APPLIED GEOPHYSICS, Vol.14, No.2 (June 2017), P. 301–313, 9 Figures.
DOI:10.1007/s11770-017-0625-x

Improved preconditioned conjugate gradient algorithm and application in 3D inversion of gravity-gradiometry data*

Wang Tai-Han♦[1], Huang Da-Nian[1], Ma Guo-Qing[1], Meng Zhao-Hai[2], and Li Ye[3]

Abstract: With the continuous development of full tensor gradiometer (FTG) measurement techniques, three-dimensional (3D) inversion of FTG data is becoming increasingly used in oil and gas exploration. In the fast processing and interpretation of large-scale high-precision data, the use of the graphics processing unit process unit (GPU) and preconditioning methods are very important in the data inversion. In this paper, an improved preconditioned conjugate gradient algorithm is proposed by combining the symmetric successive over-relaxation (SSOR) technique and the incomplete Choleksy decomposition conjugate gradient algorithm (ICCG). Since preparing the preconditioner requires extra time, a parallel implement based on GPU is proposed. The improved method is then applied in the inversion of noise-contaminated synthetic data to prove its adaptability in the inversion of 3D FTG data. Results show that the parallel SSOR-ICCG algorithm based on NVIDIA Tesla C2050 GPU achieves a speedup of approximately 25 times that of a serial program using a 2.0 GHz Central Processing Unit (CPU). Real airborne gravity-gradiometry data from Vinton salt dome (south-west Louisiana, USA) are also considered. Good results are obtained, which verifies the efficiency and feasibility of the proposed parallel method in fast inversion of 3D FTG data.
Keywords: Full Tensor Gravity Gradiometry (FTG), ICCG method, conjugate gradient algorithm, gravity-gradiometry data inversion, CPU and GPU

Introduction

With the dramatic development of airborne geophysical equipment and technology, gravity gradiometry has become a method of exploration used in mineral, oil, and gas exploration in recent years (Bell et al., 1997). Compared with ground gravity methods, airborne full tensor gravity gradiometer measurements have the advantage of providing fast speeds with high efficiency. In addition, they are not affected by conditions within the survey area or terrain relief. Gravity gradient

Manuscript received by the Editor September 21, 2016; revised manuscript received April 27, 2017.
*This research was supported by the Sub-project of National Science and Technology Major Project of China (No. 2016ZX05027-002-003), the National Natural Science Foundation of China (No. 41404089), the State Key Program of National Natural Science of China (No. 41430322) and the National Basic Research Program of China (973 Program) (No. 2015CB45300).
1. College of Geo-Exploration Science and Technology, Jilin University, Changchun 130021, China.
2. Tianjin Navigation Instrument Research Institute, Tianjin 300131, China.
3. Jilin Provincial Electric Power Survey and Design Institute, Changchun 130021, China.
♦Corresponding author: Wang Tai-han (Email: thwang15@jlu.edu.cn)

measurements are more affected by near-surface density variations, because the gradient of the gravity field decays with inverse distance cubed, while the decay of the vertical gravity field is inverse distance squared. Compared with gravity data, the gradient of the gravity field can reflect slight changes caused by changes in the underground density, and it also has a higher resolution, thereby improving the accuracy of geological interpretation. Based on the development of fast mobile platform exploration technology (airborne and shipborne), the volume of gravity gradient data is increasing, and the accuracy of measurements are becoming more reliable.

However, it is well known that inversion of gravity gradient data is often ill-posed, and it requires certain prior information and constrains to guarantee results that are unique and stable. Based on regularization methods, scholars have attempted to overcome the non-uniqueness problem. For example, Li and Oldenburg (1996, 1998) adopted the depth-weighing function of the objective function in magnetic susceptibility inversion and density inversion. An effective way of counteracting the inherent decay of kernel function ("skin effect") is to provide more weight to rectangular prisms as depth increases. In this respect, Portniaguine and Zhdanov (2002) proposed the focusing inversion to invert magnetic susceptibility by using the gradient of the model parameter., and a three-dimensional gravity inversion algorithm was proposed by combining Lagrange's formula with multiple constraints by Boulanger and Chouteau (2001). In addition, Tontini et al. (2006) developed an inversion algorithm for magnetic data by inserting a depth-weighing function into model regularization function, and Shamsipour et al. (2010) adapted the cokriging method to gravity data inversion. Furthermore, Qin et al. (2016) used additional information in the inversion by introducing the spatial gradient weighing function.

Full tensor graviometer (FTG) data provide six times the amount of gravity anomaly data for the same survey conditions, and therefore, the solution of large-scale 3D gravity gradient data inversion necessitates that two obstacles are tackled. Firstly, sufficient computer memory is required to store the matrices, and secondly, there needs to be adequate computational time for matrix-vector multiplications to solve the central system of equations (Čuma and Zhdanov, 2014; Hou et al., 2015). Pilkington (1997) introduced the conjugated gradient (CG) method for three-dimensional magnetic susceptibility inversion in order to save computing time and storage space. With a suitable preconditioner, preconditioned conjugated gradient (PCG) has proven its efficiency and increasing performance in a wide range of geophysical applications (Smith, 1985; Canning and Scholl, 1996; Pilkington, 1997; Chen et al., 2000, 2002). However, although the preconditioned method can improve the efficiency of inversion and reduce the number of inversion iterations (thus reducing the computation time), additional computational time is required to prepare the preconditioner. Computational efficiency is determined by both the iteration number and total computing time, and the time taken for iteration inversion increases significantly when the order of the coefficient matrix is large. Therefore, using a computer to achieve the parallel method is an effective way of improving the speed of the solution.

With the continual increase in computing power and programmability of the graph process unit (GPU), the use of GPU for calculations has become the focus of a considerable amount of research. CUDA (Compute Unified Device Architecture) is a general-purpose parallel computing architecture from NVIDIA Corp. (NVIDIA, 2007), which is a leading manufacturer of GPUs that made the first GPU-only graphic rendering tool for massively parallel computing. CUDA is one of the most suitable means of parallel computing for applications, and it has been successfully used in many potential computing fields (Moorkamp et al., 2010; Chen et al., 2012; Čuma and Zhdanov, 2014; Hou et al., 2015).

In the present paper, we develop an improved incomplete Cholesky preconditioned method to integrate the symmetric successive over-relaxation (SSOR) technique and the incomplete Cholesky decomposition conjugate gradient algorithm (ICCG). The new method is tested on a single model with different grid sizes to illustrate GPU acceleration in conjugate gradient inversion for large-scale data, and the algorithm flow of the parallel SSOR-ICCG method based on GPU is then provided. Results and acceleration ratio are provided for a synthetic noisy model test, and differences between our algorithm and the conventional CG algorithm in 3D inversion of FTG data are discussed. Finally, the approach is applied for inversion of real FTG data. In addition, the applicability of the improved parallel preconditioned algorithm for 3D fast inversion of FTG data is verified by a comparison with the conventional method and with results of previous studies.

Forward modeling and inversion of full tensor gradiometer (FTG) data

Forward modeling

In the three-dimensional (3D) Cartesian coordinate system, the gravitational potential, V, can be calculated by the underground density distribution, ρ, based on Newton's law. The three components of gravity anomalies can then be obtained by derivation of the gravitational potential (Blakely, 1995),

$$\mathbf{g} = (\frac{\partial V}{\partial x}, \frac{\partial V}{\partial y}, \frac{\partial V}{\partial z})^T. \quad (1)$$

The nine gravity gradient components, namely the full tensor gravity gradient (FTG), are the three-directions of the gravity components and are written in a matrix form (Forsberg, 1984),

$$\mathbf{T} = \begin{bmatrix} \frac{\partial^2 V}{\partial x^2}, & \frac{\partial^2 V}{\partial xy}, & \frac{\partial^2 V}{\partial xz} \\ \frac{\partial^2 V}{\partial yx}, & \frac{\partial^2 V}{\partial y^2}, & \frac{\partial^2 V}{\partial yz} \\ \frac{\partial^2 V}{\partial zx}, & \frac{\partial^2 V}{\partial zy}, & \frac{\partial^2 V}{\partial z^2} \end{bmatrix} = \begin{bmatrix} T_{xx} & T_{xy} & T_{xz} \\ T_{yx} & T_{yy} & T_{yz} \\ T_{zx} & T_{zy} & T_{zz} \end{bmatrix}. \quad (2)$$

As the gravity gradient tensor is a symmetric tensor, i.e., $T_{xy} = T_{yx}$, $T_{xz} = T_{zx}$, $T_{yz} = T_{zy}$, and it satisfies the Laplace equation $T_{xx} + T_{yy} + T_{zz} = 0$, there are only five independent components of the gravity gradient tensor (FTG).

Generally, the 3D subsurface domain is divided into a finite number of rectangular cells with a constant density for calculating the gravity anomaly and gravity gradient tensor (discretization). We use the Cartesian coordinate system, where the x–axis and y-axis point in horizontal directions and the z–axis points downwards. Following the formula of Haáz (1953), gravity data produced at a point of coordinates $O = (x_0, y_0, z_0)$ by a single rectangular cell with constant density, ρ, is (Li and Chouteau, 1998)

$$g = -G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \cdot \left[x_i \ln(y_j + r_{ijk}) + y_i ln(x_j + r_{ijk}) - z_k \arctan(\frac{x_i y_j}{z_k r_{ijk}}) \right]. \quad (3)$$

The gravity gradient tensor equation is defined as

$$T_{xx} = G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \tan^{-1} \frac{y_j z_k}{x_i r_{ijk}},$$

$$T_{yy} = G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \tan^{-1} \frac{x_i z_k}{y_j r_{ijk}},$$

$$T_{zz} = G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \tan^{-1} \frac{x_i y_j}{z_k r_{ijk}},$$

$$T_{xy} = -G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \ln[r_{ijk} + z_k],$$

$$T_{yz} = -G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \ln[r_{ijk} + x_i],$$

$$T_{xz} = -G\rho \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} u_{ijk} \ln[r_{ijk} + y_j], \quad (4)$$

where

$$x_i = x_0 - \xi_i,\ y_j = y_0 - \eta_j,\ z_k = z_0 - \zeta_k,\ i, j, k = 1, 2,$$

$$r_{ijk} = \sqrt{x_i^2 + y_j^2 + z_k^2},\ u_{ijk} = (-1)^i (-1)^j (-1)^k,$$

and G is Newton's gravitational constant, and ξ, η, ζ define the eight corners of one rectangular cell. Considering the linear relationship between the gravity gradient tensor and density, equation (4) can thus be expressed in a matrix form as

$$\mathbf{T}_{6n\times 1} = \mathbf{A}_{6n\times m}\rho_{m\times 1} \Leftrightarrow \begin{bmatrix} T_{xx} \\ T_{xy} \\ T_{xz} \\ T_{yy} \\ T_{yz} \\ T_{zz} \end{bmatrix} = \begin{bmatrix} A_{xx} \\ A_{xy} \\ A_{xz} \\ A_{yy} \\ A_{yz} \\ A_{zz} \end{bmatrix} \rho, \quad (5)$$

where $\mathbf{T}$ is a vector of the six gravity gradient tensor data at n observation points; ρ is a density vector of the order m; and $\mathbf{A}$ is the matrix of the geometric terms.

Inversion method

When conducting the 3D inversion of gravity gradient data, we use the observed gravity gradient data, $\mathbf{T}$, to recover the unknown density distribution, ρ. The inversion problem is usually under-determined with infinite and unstable solutions, and therefore regularization should be introduced to ameliorate instability in the inversion process (Čuma and Zhdanov, 2014). We adopt the 3D gravity inversion method introduced by Li and Oldenburg (1998) to minimize the Tikhnov parametric functional as

$$\min \quad \varphi^a(\rho) = \varphi_d(\rho) + \alpha\varphi_s(\rho) = \left\|\mathbf{W_d A}\rho - \mathbf{W_d T}\right\|_2^2 + \alpha\left\|\mathbf{W_z}(\rho)\right\|_2^2. \quad (6)$$

In equation (6), $\varphi_s(\rho)$ is the stabilizing functional, $\varphi_d(\rho)$ is the data-misfit functional of measured and inversion predicted data, and α is the regularizing parameter, which makes a big difference to the gravity inversion results. In addition, $\mathbf{W_d}$ is the diagonal data-misfit weighing matrix normalizing self-respective standard deviation, Wz is a depth-weighing function of the form $\mathbf{W_z} = (\mathbf{z} + \mathbf{z}_0)^{-3/2}$ used to counteract the inherent decay of the kernel function, $\mathbf{A}$, and prevent most of the density from occurring close to the surface, $\mathbf{z}$ is the average rectangular cell depth, and $\mathbf{z}_0$ depends on the block size of model discretization and the observation height of the data.

The depth-weighing function was first proposed by Li and Oldenburg (1996) and has been well applied in many geophysical inversion algorithms (Boulanger and Chouteau, 2001; Caratori Toniti et al., 2006; Cella and Fedi, 2011). We use the adaptive regularization, which decreases the stabilizer contribution as the inversion approaches the converged result. The value of α after the first iteration is expressed as (Zhdanov, 2002)

$$\alpha_1 = \varphi_d \Big/ \varphi_s, \quad (7)$$

and during the inversion process the value for α is determined by decreasing α to q times that of its previous value, using the expression,

$$\alpha_k = q\alpha_1,\ q = \frac{1}{2^{k-1}},\ k = 2,\ 3,\ \ldots,\ n. \quad (8)$$

Equation (8) can then be rewritten as

$$\begin{bmatrix} \mathbf{W_d A} \\ \sqrt{\alpha}\mathbf{W_z} \end{bmatrix}[\rho] = \begin{bmatrix} \mathbf{W_d T} \\ 0 \end{bmatrix} \Leftrightarrow \mathbf{M}\rho = \mathbf{b}. \quad (9)$$

To solve a large-scale linear system such as that of equation 9, use of the conjugate gradient method is beneficial as it is a fast and effective iterative algorithm. With only matrix-vector multiplication and inner vector products, the CG method does not need to continue with the inverse operation of the matrix, and it is therefore widely used in geophysical inversion. We improve the gravity inversion algorithm proposed by Pilkington (1997) and obtain the full tensor gravity gradient inversion algorithm as follows,

for $k = 0;\ \rho_0 = 0$ *and* $\mathbf{r}_0 = \mathbf{M}^{\mathrm{T}}(\mathbf{b} - \mathbf{M}\rho_0)$
while $\mathbf{r}_0 \neq 0$
$k = k + 1,$
if $k = 1,$
$\mathbf{p}_1 = \mathbf{r}_0$
else $\beta_k = \mathbf{r}_{k-1}^{\mathrm{T}}\mathbf{r}_{k-1} / \mathbf{r}_{k-2}^{\mathrm{T}}\mathbf{r}_{k-2},$
$\mathbf{p}_k = \mathbf{r}_{k-1} + \beta_k\,\mathbf{p}_{k-1},$
end
$\mathbf{q}_{\mathrm{k}} = \mathbf{M}\mathbf{p}_k,$
$a_k = \mathbf{r}_{k-1}^{\mathrm{T}}\mathbf{r}_{k-1} / \mathbf{q}_k{}^{\mathrm{T}}\mathbf{q}_k,$
$\rho_k = \rho_{k-1} + a_k\mathbf{p}_k,$
$\mathbf{r}_k = \mathbf{r}_{k-1} - a_k\mathbf{M}^{\mathrm{T}}\mathbf{q}_k,$
end.

From this algorithm, it is possible to easily find the inverse of matrix, $\mathbf{M}$, that has not been calculated, which saves a considerable amount of time and requires minimal storage space. However, when the domain of the data obtained grows, the number of discretized cells also increases. In practical problems, the iteration number and computational cost needs to be considered in 3D FTG data inversion. In addition, for density inversion, the condition number of the coefficient matrix is large, and the convergence speed of the inversion is controlled by the condition number. To reduce the iteration number, many researchers have accelerated the rate of convergence by proposing preconditioners to cluster the eigenvalues in a few small intervals (Smith, 1985; Canning and Scholl, 1996; Pilkington, 1997; Chen et al., 2000, 2002). The method used to improve the number of conditions of the system is called the PCG method, and the algorithm is as follows

for $k = 0;\ \rho_0 = 0$ *and* $\mathbf{r}_0 = \mathbf{M}^{\mathrm{T}}(\mathbf{b} - \mathbf{M}\rho_0)$
while $\mathbf{r}_0 \neq 0$
$\mathbf{z}_{\mathrm{k}} = \mathbf{P}\mathbf{r}$
$k = k + 1,$
if $k = 1,$
$\mathbf{p}_1 = \mathbf{z}_0$
else $\beta_k = \mathbf{r}_{k-1}^{\mathrm{T}}\mathbf{r}_{k-1} / \mathbf{r}_{k-2}^{\mathrm{T}}\mathbf{z}_{k-2},$
$\mathbf{p}_k = \mathbf{z}_{k-1} + \beta_k\,\mathbf{p}_{k-1},$

$$
\begin{aligned}
&\textit{end}\\
&\mathbf{q}_k = \mathbf{M}\mathbf{p}_k,\\
&a_k = \mathbf{r}_{k\text{-}1}^{\mathrm{T}}\mathbf{z}_{k\text{-}1} / \mathbf{q}_k{}^{\mathrm{T}}\mathbf{q}_k,\\
&\rho_k = \rho_{k\text{-}1} + a_k\mathbf{p}_k,\\
&\mathbf{r}_k = \mathbf{r}_{k\text{-}1} - a_k\mathbf{M}^{\mathrm{T}}\mathbf{q}_k,\\
&\textit{end},
\end{aligned}
$$

where **P** is the preconditioner matrix used to accelerate the CG inverse algorithm.

$$\mathbf{M}^{\mathrm{T}}\mathbf{M}\rho = \mathbf{M}^{\mathrm{T}}\mathbf{b}. \tag{10}$$

In the inversion of FTG data, the kernel function matrix, **M**, is relatively large, and its computation and storage require a large amount of time and space, respectively. To reduce the iteration number and improve the convergence rate, equation (10) above can be rewritten as

$$\mathbf{P}\mathbf{M}^{\mathrm{T}}\mathbf{M}\rho = \mathbf{P}\mathbf{M}^{\mathrm{T}}\mathbf{b}. \tag{11}$$

Under ideal conditions, **P** is an approximation of $(\mathbf{M}^{\mathrm{T}}\mathbf{M})^{-1}$, so that $\mathbf{P}\mathbf{M}^{\mathrm{T}}\mathbf{M}$ is approximately equal to the identity matrix. Compared with the original symmetric positive definite matrix, $\mathbf{M}^{\mathrm{T}}\mathbf{M}$, the eigenvalues of PMTM are more clustered; therefore, the condition number is reduced and the convergence rate of the CG method is accelerated. Obviously, it is very important to choose an effective preconditioner to reduce the iteration number of 3D FTG data inversion.

Incomplete Cholesky (IC) decomposition is a kind of common preconditioned method. In equation (10), the coefficient matrix $\mathbf{M}^{\mathrm{T}}\mathbf{M}$ is a symmetric positive definite matrix, and the preconditioner can be prepared with the IC method as,

$$\mathbf{M}^{\mathrm{T}}\mathbf{M} = \mathbf{L}\mathbf{L}^{\mathrm{T}} - \mathbf{R}, \tag{12}$$

where **L** is the sparse lower triangular matrix, and **R** is the residual matrix. Therefore, the preconditioner $\mathbf{P} = (\mathbf{L}\mathbf{L}^{\mathrm{T}})^{-1}$.

Golub and Van Loan (1996) demonstrated that the ICCG PCG algorithm can reduce the number of iterations in the inversion and hence improve the convergence rate. Moreover, the decomposition of the preconditioned matrix directly influences the convergence and speed of the method,

Another approximate inverse of the coefficient matrix $\mathbf{M}^{\mathrm{T}}\mathbf{M}$ can be obtained using SSOR. We assume that the matrix $\mathbf{M}^{\mathrm{T}}\mathbf{M}$ is decomposed as follows (Sajo-Castelli et al., 2014),

$$\mathbf{M}^{\mathrm{T}}\mathbf{M} = \mathbf{D} - \mathbf{L}_{\mathrm{AA}} - \mathbf{L}_{\mathrm{AA}}{}^{\mathrm{T}},$$

$$\mathbf{L} = \frac{(\mathbf{D} - w\mathbf{L}_{\mathrm{AA}})\mathbf{D}^{-1/2}}{\sqrt{w(2-w)}},\ \mathbf{L}^{\mathrm{T}} = \frac{\mathbf{D}^{-1/2}(\mathbf{D} - w\mathbf{L}_{\mathrm{AA}}{}^{\mathrm{T}})}{\sqrt{w(2-w)}}, \tag{13}$$

where **D** is the diagonal matrix of diagonal elements of MTM, and LAA is the negative lower triangular matrix of MTM. ω ($0 < \omega < 2$) is an important parameter, and it has a significant influence on the convergence and result of CG inversion. However, there is no good choice for the value of ω, and we therefore choose it empirically. In addition, the successive over-relaxation (SSOR) preconditioner is defined by

$$\mathbf{P} = \mathbf{L}\mathbf{L}^{\mathrm{T}} = \frac{(\mathbf{D} - w\mathbf{L}_{\mathrm{AA}})\mathbf{D}^{-1}(\mathbf{D} - w\mathbf{L}_{\mathrm{AA}}{}^{\mathrm{T}})}{w(2-w)},\ (0< w < 2). \tag{14}$$

We combine the SSOR decomposition method with the ICCG, which means that the decomposition result obtained by equation (13) is brought into the conventional ICCG algorithm, and the improved SSOR-ICCG algorithm is obtained. The iterative process is as follows,

$$
\begin{aligned}
&\textit{for } k = 0;\ \rho_0 = 0 \textit{ and } \mathbf{r}_0 = (\mathbf{M}^{\mathrm{T}}\mathbf{M})^{\mathrm{T}}(\mathbf{b} - (\mathbf{M}^{\mathrm{T}}\mathbf{M})\rho_0)\\
&\mathbf{rr}_0 = \mathbf{L}^{-1}\mathbf{r}_0,\ \mathbf{p}_0 = (\mathbf{L}^{\mathrm{T}})^{-1}\mathbf{rr}_0\\
&\textit{while } \mathbf{r}_0 \neq 0\\
&\qquad k = k + 1,\\
&\qquad \textit{if } k = 1,\\
&\qquad\quad \mathbf{p}_1 = \mathbf{p}_0\\
&\qquad \textit{else}\quad \beta_k = \mathbf{rr}_{k\text{-}1}^{\mathrm{T}}\mathbf{rr}_{k\text{-}1} / \mathbf{rr}_{k\text{-}2}^{\mathrm{T}}\mathbf{rr}_{k\text{-}2},\\
&\qquad\qquad \mathbf{p}_k = (\mathbf{L}^{\mathrm{T}})^{-1}\mathbf{rr}_k + \beta_k\ \mathbf{p}_{k\text{-}1},\\
&\qquad \textit{end}\\
&\qquad a_k = \mathbf{rr}_{k\text{-}1}^{\mathrm{T}}\mathbf{rr}_{k\text{-}1} / ((\mathbf{M}^{\mathrm{T}}\mathbf{M})\mathbf{p}_{k\text{-}1})^{\mathrm{T}}\mathbf{p}_{k\text{-}1},\\
&\qquad \rho_k = \rho_{k\text{-}1} + a_k\mathbf{p}_k,\\
&\qquad \mathbf{rr}_k = (\mathbf{L}^{\mathrm{T}})^{-1}\mathbf{rr}_{k\text{-}1} - a_k\mathbf{L}^{-1}(\mathbf{M}^{\mathrm{T}}\mathbf{M})\mathbf{q}_k.\\
&\qquad \textit{end}.
\end{aligned}
$$

This method contains a greater amount of information about the coefficient matrix than the conventional ICCG algorithm, and thus more subsurface geophysical information can be used in the 3D gravity gradient inversion to reduce the number of iterations and speedup the convergence speed of inversion.

However, due to the huge amount of 3D FTG data, the inversion iterations consume a large amount of computer memory and a long computational time. In this respect, research has been focused on high-performance parallel computing using a CPU (Central Processing Unit) and GPU (Graphics Processing Unit) to enable the fast processing of a large amount of high-precision data.

Parallel algorithm using graphics processing unit (GPU)

GPU and CUDA

A graphics processing unit (GPU) has the advantages of providing a high bandwidth and strong computing power, and it enables highly parallel calculations compared to CPU computing. However, as it was originally used to complete image rendering, the early GPU did not enable programmability (Zhang, 2009). In recent years, general-purpose computing based on GPU has been widely adopted, and the graphics card maker NVIDIA Corporation have released a unified computing architecture for a general-purpose computing platform (Compute Unified Device Architecture, CUDA). Faced with the demand for large-scale high-precision data processing, the GPU has been successfully applied in a considerable amount of geophysical parallel computing research (Moorkamp et al., 2010; Chen et al., 2012; Liu et al., 2012; Čuma and Zhdanov, 2014). CUDA is a hardware and software co-parallel computing architecture that allows the GPU and CPU to work together, and thus gives full-play to the GPU's high-speed floating-point performance, with the aim of realizing accelerated calculations of complex and time-consuming problems. Initially, CUDA only supported C and C++ language and required a high level of programming (Liu, 2012). With the development of GPU, MATLAB began to support CUDA based on version R2010b (Szymczyk and Szymczyk, 2012) and offered a Parallel Computing Toolbox (PCT) to simplify MATLAB parallel computing. The powerful GPU computing capabilities achieved in MATLAB require the use of the MATLAB of version R2010b, recent CUDA-capable NVIDIA GPUs, such as the NVIDIA Tesla 10-series or 20-series products supporting a compute capability of 1.3 or above.

The main task of this research is designed using MATLAB R2015a, which enables discussion of the algorithm optimization problems found with CPU and GPU.

Algorithmic programming

The conjugate gradient algorithm was run with a regular serial code and execution was timed. The update operations for $\mathbf{r}_0$, α_k, and $\mathbf{r}_k$ per iteration were found to be the most time-consuming. In general, the crucial time-consuming problems in gravity gradient inversion are the matrix-vector multiplications and the inner product about matrix $\mathbf{M}$. Therefore, we computed these with GPU and performed the rest of the inversion steps in sequence. Operation and implementation of the algorithm used the following hardware configuration: the CPU was Intel Xeon E5-2620 6-core 2.0 GHz, 7.2 GT/s, and the GPU was NVIDIA Tesla C2050. To test the GPU acceleration of the CG algorithm, we completed the conjugate gradient inversion of 3D gravity gradient data using model with different grid sizes and recorded the spend time of 500 iterations of CPU serial and CPU/GPU parallel algorithms. As shown in Figure 1, it is

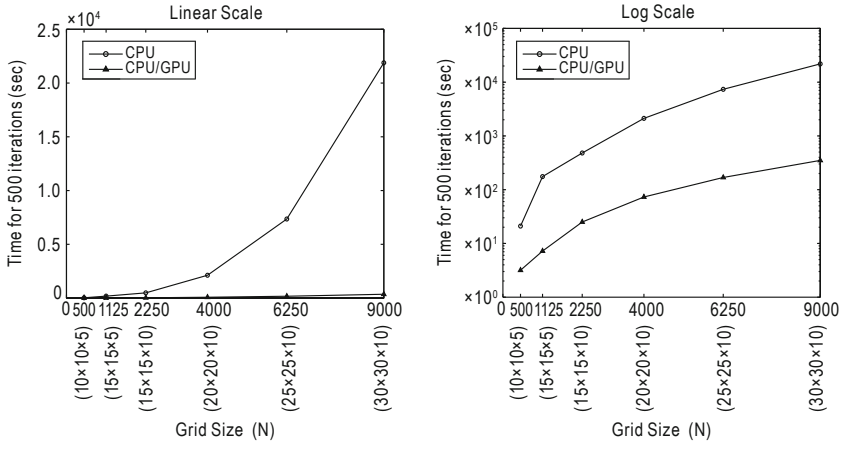

Fig.1 CPU and CPU/GPU computing time for CG algorithm in 3D FTG data inversion; the ordinate on the left is linear and the ordinate on the right is logarithmic.

evident that CPU/GPU computing can largely improve the efficiency, particularly for inverting large volumes of data. We also noticed that the parallel CG algorithm is 62.56 times faster than its CPU implementation for a grid size of 9000.

The above-mentioned PCG can reduce the condition number of the coefficient matrix and the iteration number, which leads to faster convergence and is less time-consuming. However, the computational cost has to be considered with an increase in the amount of data and grid size, since the preconditioner occupies additional memory which needs to be recalculated per iteration. GPU is currently used to accelerate the PCG method in 3D FTG data inversion, not only to reduce iterations but also to solve the time-consuming problem. It is of note that GPU has a limited memory (like CPU). Therefore, for calculations using large-scale data, the storage and removal of the matrix should be considered to avoid exceeding the memory and causing the calculation to stop.

In summary, the computational flow of the PCG parallel algorithm is (Figure 2): (1) initialize the MATLAB environment; (2) load the observed FTG data and start timing; (3) parallel-compute the model weights, $\mathbf{W_m}$, and the depth weighs, $\mathbf{W_z}$, then gather all the results to CPU with the gather () function; (4) initialize ρ_0 and compute $\mathbf{M}$, $\mathbf{b}$; (5) use the gpuArray () function to gather the data to GPU and start iterations, where the preconditioner, $\mathbf{P}$, the conjugated direction, $\mathbf{P}_k$, and the search direction, $\boldsymbol{\alpha}_k$, are computed with GPU while the rest are in serial with CPU; (6) stop timing at the end of the iteration and gather the density result, ρ, to the CPU to plot Figures. The PCT provided by MATLAB provides gpuArray () and gather () as the data transmission and collection functions between CPU and GPU.

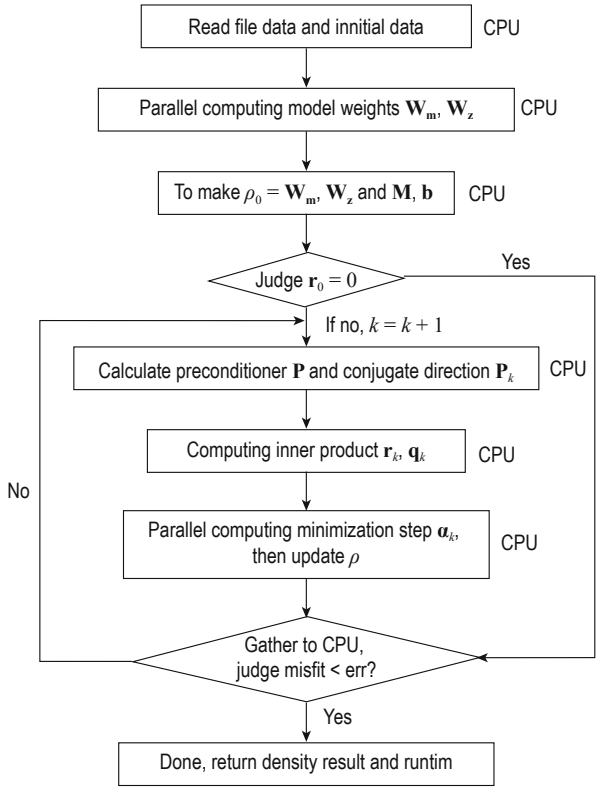

Fig.2 Flowchart of parallel PCG algorithm.

Model test

In the above section, we have illustrated acceleration of the inversion based on GPU using a single model. A complex model consisting of five prism bodies is constructed illustrate the advantages of the improved parallel PCG algorithm in 3D FTG data inversion, and the spatial distribution is shown in Figure 3. The model parameters (including grid size, density, and central depth) are shown in Table 1. The subsurface is divided into 20 × 20 × 10 small rectangular cells with a volume of 100 × 100 × 100 m in the **x**, **y**, and **z** directions. 3D models are arranged underground with different sizes and have a constant residual density at different depths. Random Gaussian noise (10%) is then added to the synthetic gravity gradient data to simulate the real geophysical information. The maximum depth of the underground subspace is 1000 m and 20 × 20 = 400 data observations are distributed at an interval of 100 m. Contour maps of the observed gravity gradient components are shown in Figure 4.

Table 1 Synthetic model parameters

Model number	$x \times y \times z$ dimensions (m)	Depth (m)	Residual density (g/cm^3)
1	700 × 300 × 300	350	0.4
2	300 × 300 × 300	450	0.9
3	200 × 300 × 400	400	0.7
4	300 × 200 × 400	500	1.0
5	300 × 300 × 200	400	0.8

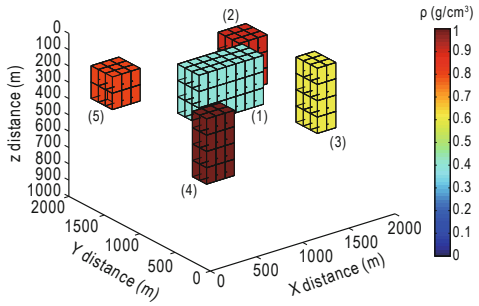

Fig.3 3D distribution of synthetic model.

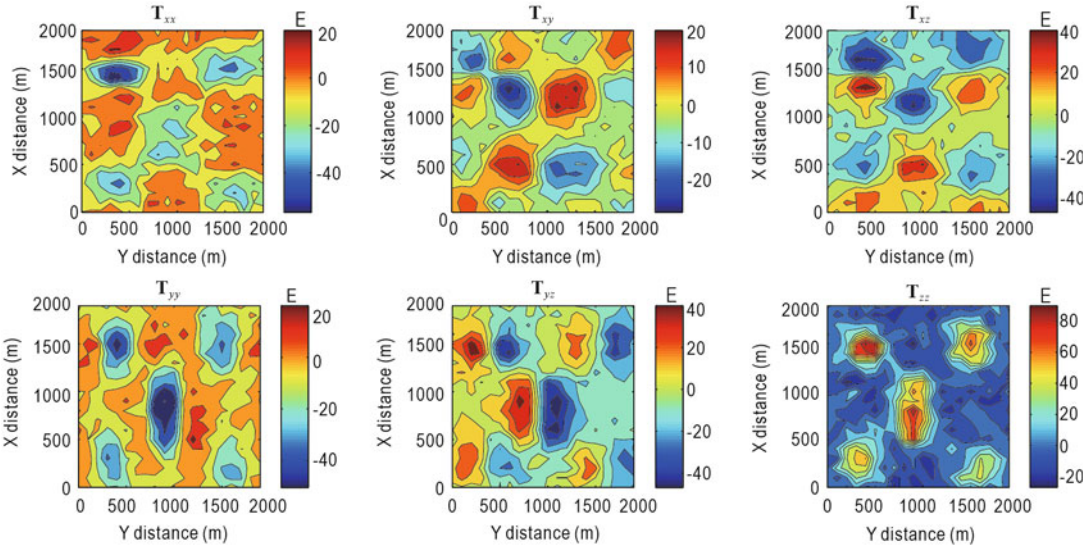

Fig.4 Contour maps of observed gravity gradient components contaminated by 10% Gaussian noise.

To compare the performance and acceleration capability of the proposed algorithm, g_{xx}, g_{xz}, g_{yy}, g_{yz}, g_{xy}, g_{zz} data are selected in the serial CG inversion and parallel SSOR-ICCG preconditioned inversion, respectively. The model gravity anomaly data are then applied to the 3D inversion using our improved

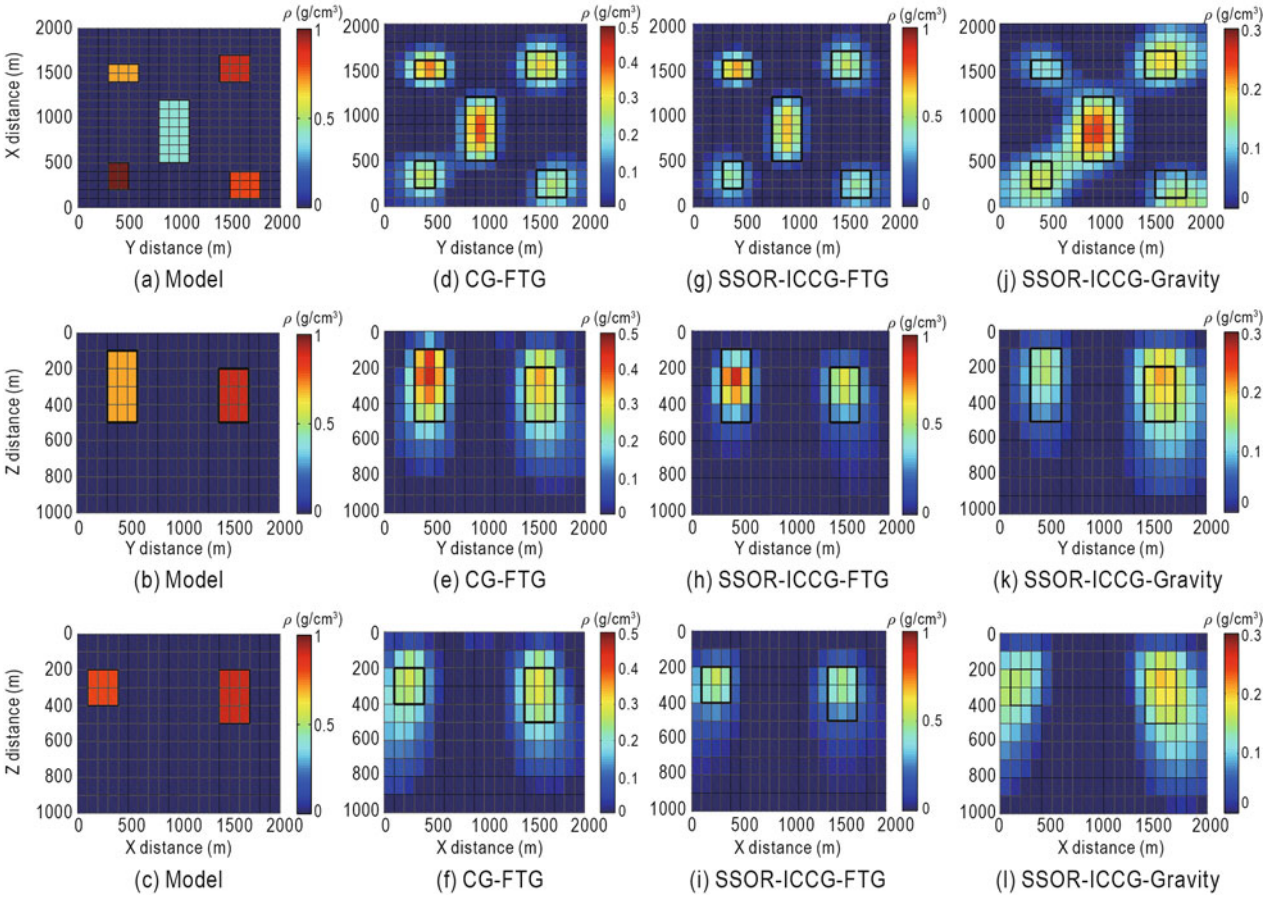

Fig.5 Depth section (a) and vertical sections (b, c) of true model; inversion results of FTG data ((d, e, f) using conventional CG algorithm, and (g, h, i) using SSOR-ICCG algorithm); results of model gravity data (j, k, l) where the true position of the anomaly is shown by the black rectangular border in the figure.

preconditioned algorithm, to demonstrate the advantages of the inversion of FTG data. Figure 5 shows slices of the true model and inversion results of FTG data and gravity data. The depth slices of the recovered model at Z = 400 m are shown in Figures 5d, 5g, and 5k, while vertical sections at a horizontal position of X = 1600 m, Y = 1700 m are shown separately in Figures 5e, 5h, and 5j, and Figures 5f, 5i, and 5l. Results show that the recovered model using the parallel preconditioned algorithm presented in this paper is more reasonable than the traditional CG algorithm, and it is closer to the density value of the true model. A comparison between Figures 5g, 5h, and 5i and Figures 5j, 5k, and 5l shows that the inversion results of FTG data are more compact than the gravity inversion, and the shape of the recovered model is refined. Table 2 shows the standard deviation of Gaussian noise, and the estimated standard deviation of the residual between predicted data and observed data for the two algorithms. It is clear that compared with the conventional conjugate gradient algorithm, the standard deviation of the residual between predicted and observed data obtained from our parallel SSOR-ICCG preconditioned algorithm is closer to the standard deviation of Gaussian noise. Results again quantitatively confirm that inversion results are improved and that they are in line with the more compact recovered models in Figures 5g, 5h, and 5i.

Table 2 Standard deviation (in E) of Gaussian noise and estimated standard deviation of residual between predicted data and observed data for two algorithms

	g_{xx}	g_{xy}	g_{xz}	g_{yy}	g_{yz}	g_{zz}
Noise	5.0324	2.4458	5.1922	4.9766	4.6654	9.4016
CG-CPU	5.7855	2.7372	5.4833	6.0064	4.9914	11.9590
PCG-GPU	5.1792	2.7133	6.0955	5.1804	5.0213	9.5759
PCG (gravity)	7.1574	3.6041	8.1508	7.2017	7.3696	15.8269

In this paper, the residual error curve $(\mathbf{T}\text{-}\mathbf{A}\rho)^{\mathrm{T}}(\mathbf{T}\text{-}\mathbf{A}\rho)/\ \mathrm{N}$ proposed by Pilkington (1997) is adopted to show improvement in the convergence (see Figure 6). Compared with the CG algorithm, there is an evidently faster decrease in the residual error norm of the SSOR-ICCG method, which indicates that the SSOR-ICCG algorithm converges significantly faster. When assuming 500 iterations, the conventional CG algorithm with no preconditioner reaches a minimum residual error of 306, whereas the improved SSOR-ICCG algorithm reaches a minimum value of 213.2. For the same residual error, the iteration number in relation to computing time can be obtained from the black line in Figure 6; at the same residual error of 350, the iteration number of the no-preconditioner CG method is 50, while that of the SSOR-ICCG preconditioned algorithm only requires 8 iterations to achieve this residual. Additional computational time is required to prepare (decompose) the preconditioner, and therefore computational efficiency is determined by both the numbers of iterations and the total computing time used. The GPU time includes computation of the preconditioner in all iterations, which has a considerable time cost in the total runtime. In Table 3, with the same iteration number (500), the total computation time of the SSOR-ICCG algorithm in the CPU serial code is 1644.4

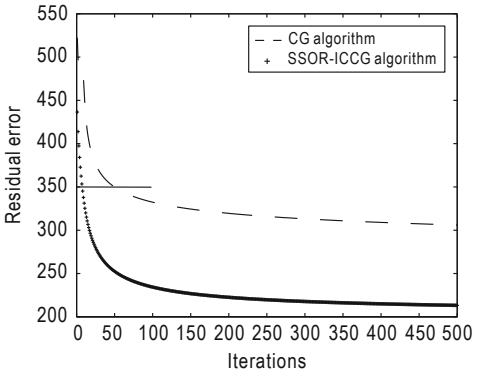

Fig.6 Residual error curve: dashed line represents conventional CG algorithm and plus sign represents the improved SSOR-ICCG algorithm. The number of iterations at the same residual error can be identified by the black horizontal line.

Table 3 Runtime comparison

Method	Time taken to complete 500 iterations	Iteration number for inversion result	Runtime	Speedup
CG-CPU	2224.7 s	50	204.47 s	1.0x
SSOR-ICCG-CPU	1644.4 s	8	46.51 s	9.5x
SSOR-ICCG-GPU	235.5 s	8	8.21 s	24.9x

s, which is 1.4 times faster than the conventional CG algorithm in CPU. However, our GPU-based parallel SSOR-ICCG algorithm only costs 235.5 s, which is a speedup of 9.5x. In fact, for the convergent results with the same residual error, the parallel PCG algorithm (8.66 s) in this paper achieves a speedup of 5.7x compared to its serial code (46.5 s), and the speedup ratio rises to 24.9x compared to the conventional CG algorithm on CPU. This confirms that the parallel preconditioned algorithm presented in this paper can reduce the number of iterations in the 3D inversion of FTG data, thereby reducing the time cost and achieving a fast inversion.

Application to real data

The improved method is applied to real airborne FTG data taken from the Vinton salt dome (in southwest Louisiana, USA), to further illustrate the effectiveness and applicability of our parallel preconditioned algorithm. Data were provided by Bell Geospace Inc., and were measured by Air-FTG from July 3 to July 6, 2008. The survey was conducted at a height of 80 m in a north-south direction; the sampling distance was 50 m; and the flying line spacing was 250 m. This region has been previously studied and presented in papers (Coker et al., 2007; Oliveira Jr and Barbosa, 2013; Geng et al., 2014; Qin et al., 2016), and such results have shown that the anomaly of the salt dome is predominantly caused by the cap-rock with a mean residual density of 0.55 g/cm3 and a depth between 200 and 600 m, approximately. We removed the regional field and chose a 3000 × 3000 m subset in the middle part of the survey area. The inversion domain was discretized into 20 × 20 × 20 = 8000 rectangular cells with a dimension of 150 × 150 × 50 m. Figure 7 shows contour maps of the gravity gradient components over Vinton salt dome.

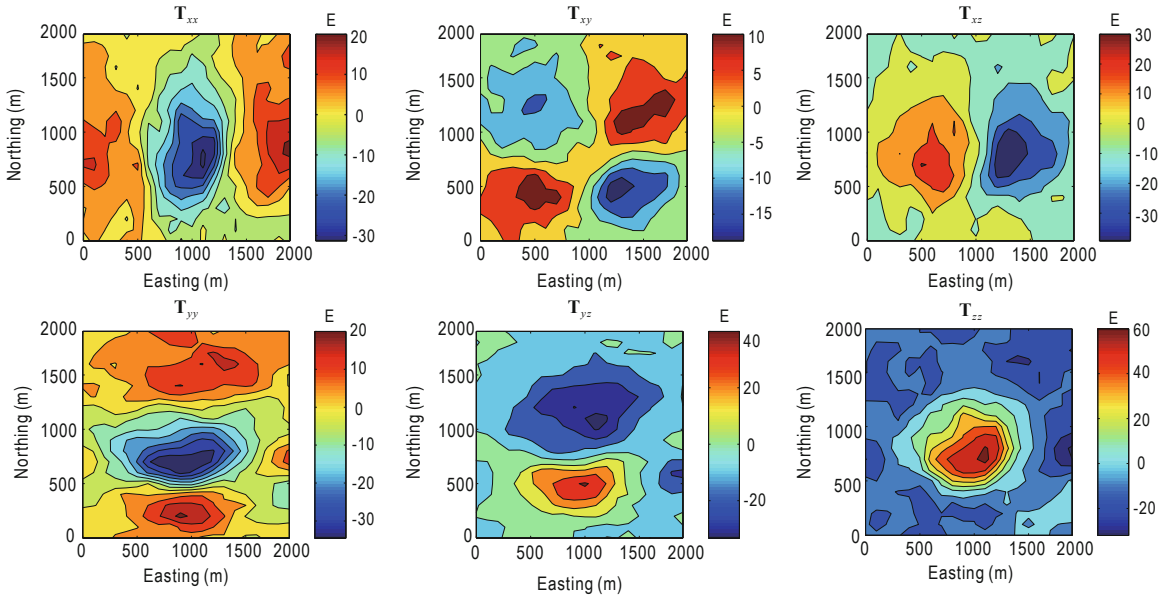

Fig.7 Observed data for Vinton salt dome.

To test our improved GPU-based SSOR-ICCG algorithm with real data, we applied the inversion to six components of the FTG data. According to Ennen and Hall (2011), the depth of the cap-rock is 160 m, with a density of 2.75 g/cm^3; the density of surrounding sediments (shale and sandstone) is 2.2 g/cm^3. In this case, the main anomaly is predominantly caused by the cap-rock, and thus the upper and lower bounds for the residual density are 0 g/cm^3 and 0.55 g/cm^3, respectively. Figure 8 shows the 3D inversion result using our method. Two perpendicular cross-sections through the center of the recovered model are shown in Figure 8a, and Figure 8b shows a volume-rendered image of the 3D density-contrast model. The recovered model clearly indicates that the maximum east-west and north-south lengths of the anomaly are 1100 m and 900 m, respectively, which is in agreement with the result by Thompson and Eichelberger (1928). The southern part of inversion result is about 200 m and it deepens further to the north to reach 400 m (with an elongated form from northeast to southwest), which is consistent with the main fault described in the study by Coker et al. (2007).

The solution for the Euler deconvolution of gravity in this region is shown in Figure 9. The central depth of the interpreted cap-rock in this paper is about 300 m, which is little shallower than Euler deconvolution depths (average depth 400 m), but is close to the result of Thompson and Eichelberger (1928) (average depth of 304.5 m) and the recovered model by Oliveira Jr and Barbosa (2013) (of 335 m), further verifying the reasonableness of the inversion results. In calculation, the total computational time for convergence of the result using the conventional conjugate gradient algorithm was 518 s with 51 iterations. However, applying the parallel SSOR-ICCG preconditioned method to 3D FTG data inversion took only 136 s and used 30 iterations. This result also confirms that the improved method can reduce the number of iterations and reduce the computation time compared with the serial program, thereby achieving a fast inversion when using a real application. Therefore, the improved parallel SSOR-ICCG preconditioned algorithm is shown to be an effective speedup algorithm that provides a feasible method for large-scale fast 3D density inversion of gravity gradient data.

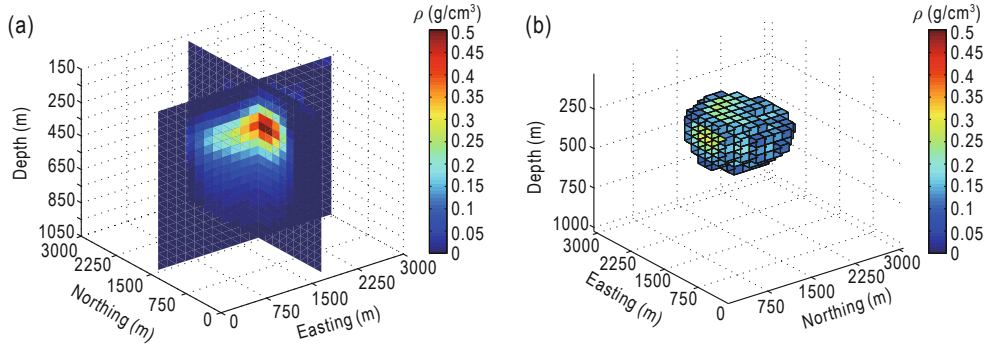

Fig.8 3D inversion result: (a) two perpendicular cross-sections at Northing = 1500 m and Easing = 1800 m, (b) volume-rendered image of 3D density-contrast model with density contrast 0.1 g/cm3 removed. Central depth is approximately 300 m.

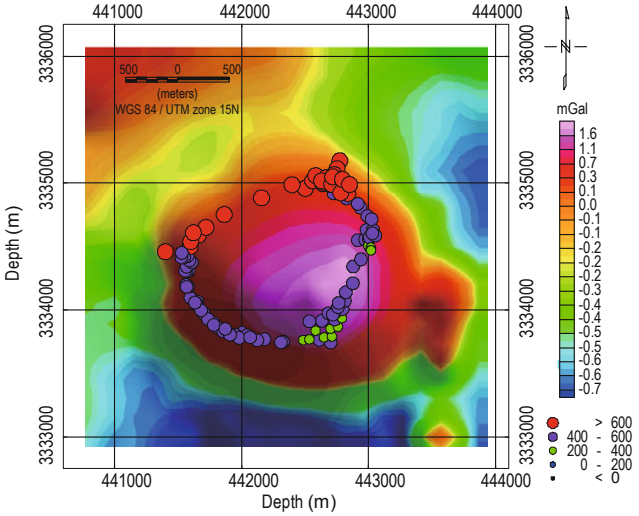

Fig.9 Euler deconvolution results of Vinton gravity data.

Conclusions

In this study, an improved fast preconditioned conjugated gradient (SSOR-ICCG) algorithm is proposed by combining the symmetric successive over-relaxation (SSOR) technique with the incomplete Cholesky decomposition conjugate gradient algorithm (ICCG). The parallel preconditioned algorithm based on GPU is applied to the inversion of 3D full tensor gravity-gradiometry data. One-single model conjugate gradient inversion tests show that GPU can improve the efficiency of inversion for large-scale data. Compared with the conventional CG algorithm, the parallel PCG method not only reduces the iteration number but also negates the need for extra time preparing the preconditioner to achieve a fast inversion.

The inversion results for a synthetic model show that the resolution of FTG inversion is higher than inversion of single gravity data. Compared with the traditional CG method, our algorithm provides a better density value and the geometric position is closer to that of the actual model. The computational efficiency is evaluated by the iteration number and total computing time, and the improved algorithm converges faster and costs less time in three-dimensional inversion. In addition, results are given with a speed approximately 25 times faster than that of the serial code for the CG method on CPU. However, the computation matrix and precondition matrix need to be reasonably allocated between the CPU and GPU to ensure that the video memory does not overflow.

Finally, the method is applied in the 3D inversion of airborne FTG data acquired over the Vinton salt dome (Louisiana, USA). The inversion results are found to be in agreement with geological data and the central depth is close to that determined in previous research. Although the recovered depth is a little shallower than that provided with Euler deconvolution results, but the speed of inversion is improved, which thus verifies the efficiency and applicability of the improved parallel preconditioned algorithm in fast inversion of 3D FTG data.

Acknowledgments

The authors would like to thank Bell Geospace Inc. for providing FTG data from the Vinton salt dome. We also thank the reviewers for their detailed comments and suggestions, which helped to improve the paper.

References

Bell, R. E., Anderson, R., and Pratson, L., 1997, Gravity gradiometry resurfaces: Leading Edge, **16**(1), 55−59.

Blakely, R. J., 1995, Potential Theory in Gravity and Magnetic Applications: Cambridge University Press, Cambridge, UK.

Boulanger, O., and Chouteau, M., 2001, Constraints in 3d gravity inversion: Geophysical Prospecting, **49**(2), 265−280.

Canning, F. X., and Scholl, J. F., 1996, Diagonal preconditioners for the EFIE using a wavelet basis: IEEE Transactions on Antennas & Propagation, **44**(9), 1239−1246.

Cella, F., and Fedi, M., 2011, Inversion of potential field data using the structural index as weighting function rate decay: Geophysical Prospecting, **60**(2), 313−336.

Chen, R. S., Yung, E. K. N., Chan, C. H., and Fang, D. G., 2000, Application of preconditioned CG–FFT technique to method of lines for analysis of the infinite-plane metallic grating: Microwave & Optical Technology Letters, **24**(3), 170−175.

Chen, R. S., Yung, K. N., Chan, C. H., Wang, D. X., and Fang, D. G., 2002, Application of the SSOR preconditioned CG algorithm to the vector fem for 3D full-wave analysis of electromagnetic-field boundary-value problems: IEEE Transactions on Microwave Theory & Techniques, **50**(4), 1165−1172.

Chen, Z., Meng, X., Guo, L., and Liu, G., 2012, GICUDA: a parallel program for 3D correlation imaging of large scale gravity and gravity gradiometry data on graphics processing units with CUDA: Computers & Geosciences, **46**(3), 119−128.

Coker, M. O., Bhattacharya, J. P., and Marfurt, K. J., 2007, Fracture patterns within mudstones on the flanks of a salt dome: Syneresis or slumping?: Gulf Coast Association of Geological Societies Transactions, **57**, 125−137.

Čuma, M., and Zhdanov, M. S., 2014, Massively parallel regularized 3D inversion of potential fields on CPUs and GPUs: Computers & Geosciences, **62**(1), 80−87.

Ennen, C., and Hall, S., 2011, Structural mapping of the Vinton salt dome, Louisiana, using gravity gradiometry data: 81st Annual International Meeting, SEG, Expanded Abstracts, **30**(1), 830−835.

Forsberg R., 1984, A study of terrain reductions, density an omalies and geophysical inversion methods in gravity fie ld modelling: Report 355, Department of Geodetic Scien ce and Surveying, Ohio State University.

Geng, M., Huang, D., Yang, Q., and Liu, Y., 2014, 3D inversion of airborne gravity-gradiometry data using cokriging: Geophysics, **79**(4), G37−G47.

Golub, G. H., and Van Loan, C. F. 1996, Matrix computations (3rd edition.): Johns Hopkins University Press, Baltimore, America.

Haáz, I. B., 1953, Relationship between the potential of the attraction of the mass contained in a finite rectangular prism and its first and second derivatives: Geophysical Transactions, **II**, 57–66

Hou, Z. L., Wei, X. H., Huang D. N., et al., 2015, Full tensor gravity gradiometry data inversion: performance analysis of parallel computing algorithms: Applied Geophysics, **12**(3), 292–302

Li X., and Chouteau M., 1998, Three- dimensional gravity modelling in all space: Survey in Geophysics, **19**(4), 339–368.

Li, Y., and Oldenburg, D. W., 1996, 3-D inversion of magnetic data: Geophysics, **61**(2), 394−408.

Li, Y., and Oldenburg, D. W., 1998, 3-d inversion of gravity data: Geophysics, **63**(1), 109−119.

Liu, G., Meng, X., and Chen, Z., 2012, 3D magnetic inversion based on probability tomography and its GPU implement: Computers & Geosciences, **48**(9), 86−92.

Liu, W., 2012, Parallel program design of Matlab: Beihang University Press, Beijing.

Moorkamp, M., Jegen, M., Roberts, A., and Hobbs, R., 2010, Massively parallel forward modeling of scalar and tensor gravimetry data: Computers & Geosciences, **36**(5), 680−686.

NVIDIA, 2007, NIVIDIA CUDA compute unified device architecture programming guide. Santa Clara, CA.

Oliveira Jr, V. C., and Barbosa, C. F., 2013, 3-D radial

gravity gradient inversion: Geophysical Journal International, **195**(2), 883−902.
Pilkington, M., 1997, 3-D magnetic imaging using conjugate gradients: Geophysics, **62**, 1132−1142.
Portniaguine, O., and Zhdanov, M. S., 2002, 3-D magnetic inversion with data compression and image focusing: Geophysics, **67**(5), 1532−1541.
Sajo-Castelli A M, Fortes M A., and Raydan M., 2014,Preconditioned conjugate gradient method for finding minimal energy surfaces on Powell–Sabin triangulations. Journal of Computational & Applied Mathematics, **268**(1), 34−55.
Qin, P., Huang, D., Yuan, Y., Geng, M., and Liu, J., 2016, Integrated gravity and gravity gradient 3d inversion using the non-linear conjugate gradient: Journal of Applied Geophysics, **126**, 52−73.
Shamsipour, P., Marcotte, D., Chouteau, M., and Keating, P., 2010, 3D stochastic inversion of gravity data using cokriging and cosimulation: Geophysics, **75**(1), I1−I10.
Smith, G. D., 1985, Numerical solution of partial differential equations: finite difference methods: Oxford University Press, England.
Szymczyk, M., and Szymczyk, P., 2012, Matlab and parallel computing: Image Processing & Communications, **17**(4), 207−216.
Thompson, S. A., and Eichelberger, O. H., 1928, Vinton salt dome, Calcasieu Parish, Louisiana: AAPG Bulletin, **12**, 385−394.
Tontini, C. F., Cocchi, L., and Carmisciano, C., 2006, Depth-to-the-bottom optimization for magnetic data inversion: magnetic structure of the latium volcanic region, Italy: Journal of Geophysical Research Atmospheres, **111**(B11), 220−222.
Zhang, S., 2009, GPU high performance computing of CUDA: China Water & Power Press, Beijing.
Zhdanov, M. S., 2002, Geophysical inverse theory and regularization problems: Elsevier, Salt Lake City, USA.

Wang Tai-Han received his B.S. (2013) in Geophysics at the College of Geo-Exploration Science and Technology, Jilin University, and is currently a Ph.D. candidate in Solid Geophysics at the college. His major research interests are in the field of processing and fast inversion of gravity, magnetic, and gradient tensor data.