular grid is not disturbing the method.

In figure **??** the upward-continuation transformation using the estimated equivalent layer with the decimated grid was made in a horizontal plane at $5,000$ meters and took

2.64 seconds, showing good results without visible errors or border effects problems and accentuating the long wavelenghts.

# CONCLUSIONS

In this work, we were able to develop a fast equivalent layer technique for processing magnetic data with the method of Conjugate Gradient Least Square using the convolutional equivalent layer theory to obtain results of performance more than four orders of magnitude less than the classical equivalent layer. The sensitivity matrix of the magnetic equivalent layer carries the structure of BTTB matrices, which means a very low computational cost matrix-vector product and also the possibility to store only the first column of the matrix BCCB. In this work we propose a novel method to use only one equivalent source and calculating the first six columns of the inverse of distance second derivatives matrices to arrive in the first column of the BCCB matrix embbeded from the original magnetic kernel sensitivity matrix.

Synthetic tests showed similar results estimating the physical property using a classical approach to solve a linear system and our method using the CGLS combined with the BTTB matrix-vector product. The difference in time, however, is noticeable: 2.04 seconds using the classical approach and 0.083 seconds using our approach. This difference was obtained with a mid-size mesh of $80 \times 80$ points, greater results can be obtained if more observation points are used.

Real data test were also conducted in the region of Carajás, Pará, Brazil. With an irregular grid of $1,310,000$ observation points, store the full sensitivity matrix it would be necessary 12.49 Terabytes of RAM. However, taking advantage of the symmetric or skew-symmetric matrices structures, it is possible to reconstruct the whole sensitivity matrix using only 59.97 Megabytes. Using 200 iterations of the CGLS method took 385.56 seconds and very good results of property estimative were obtained. Also the upward-continuation transformation showed good results and took only 2.64 seconds.

## ACKNOWLEDGEMENTS

Diego Takahashi was supported by a Phd scholarship from CAPES. Valéria C.F. Barbosa was supported by fellowships from CNPq (grant 307135/2014-4) and FAPERJ (grant 26/202.582/2019). Vanderlei C. Oliveira Jr. was supported by fellowships from CNPq (grant 315768/2020-7) and FAPERJ (grant E-26/202.729/2018). The authors thank the Geological Survey of Brazil (CPRM) for providing the field data.

# APPENDIX A

# FLOPS COMPUTATIONS

## Classical flops count

The flops count of the classical approach to solve the linear system (equation **??**) using the Cholesky factorization is given by equation 68. The step-by-step count follows:

**(1)** $J = \mathbf{A}^\top \mathbf{A}$: $\frac{1}{2}N^3$ (lower triangular matrix-matrix product).

**(2)** $\mathbf{C_f}$ : $\frac{1}{3}N^3$ (one Gaxpy Cholesky factorization of J (Golub and Loan, 2013,p. 164)).

**(3)** $\mathbf{A}^\top \mathbf{d}^o$: $2N^2$ (one matrix-vector product).

**(4)** $\mathbf{C_f}(\mathbf{A}^\top \mathbf{d}^o)$: $2N^2$ (one matrix-vector product).

**(5)** $\mathbf{C_f}^\top(\mathbf{C_f} \mathbf{A}^\top \mathbf{d}^o)$: $2N^2$ (one matrix-vector product).

Summing all calculations:

$$f_{classical} = \frac{5}{6}N^3 + 6N^2 \, , \tag{A-1}$$

## CGLS flops count

The flops count of CGLS algorithm **??** can be summarized as:

Out of the loop:

**(1)** $\mathbf{A}^\top \mathbf{s}$: $2N^2$ (one matrix-vector product).

Inside the loop:

**(1)** $\dfrac{\mathbf{r}^{(it)^\top}\,\mathbf{r}^{(it)}}{\mathbf{r}^{(it-1)^\top}\,\mathbf{r}^{(it-1)}}$: $4N$ (two vector-vector products).

**(2)** $\mathbf{r}^{it} - \alpha_{it}\,\beta_{it}\,\mathbf{c}^{(it-1)}$: $2N$ (one scalar-vector product and one vector subtraction).

**(3)** $\frac{||\mathbf{r}^{(it)}||_2{}^2}{(\mathbf{c}^{(it)^\top}\,\mathbf{A}^\top)(\mathbf{A}\,\mathbf{c}^{(it)})}$: $2N^2 + 2N$ (one matrix-vector and one vector-vector product).

**(4)** $\hat{\mathbf{p}}^{it} - \alpha_{it}\,\mathbf{c}^{(it)}$: $2N$ (one vector subtraction).

**(5)** $\mathbf{s}^{it} - \alpha_{it}\,\mathbf{A}\,\mathbf{c}^{(it)}$: $2N$ (one vector subtraction, the matrix-vector product was calculated in step 3).

**(6)** $\mathbf{A}^\top\,\mathbf{s}^{(it+1)}$: $2N^2$ (one matrix-vector product).

The result of all flops count leads to:

$$f_{cgls} = 2N^2 + it\,(4N^2 + 12N)\,. \tag{A-2}$$

## Our modified CGLS flops count

All the flops count presented in previous section for the CGLS remains, only substituting the out of the loop matrix-vector product in step 1 and the two matrix-vector products inside the loop in steps 3 and 6. The computations necessary to carry the matrix-vector used in this work are given by:

**(1)** $\mathbf{L}$: $\kappa\,4N\log_2(4N)$ (one 2D FFT for the eigenvalues calculation of the sensitivity matrix $\mathbf{A}$ or the transposed sensitivity matrix $\mathbf{A}^\top$).

**(2)** $\mathbf{F}_{2Q}\,\mathbf{V}\,\mathbf{F}_{2P}$: $\kappa\,4N\log_2(4N)$ (one 2D FFT).

**(3)** $\mathbf{L}\circ(\mathbf{F}_{2Q}\,\mathbf{V}\,\mathbf{F}_{2P})$: $24N$ (one complex Hadamard matrix multiplication).

**(4)** $\mathbf{F}_{2Q}^{*}\left[\mathbf{L}\circ(\mathbf{F}_{2Q}\,\mathbf{V}\,\mathbf{F}_{2P})\right]\mathbf{F}_{2P}^{*}$: $\kappa\, 4N\log_2(4N)$ (one inverse 2D FFT).

Matrix-vector product total: $\kappa\, 12N\log_2(4N) + 24N$.

As matrix $\mathbf{A}$ (equation 3) and its transposed never changes, it is not necessary to calculate the eigenvalues inside the loop at each iteration, we are considering that both are calculated out of the loop. Inside the loop, steps 2 to 4 are repeated two times per iteration. Substituting this result into the CGLS flops count (equation A-2) leads to:

$$f_{ours} = \kappa\, 16N\log_2(4N) + 24N + it\,(\kappa\, 16N\log_2(4N) + 60N). \tag{A-3}$$

## REFERENCES

Aster, R. C., B. Borchers, and C. H. Thurber, 2019, Parameter estimation and inverse problems, 3 ed.: Elsevier.

Blakely, R. J., 1996, Potential theory in gravity and magnetic applications, 1 ed.: Cambridge University Press.

Bott, M., 1967, Solution of the linear inverse problem in magnetic interpretation with application to oceanic magnetic anomalies: Geophysical Journal International, **13**, 313–323.

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

Dampney, C. N. G., 1969, The equivalent source technique: Geophysics, **34**, no. 1, 39–53.

Danes, Z., 1961, Structure calculations from gravity data and density logs: Mining Trans., 223 C.

Dean, W. C., 1958, Frequency analysis for gravity and magnetic interpretation: Geophysics, **23**, 97–127.

Emilia, D. A., 1973, Equivalent sources used as an analytic base for processing total magnetic field profiles: Geophysics, **38**, no. 2, 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.

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

Gunn, P. J., 1975, Linear transformations of gravity and magnetic fields: Geophysical Prospecting, **23**, 300–312.

Hansen, R. O., and Y. Miyazaki, 1984, Continuation of potential fields between arbitrary surfaces: Geophysics, **49**, no. 6, 787–795.

Hogue, J. D., R. A. Renaut, and S. Vatankhah, 2020, A tutorial and open source software for the efficient evaluation of gravity and magnetic kernels: Computers & Geosciences, **144**, 104575.

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**, no. 4, 497–507.

Levinson, N., 1946, The wiener (root mean square) error criterion in filter design and prediction: Journal of Mathematics and Physics, **25**, no. 1-4, 261–278.

Li, Y., M. Nabighian, and D. W. Oldenburg, 2014, Using an equivalent source with positivity for low-latitude reduction to the pole without striation: Geophysics, **79**, J81–J90.

Li, Y., and D. W. Oldenburg, 2010, Rapid construction of equivalent sources using wavelets: Geophysics, **75**, no. 3, L51–L59.

Oliveira Jr., V. C., V. C. F. Barbosa, and L. Uieda, 2013, Polynomial equivalent layer: Geophysics, **78**, no. 1, G1–G13.

Peters, L. J., 1949, The direct approach to magnetic interpretation and its practical application: Geophysics, **14**, 290–320.

Renaut, R. A., J. D. Hogue, S. Vatankhah, and S. Liu, 2020, A fast methodology for large-scale focusing inversion of gravity and magnetic data using the structured model matrix and the 2-d fast fourier transform: Geophysical Journal International, **223**, 1378–1397.

Silva, J. B. C., 1986, Reduction to the pole as an inverse problem and its application to low-latitude anomalies: Geophysics, **51**, no. 2, 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**, no. 4, G57–G69.

Sun, S., C. Chen, and Y. Liu, 2019, Constrained 3d inversion of magnetic data with structural orientation and borehole lithology: A case study in the macheng iron deposit, hebei, china: Geophysics, **84**, B121–B133.

Takahashi, D., V. C. Oliveira Jr, and V. C. Barbosa, 2020, Convolutional equivalent layer for gravity data processing: Geophysics, **85**, G129–G141.

Trench, W. F., 1964, An algorithm for the inversion of finite Toeplitz matrices: Journal of the Society for Industrial and Applied Mathematics, **12**, no. 3, 515–522.

Van Loan, C., 1992, Computational frameworks for the Fast Fourier Transform: SIAM.

Zhang, Y., and Y. S. Wong, 2015, BTTB-based numerical schemes for three-dimensional gravity field inversion: Geophysical Journal International, **203**, no. 1, 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 TABLES

| $N$ | **Matrix A** | **All six first columns of BCCB matrices** | **Matrix L** |
|---|---|---|---|
| 100 | 0.0763 | 0.0183 | 0.00610 |
| 400 | 1.22 | 0.0744 | 0.0248 |
| $2,500$ | 48 | 0.458 | 0.1528 |
| $10,000$ | 763 | 1.831 | 0.6104 |
| $40,000$ | 12,207 | 7.32 | 2.4416 |
| $250,000$ | 476,837 | 45.768 | 15.3 |
| $500,000$ | 1,907,349 | 91.56 | 30.518 |
| $1,000,000$ | 7,629,395 | 183.096 | 61.035 |

Table 1: This table shows the RAM memory usage (in Megabytes) for storing the whole matrix $\mathbf{A}$ (equation 3), the sum of all six first columns of the BCCB matrices embedded from the components of the matrix $\mathbf{H}$ from equation **??** (both need 8 bytes per element) and the matrix $\mathbf{L}$ containing the eigenvalues complex numbers (16 bytes per element) resulting from the diagonalization of matrix $\mathbf{C}$ (equation 62). Here we must consider that $N$ observation points forms a $N \times N$ matrix.