Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

A Parallel Multigrid Solver For Block-Tridiagonal Stencil Matrices Derived From Acoustic Wave Equation On Large Finite Difference Grids

Marcelo Carrion[1]
Lucia Catabriga[2]
Computer Science Department, Federal University of Espírito Santo
Alvaro L. G. A. Coutinho[3]
High Performance Computing Center, Federal University of Rio de Janeiro
William Batty[4]
Ehsan Naeini[5]
Ikon Science Ltd

Abstract. In this work we present efficient numerical solvers for large linear systems originated from a second-order finite difference discretization of three-dimensional acoustic wave problems in the frequency domain. We consider Geometric Multigrid strategies combined with the well known Conjugate Gradient and Red-Black SOR algorithms. Our method is designed for parallel execution on shared-memory platforms using the *OpenMP* paradigm. The experiments show that the parallel Conjugate Gradient method applying Multigrid as a preconditioner offers the best results in terms of CPU time and number of iterations.

Keywords. Geometric Multigrid, Conjugate Gradient, Red-Black SOR, Acoustic wave, Block-tridiagonal stencil matrix

1 Introduction

Multigrid solvers have a long history in exploration seismology. Gray and Epton [1] introduced multigrid migration to image both mild and steeply dipping events but with a computational cost similar to that of small migration aperture. Although, strictly speaking, this was not a standard multigrid algorithm but such approach suits the seismic data well. This is because based on the Nyquist criterion a smaller emergence angle allows a larger trace spacing, while a larger emergence angle requires smaller trace spacing. The authors used this property to optimize the migration in a multigrid fashion.

Moving on to inversion applications, Saleck et al. [2] combined multigrid and gradient methods to solve 2D acoustic wave equation. In such cases, solving a nonlinear optimization problem requires a discretization of the problem followed by numerical integration.

[1] mcarrion@inf.ufes.br
[2] luciac@inf.ufes.br
[3] alvaro@nacad.ufrj.br
[4] wbatty@ikonscience.com
[5] enaeini@ikonscience.com

DOI: 10.5540/03.2017.005.01.0098

Forward modelling is a central ingredient of Full Waveform Inversion (FWI), which is a popular and technically accurate method for obtaining a better velocity model. At the same time, a fundamental part of iterative methods for forward modelling is the preconditioner in which a common practice is to use multigrid solvers. The ideas introduced in this paper have the potential to be implemented as part of FWI, though understanding the time efficiency of our algorithms in such contexts requires more in-depth analysis and comparison with the state-of-the-art methods.

In this work, we analyze the computational impact of parallelizing the solving of large systems of linear equations stemming from second-order finite difference discretization of the three-dimensional acoustic wave equation in the frequency domain. We apply Multigrid strategies combined with the Conjugate Gradient and Red-Black SOR solvers, considering a shared-memory execution under *OpenMP*.

The paper is organized as follows. Section 2 describes the acoustic wave equation and the resulting linear system. In section 3, we introduce the Geometric Multigrid, outline some of its features, discuss the use of Multigrid as a preconditioner for the Conjugate Gradient algorithm and explain our implementation details. The results from numerical experiments are shown in section 4 and, finally, in section 5, we draw our main conclusions.

2 Problem description

The three-dimensional acoustic wave equation is defined in the frequency domain as

$$\nabla^2\psi(\mathbf{r}) + k^2\psi(\mathbf{r}) = f(\mathbf{r}) \qquad \text{in} \quad \Omega \subset \mathbb{R}^3 \quad , \tag{1}$$

which is called the Helmholtz equation and where $\psi(\mathbf{r})$ is the wavefield at $\mathbf{r} = (x, y, z)$, $f(\mathbf{r})$ is a source term, $k = \omega/c$ is the wave number, ω is the angular frequency and c is the media velocity. Proper boundary conditions should be specified [3]. Considering a second-order finite difference formulation, Ω is discretized into an uniform grid with $N = n^3$ unknown points, as shown in Fig. 1. The discretization yields a linear system $Au = f$ with a block-tridiagonal coefficient matrix A, which has heptadiagonal structure as $(-1, -1, -1, d_i, -1, -1, -1)$, where d_i is a diagonal value.

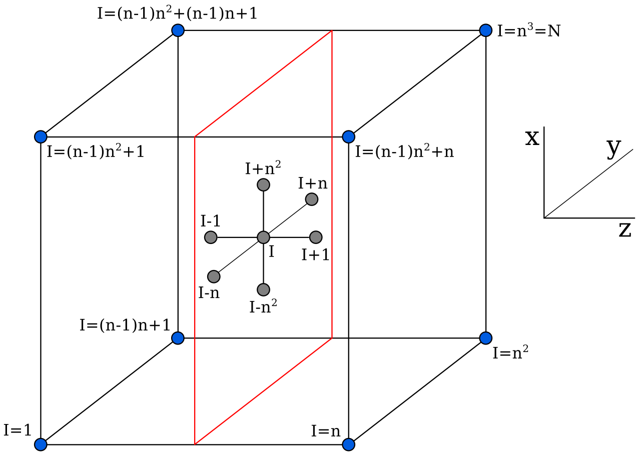

Figure 1: Cubic domain, where each x-y plane (in red) is called a *z-layer*.

3 Multigrid strategies

The Multigrid method originated from an attempt to correct the limitations of traditional relaxation schemes, such as Jacobi and Gauss-Seidel [4]. These methods possess the smoothing property, i.e., they efficiently reduce (oscilatory) high-frequency components of the solution error, but have little effect on the low-frequency ones. Multigrid improves on those methods by trying to eliminate low-frequency error through the coarse-grid correction strategy. Given an approximate solution v to equation $Au = f$, the idea is to move *(down)* to a coarser grid and relax on the error $e = u - v$ by using the residual equation $Ae = r = f - Av$, and then move back *(up)* to the fine grid transferring the solution.

The algorithm works by performing a sequence of so called V-cycles, until convergence is reached. They combine smoothing and coarse-grid correction to approximate a solution v^0 for the system $A^0u^0 = f^0$, as shown in the pseudocode below.

```
procedure V-Cycle ($v^0, f^0, A^0, A^1, ..., A^L$)
    for $l = 0$ to $L - 1$ do
        Relax $\nu_1$ times on $A^l u^l = f^l$ with initial guess $v^l$
        Compute $f^{l+1}$, which is the residual $f^l - A^l v^l$ restricted $[l \to l + 1]$ to coarser grid
        $v^{l+1} \leftarrow 0$
    end for
    Relax $\nu_1 + \nu_2$ times on $A^L u^L = f^L$ with initial guess $v^L$
    for $l = L - 1$ downto $0$ do
        Compute $err$, which is $v^{l+1}$ prolonged $[l + 1 \to l]$ to finer grid
        Correct $v^l \leftarrow v^l + err$
        Relax $\nu_2$ times on $A^l u^l = f^l$ with initial guess $v^l$
    end for
end procedure
```

In Geometric Multigrid, the matrices $A^0, A^1, ..., A^L$ are defined by discretizing the original problem at different levels of resolution: 0 is the original and finest level, with L being the coarsest. Regarding the coarse-grid levels, a coarse grid always has twice the grid spacing of the next finer grid. For intergrid transfers, we use linear interpolation – or prolongation – to move data from coarse to fine grids, and restriction (moving vectors from a fine grid to a coarse grid) is done by averaging values at neighboring fine grid points. Finally, some decisions must be made when using Multigrid: one must choose the relaxation method for the smoothing task, and also determine the number of coarse-grid levels L and the number of relaxations ν_1 and ν_2 to be performed. These three parameters are generally very small numbers.

The relaxation scheme chosen here is the Red-Black Successive Over-Relaxation (RB-SOR), a variation of the standard SOR where the grid points of the domain are divided into red and black points: a red point is always surrounded by black points and vice-versa. At each relaxation step, first the red points are updated (in any order, as they are dependent only on black points), then the black ones. Points of the same color are independent from each other and can be updated simultaneously [5], within multiple parallel threads.

We also can exploit the symmetry and positive definiteness of A, by using a preconditioned Conjugate Gradient method to solve the system $Au = f$. A preconditioner matrix M is needed, which should be a good approximation to A, easy to compute and must also be symmetric positive definite. There are several known techniques to obtain the preconditioning $p = M^{-1}v$, and in the present work we adopt a V-cycle of Multigrid as the preconditioner, following the ideas found in [6]. More specifically, we perform one V-cycle for the problem $Ap = v$, with zero initial guess and zero boundary conditions.

In the algorithms implemented for this work, no matrices are stored in memory. Every matrix operation, be it matrix-vector multiplication or RB-SOR relaxation, is optimized to exploit the known stencil structure of coefficient matrix A. To improve performance even further, we parallelized the main components of those algorithms. For Multigrid, the RB-SOR relaxations and both the restriction and prolongation operations – i.e., the computation of the restricted residual and the prolonged solution error – are performed using parallel threads. In the Conjugate Gradient method, the matrix-vector multiplication is parallelized and naturally so is the preconditioning using the Multigrid V-cycle. Those four operations rely on for loops, which are the target of our efforts.

In the following section, we analyze the behavior of the three different methods applied to solve the linear systems derived from the acoustic wave equation: the Conjugate Gradient (CG), a Multigrid solver using Red-Black SOR as smoother (MG-RBSOR) and a Multigrid preconditioned Conjugate Gradient (MGpCG), which also employs the RB-SOR in preconditioning.

4 Numerical results

We consider both the heterogeneous and homogeneous versions of the acoustic wave problem stated in section 2, subject to zero boundary conditions, to run our tests. For the heterogeneous case, the diagonal component d_i in coefficient matrix A has a different value for each *z-layer* in the domain Ω, while in the homogeneous case d_i is a constant. The right-hand side vector f is a point source δ-function. Our code was written in C and compiled using gcc version 4.8.2 with *-O3* optimization level. The numerical experiments were run under Ubuntu 14.04 LTS on a 64-bit machine with 64GB RAM and a four-cored Intel Xeon CPU E5-46030 @ 2.00GHz processor.

In all tests, iterations are halted when the residual drops below 10^{-8}. For the MG-RBSOR solver, the following parameters were set: $L = 3$; $\omega = 1.6$ (for the RB-SOR); $\nu_1 = \nu_2 = 2$. With respect to the V-cycle performed inside the MGpCG, these parameters apply: $L = 3$; $\omega = 1.6$; $\nu_1 = \nu_2 = 4$, and instead of relaxing $\nu_1 + \nu_2$ times at the coarsest level, CG iterations are applied to solve the equation system, stopping when the residual drops below 10^{-3}. In the shared-memory parallel execution using *OpenMP*, four parallel threads are employed. Our results follow next.

Figures 2 and 3 display the results from solving – in sequential and in parallel mode, respectively – the heterogeneous problem by MG-RBSOR, CG and MGpCG. In the graphs, the colored bars give the runtime and inside the circles are the number of iterations for each method. We can observe that, for all methods, the iterations count remains fixed or

DOI: 10.5540/03.2017.005.01.0098

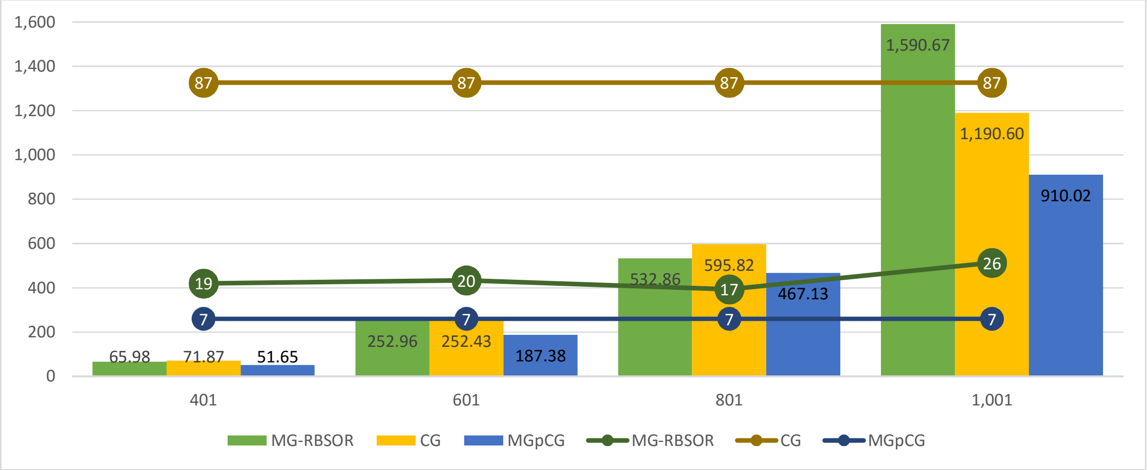

Figure 2: CPU time, in seconds, and iterations count for different sizes (n) of the heterogeneous problem run in sequential mode.

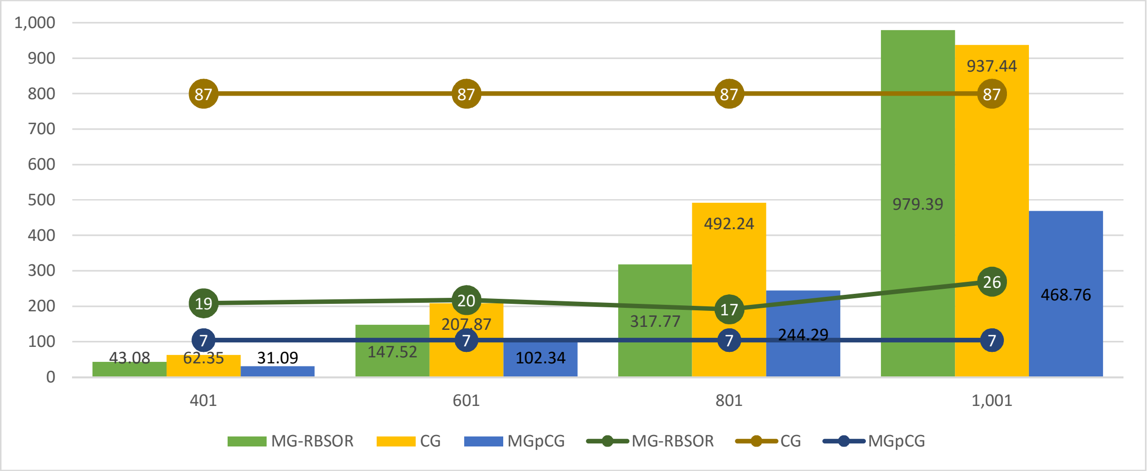

Figure 3: CPU time, in seconds, and iterations count for different sizes (n) of the heterogeneous problem run in parallel mode (4 threads).

DOI: 10.5540/03.2017.005.01.0098

around a constant value as n, the grid dimension, increases – with the MGpCG attaining the best solution times. Furthermore, by running the MGpCG program in parallel with *OpenMP*, we achieve a time reduction above 40% for all cases tested.

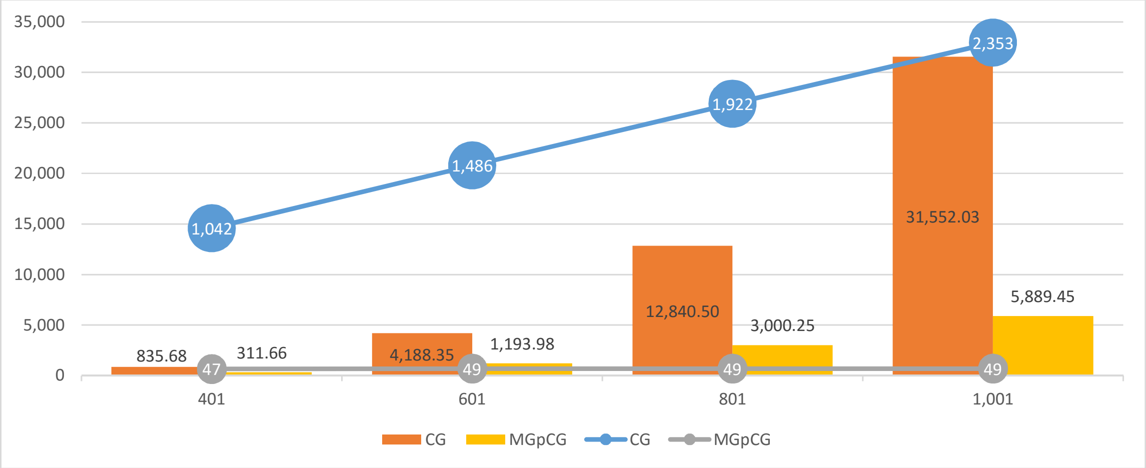

Figure 4: CPU time, in seconds, and iterations count for different sizes (n) of the homogeneous problem run in sequential mode.

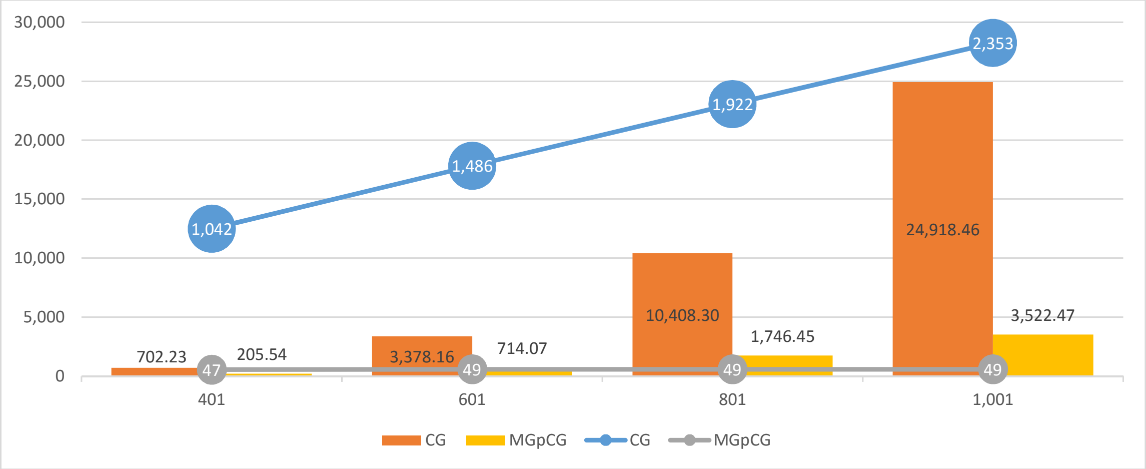

Figure 5: CPU time, in seconds, and iterations count for different sizes (n) of the homogeneous problem run in parallel mode (4 threads).

The homogeneous problem is solved using the CG and MGpCG. The results obtained are shown in Figs. 4 and 5, that provide the time for computing the solution in sequential and in parallel mode, respectively, along with the corresponding number of iterations. Analyzing the graphs, the iterations count for the CG algorithm rises steeply with the problem size, while it is constant in the MGpCG. The preconditioning clearly improves solution times (for $n = 1001$, the parallel MGpCG is 85% faster than parallel CG), and the *OpenMP* parallelized MGpCG program reduced the runtime by 39% on average.

DOI: 10.5540/03.2017.005.01.0098

5 Conclusions

In this work we presented efficient numerical Multigrid strategies to solve large systems of linear equations arising from a second-order finite difference discretization of three-dimensional acoustic wave problems in the frequency domain. We consider the Multigrid scheme combined with the Conjugate Gradient and Red-Black SOR solvers. Our experiments show that the parallel matrix-free Multigrid preconditioned Conjugate Gradient method accelerates convergence and can reduce the runtime by around 40% for the heterogeneous and homogeneous acoustic wave applications. With this optimized approach we are able to solve large linear systems ($\mathcal{O}(10^9)$ unknowns) in standard workstations in less than one hour (worst case). The present algorithms will be extended to more complicated cases, involving different sources and boundary conditions found in more practical seismic modeling problems.

Acknowledgments

This work is partially supported by ANP. Lucia Catabriga and Alvaro L. G. A. Coutinho also acknowledge the support of CNPq, FAPERJ and FAPES.

References

[1] S. H. Gray and M. A. Epton. Multigrid migration: Reducing the migration aperture but not the migrated dips. *Geophysics*, 55 (7):856–862, 1990.

[2] F. M. Saleck, C. Buncks, S. Zaleski, and G. Chavent. Combining the multigrid and gradient methods to solve the seismic inversion problem. In *Proceedings of the SEG international conference and exhibition*, 1993.

[3] W. Zhang and Y. Dai. Finite-Difference Solution of the Helmholtz Equation Based on Two Domain Decomposition Algorithms. *Journal of Applied Mathematics and Physics*, 1:18–24, 2013.

[4] W. Briggs, V. Henson, and S. McCormick. *A Multigrid Tutorial, Second Edition.* Society for Industrial and Applied Mathematics, second edition, 2000.

[5] D. J. Evans. Parallel S.O.R. iterative methods. *Parallel Computing*, 1(1):3–18, August 1984.

[6] A. McAdams, E. Sifakis, and J. Teran. A parallel multigrid Poisson solver for fluids simulation on large grids. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 65–74, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

DOI: 10.5540/03.2017.005.01.0098