Journal of Computational and Applied Mechanics, Vol. 4., No. 1., (2003), pp. 5–12

PARALLELIZATION OF AN ALGORITHM FOR SOLVING THE GRAVITY INVERSE PROBLEM

Elena N. Akimova
Institute of Mathematics and Mechanics, Urals Branch of the Russian Academy of Sciences
S. Kovalevskaya Str., 16, Ekaterinburg, 620219, Russia
aen@imm.uran.ru

[Received: August 14, 2002]

Abstract. A parallel algorithm for solving the gravity inverse problem is considered. The corresponding programme has been implemented on the Massively Parallel Computing System MVS–1000.

Mathematical Subject Classification: 65F05, 86A22
Keywords: gravity inverse problem, parallel algorithm

1. Introduction

The three-dimensional gravity inverse problem of finding the interface between mediums from the gravitational data is investigated. A model of the lower half-space consists of the three mediums with constant densities which are divided by the surfaces S_1 and S_2. The gravitational anomaly is formed by the deviation of the desired surface S from the horizontal plane $z = H$ ($H_1 = 2,\ H_2 = 10$ in our case) [1].

2. Main equations and numerical algorithms

The gravity equation with respect to the unknown surface $z = z(x, y)$ is reduced to the two-dimensional nonlinear equation with integral operator

$$B[z] \equiv f\Delta\sigma \iint\limits_a^b \iint\limits_c^d \left\{ \frac{1}{[(x-x')^2 + (y-y')^2 + z^2(x',y')]^{1/2}} - \right.$$
$$\left. - \frac{1}{[(x-x')^2 + (y-y')^2 + H^2]^{1/2}} \right\} dx'dy' = F(x,y), \quad (2.1)$$

where f is the gravitation constant, $\Delta\sigma$ is the density jump on the interface, $F(x, y)$ is the anomalous gravitational field.

For solving the nonlinear integral equation the iterative regularizing Newton method is used

$$z^{k+1} = z^k - [B'(z^k) + \alpha_k I]^{-1}(B(z^k) + \alpha_k z^k - F), \quad (2.2)$$

where $B'(z^k)$ is the Frechet derivative of the operator B in the point z^k for (2.1), I is the identity operator, α_k is a sequence of the positive parameters which are chosen taking into account the right side of equation (2.1).

After discretizing equation (2.1) on the grid $n = M \times N$ and approximating the integral operator B by the quadrature formulas, the problem of the form (2.2) can be written in the form of the system of linear equations with asymmetric and full $n \times n$ matrix and can be solved by the Gauss or Gauss-Jordan elimination algorithms for each iteration of the method (2.2).

We consider the system of n linear equations with n unknowns in the form:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = a_{1n+1} \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = a_{2n+1} \\ ..... \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = a_{nn+1} \end{cases} . \tag{2.3}$$

The main idea of the Gaussian elimination method is to reduce the full matrix A of the system (2.3) to the upper triangular form, that is to obtain the system of equations in the following form:

$$\begin{cases} x_1 + c_{12}x_2 + c_{13}x_3 + ... + c_{1n}x_n = c_{1n+1} \\ x_2 + c_{23}x_3 + ... + c_{2n}x_n = c_{2n+1} \\ ..... \\ x_n = c_{nn+1} \end{cases} . \tag{2.4}$$

From the system (2.4) we find the unknowns by the formulas:

$$x_k = c_{kn+1} - c_{kk+1}x_{k+1} - ... - c_{kn}x_n, \quad k = n, n-1, ..., 1 . \tag{2.5}$$

In the first step of the Gauss elimination method we choose the pivot element (the maximum of modulus) in the first equation of the system (2.3). Let $a_{11} \neq 0$. After dividing all the coefficients and the constant term of the first equation of the system by a_{11} we obtain the following equation:

$$x_1 + c_{12}x_2 + c_{13}x_3 + ... + c_{1n}x_n = c_{1n+1}, \tag{2.6}$$

where $c_{1j} = a_{1j}/a_{11}, \quad j = 2, 3, ..., n+1.$

With the help of equation (2.6) we eliminate the unknown x_1 from the other equations of the system, beginning with the second one. We get the following system:

$$\begin{cases} a^{(1)}_{22}x_2 + a^{(1)}_{23}x_3 + ... + a^{(1)}_{2n}x_n = a^{(1)}_{2n+1} \\ a^{(1)}_{32}x_2 + a^{(1)}_{33}x_3 + ... + a^{(1)}_{3n}x_n = a^{(1)}_{3n+1} \\ ..... \\ a^{(1)}_{n2}x_2 + a^{(1)}_{n3}x_3 + ... + a^{(1)}_{nn}x_n = a^{(1)}_{nn+1} \end{cases} \tag{2.7}$$

where $a^{(1)}_{ij} = a_{ij} - c_{1j}a_{i1}, \quad i = 2, 3, ..., n, \quad j = 2, 3, ..., n+1.$

Continuing the process, in the k-th step we obtain the equation

$$x_k + c_{kk+1}x_{k+1} + ... + c_{kn}x_n = c_{kn+1}, \tag{2.8}$$

where $c_{kj} = a_{kj}^{(k-1)}/a_{kk}^{(k-1)}$, $j = k+1, ..., n+1$, and the system of equations

$$\begin{cases} a_{k+1k+1}^{(k)} x_{k+1} + ... + a_{k+1n}^{(k)} x_n = a_{k+1k+1}^{(k)} \\ ..... \\ a_{nk+1}^{(k)} x_{k+1} + ... + a_{nn}^{(k)} x_n = a_{nn+1}^{(k)} \end{cases}, \tag{2.9}$$

where $a_{ij}^{(k)} = a_{ij}^{(k-1)} - c_{kj} a_{ik}^{(k-1)}$, $i = k+1, ..., n$, $j = k+1, ..., n+1$.

In the last step of the elimination we have the equation $x_n = c_{nn+1}$.

The Gauss-Jordan algorithm is one of the variants of the Gaussian elimination algorithm. In this case the matrix A of system (2.3) is reduced to diagonal form, but not an upper triangular form. In the $(k+1)$-th step the current matrix A_k has the following form:

$$A_k = \begin{bmatrix} 1 & 0 & \dots & 0 & a_{1,k+1}^{(k)} & \dots & a_{1n}^{(k)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & a_{k,k+1}^{(k)} & \dots & a_{kn}^{(k)} \\ 0 & 0 & \dots & 0 & a_{k+1,k+1}^{(k)} & \dots & a_{k+1,n}^{(k)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & a_{n,k+1}^{(k)} & \dots & a_{nn}^{(k)} \end{bmatrix}. \tag{2.10}$$

We divide the $(k+1)$-th row by the coefficient $a_{k+1,k+1}^{(k)}$ and eliminate all off-diagonal elements of the $(k+1)$-th column. We will make this elimination by multiplying the $(k+1)$-th row by $a_{j,k+1}^{(k)}$ and subtracting the result from the j-th row ($j=1, ..., n;\ j \neq k+1$).

To guarantee the numerical stability of the Gauss and Gauss-Jordan algorithms in the general case, a partial choice of the pivot element is necessary. If we take the maximum (with respect to the modulus) element in the k-th row as the pivot element in the k-th step of the elimination, then before the realization of the modification it is necessary to rearrange the k-th column and the column with the pivot element.

3. **Parallel realization**

The parallel realization of the Gauss method for m processors is the following. Conditionally, we divide the vectors z and F into m parts so that $n = m \cdot L$. The matrix A is divided by the horizontal lines into the m blocks, respectively (Figure 1). Assume that the rows of the matrix A with numbers $1, 2, ..., L$ are stored in the memory of the first processor (the Host), the rows with numbers $(L+1), (L+2), ..., 2L$ are stored in the memory of the second processor, the rows with numbers $(2L+1), (2L+2), ..., 3L$ are stored in the memory of the third processor, and so on. The rows with numbers $(m-1)L+1, ..., Lm$ are stored in the memory of the m-th processor. In the first step the Host processor chooses the pivot element among the elements $a_{11}, a_{12}, ..., a_{1n}$ of the first row, modifies the first row and sends it to each of the other processors. After that each processor eliminates the first unknown x_1 of every row from its own part of the L equations. After the first step of the elimination we obtain the matrix with the first column $(1, 0, 0, ..., 0)^T$. In the second step the Host processor sends the modified

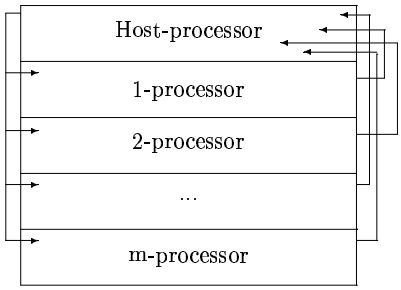

Figure 1. The diagram of the data distribution over the processors

second row to each of the other processors. After that each processor eliminates the second unknown x_2 of every row from its own part of the $(L-1)$ equations, and so on, up to the L-th step. At the end of the L-th step the second processor sends the L-th row to the Host processor. The Host chooses the pivot element, modifies the L-th row and sends it to each of the other processors. After that each processor of the others eliminates the unknown x_{L+1} of every row from its own part of the equations. At the end of the $2L$-th step the third processor sends the $(2L+1)$-th row to the Host processor. The Host chooses the pivot element, modifies the $(2L+1)$-th row and sends it to each of the other processors, and so on. In the last step the Host processor sends the modified last row to each of the other processors.

During the elimination process more and more processors become idle in every step, since the number of the equations is diminished by one. This affects the efficiency of the algorithm. The Host processor works until the end because it responds to the transfer of the modified rows and the choice of the pivot element in every step. To reduce the waiting time the Host processor sends the modified row to other processors immediately after receiving it and makes the calculations with its own part of the equations independently. In the Gauss-Jordan method all the processors make calculations with their own parts until the end. The waiting time decreases and the efficiency of the algorithm increases.

4. **Efficiency**

Parallelization of the basic algorithms and their realization on the Massively Parallel Computing System MVS–1000 [2] are implemented. The analysis of the efficiency of parallelization of the iterative algorithm with different numbers of processors is carried out. MVS–1000/16 of the Research Institute is a KVANT computer consisting of 16 Intel Pentium III-800, 256 MByte, 10 GByte disk, two 100 Mbit network controllers (Digital DS21143 Tulip and Intel PRO/100). The educational computing cluster consists of 8 Intel Pentium III-700, 128 MByte, 14 GByte disk, 100 Mbit network controller 3Com 3c905B Cyclone (Figure 2). The first 15 nodes work fast, the other 8 nodes work more slowly.

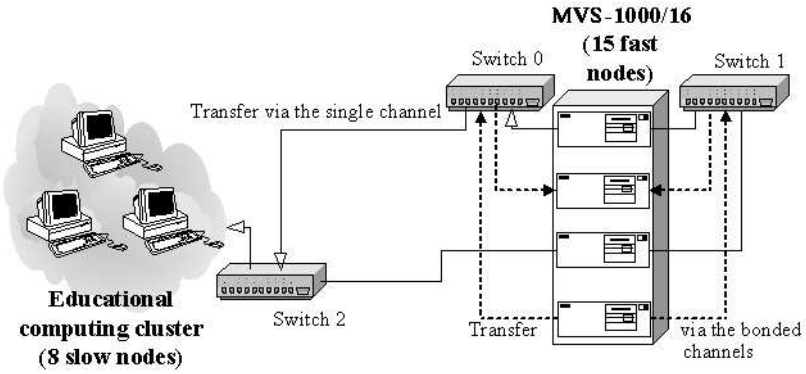

Figure 2. MVS–1000/16

For comparison of the execution times of the sequential and parallel algorithms, we will consider the coefficients of the speed-up and efficiency

$$S_m = T_1/T_m, \qquad E_m = S_m/m,$$

where T_m is the execution time of the parallel algorithm on MVS–1000 with m ($m > 1$) processors, T_1 is the execution time of the sequential algorithm on one processor.

$T_m = T_c + T_o$, where T_c is the computing time, T_o is the exchange time. The number m of processors corresponds to the division of the vectors z and F into m parts mentioned so that $n = m \cdot L$.

Table 1 shows execution times and the coefficients of the speed-up and the efficiency of the iterative regularizing Newton method with 5 iterations with using the parallel and sequential ($m = 1$) Gauss algorithms for problem (2.1)— (2.2) for 111×35 points of the grid domain.

m	T_m, min.	S_m	E_m
1	57.48	—	—
2	46.85	1.23	0.61
3	36.18	1.59	0.53
4	29.38	1.96	0.49
5	25.78	2.23	0.45
6	21.83	2.63	0.44
8	17.25	3.33	0.42
10	16.17	3.55	0.36
12	15.32	3.75	0.31

Table 1. Gauss Method

m	T_m, min.	S_m	E_m
1	114.1	—	—
2	60.50	1.89	0.94
3	42.38	2.69	0.90
4	33.53	3.40	0.85
5	28.48	4.01	0.80
6	23.88	4.78	0.79
8	19.88	5.74	0.72
10	18.45	6.18	0.62
12	17.35	6.58	0.55

Table 2. Gauss-Jordan Method

Table 2 shows execution times and the coefficients of the speed-up and the efficiency of the iterative regularizing Newton method with 5 iterations with using the parallel and sequential ($m = 1$) Gauss-Jordan algorithms for problem (2.1)–(2.2) for 111×35 points of the grid domain.

The results of the calculations show that the parallel Gauss and Gauss-Jordan algorithms have quite a high efficiency of parallelization, and the Gauss-Jordan algorithm efficiency is higher than the efficiency of the Gauss algorithm. In the case of the parallel Gauss algorithm with the number of processors $m \leq 5$, the efficiency is $E_m > 0.45$. In the case of the parallel Gauss-Jordan algorithm with the number of processors $m \leq 10$, the efficiency is $E_m > 0.6$. When the number of processors m is small, then the speed-up S_m increases almost linearly as the number m increases. On the other hand, when m is large, then the exchange time increases, so the efficiency E_m decreases.

5. **Concluding remarks**

The preliminary gravitational data processing is connected with the selection of the anomalous field for each interface S_i $(i = 1, 2)$ from the common data measured on a rectangular area in some region in the Urals. This processing was implemented by the methods from [1]. In Figures 3 and 4 the graph 1 of the boundary profiles of the interfaces S_1 and S_2 is obtained by using the methods from [1] (continuous lines). Graph 2 of the boundary profiles is obtained by solving the problem (2.1) by the iterative regularizing Newton method (2.2) with the aid of the parallel Gauss or Gauss-Jordan algorithms (dotted lines).

For approximation of the integral operator (2.2) we used the two-dimensional analog of the rectangle quadrature formulas for 111×35 points of the grid domain with grid widths $h_x = 0.5$ and $h_y = 2$ (km). The positive parameters α_k were chosen from numerical experiments taking into account the right side of equation (2.1), namely $\alpha_k \ll F(x, y)$. In Figure 3 for problem (2.1) $H = 2$ (km) is the depth of the surface

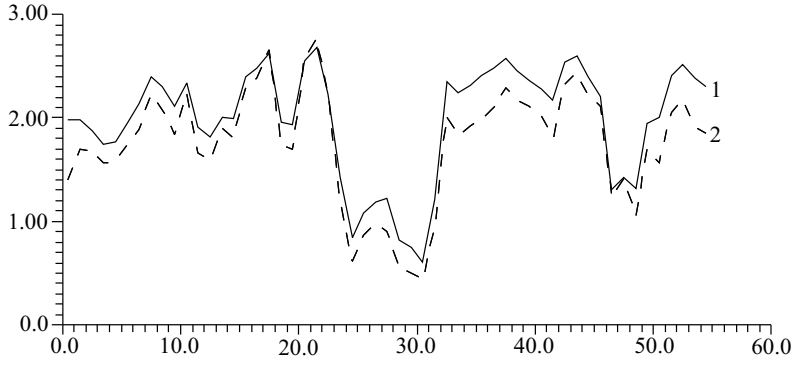

Figure 3. Boundary profiles for $H = 2$ (km)

S, $f = 6.67 \cdot 10^{-5}$ is the gravitation constant, $\Delta\sigma = 0.48$ (g/cm^3) is the density jump on the interface, $z_0(x, y) = 0.3$ (km) is the initial approximation and $\alpha_k = 2.5$ is a sequence of the positive parameters. In Figure 4 for problem (2.1) $H = 10$ (km) is the

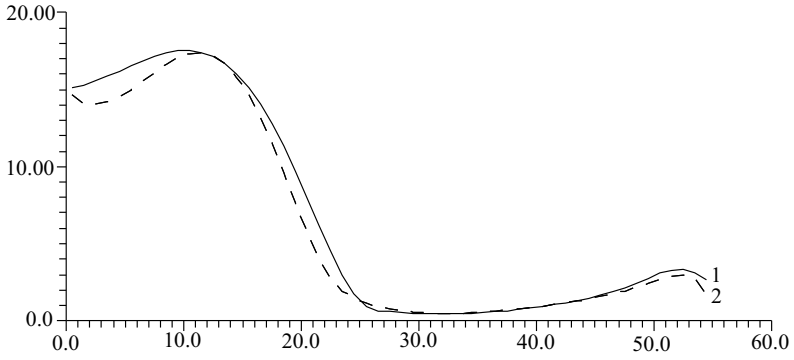

Figure 4. Boundary profiles for $H = 10$ (km)

depth of the surface S, $f = 6.67 \cdot 10^{-5}$ is the gravitation constant, $\Delta\sigma = 0.23$ (g/cm^3) is the density jump on the interface, $z_0(x, y) = 0.3$ (km) is the initial approximation and $\alpha_k = 0.8$ is a sequence of the positive parameters.

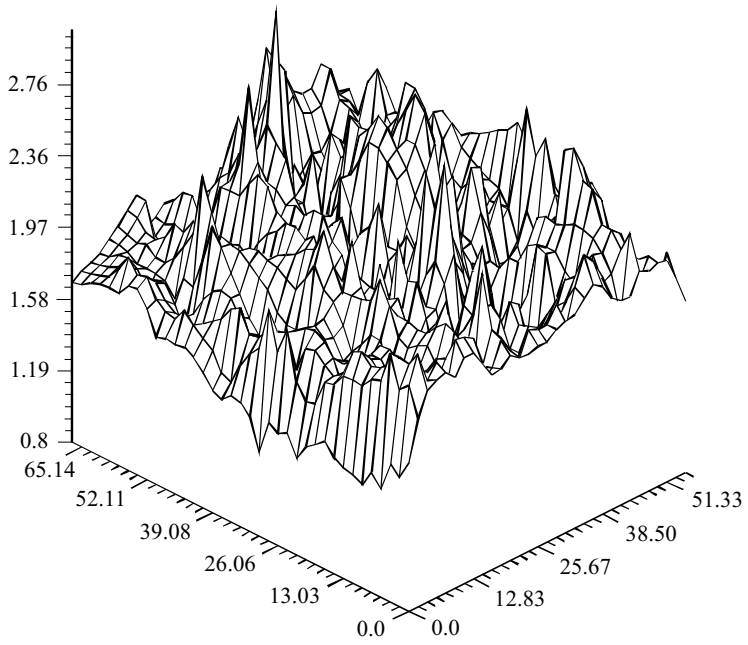

Figure 5. The reconstructed interface S_1

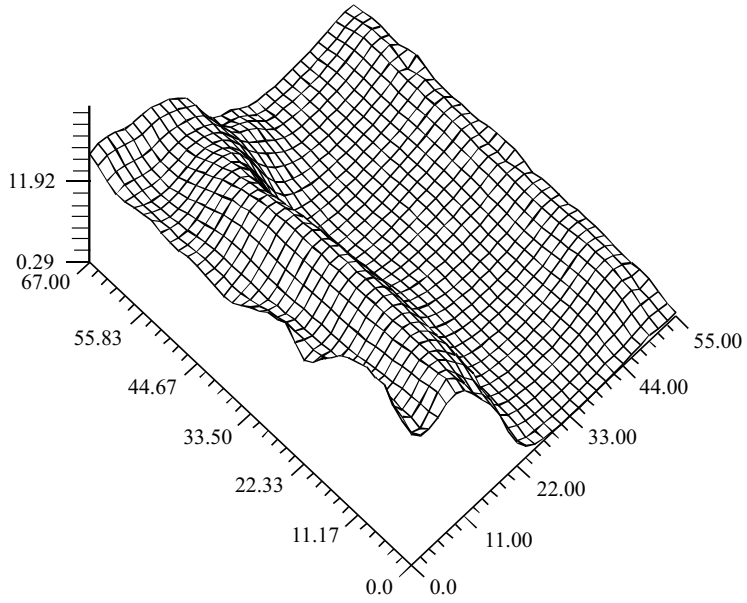

Figure 6. The reconstructed interface S_2

In Figures 5 and 6 the three-dimensional interfaces S_1 and S_2 for the real gravity field of some area in the Urals for $H = 2$ (km) and $H = 10$ (km) are represented. They are reconstructed by the iterative regularizing Newton method (2.2) with the help of the parallel Gauss or Gauss-Jordan algorithms.

The main conclusion is the following. The interfaces S_1 and S_2 obtained as solutions of the gravity inverse problem (2.1) by the iterative regularizing Newton method (2.2) correspond to the real conceptions about the investigated region in the Urals. Parallelization of the algorithms decreases the time of solving the problems.

Acknowledgement. The support provided by the Russian Foundation for Basic Research (project No. 03-01-00099) is gratefully acknowledged. The author is grateful to V.V. Vasin, the corresponding member of the Russian Academy of Sciences for the formulation of the problem and attention to the work.

REFERENCES

1. Vasin, V.V., Perestoronina, G.Ya., Prutkin, I.L. and Timerkhanova, L.Yu.: *Reconstruction of relief of geological boundaries in the three-layered medium using the gravitational and magnetic data.* In: *Proceedings of "Geophysics and Mathematics",* 35-41. Institute of Mines, UrB RAS, Perm, 2001.
2. Baranov, A.V., Latsis, A.O., Sazhin, C.V. and Khramtsov, M.Yu. The MVS–1000 System User's Guide. *http://parallel.ru/mvs/user.html.*