A time-domain preconditioner for the Helmholtz equation: Analysis and performance on GPUs

Christiaan C. Stolk[1,*]

[1]Korteweg-de Vries Institute for Mathematics, University of Amsterdam, Amsterdam, The Netherlands

*Email: C.C.Stolk@uva.nl

Abstract

Several methods in the literature determine solutions to the Helmholtz equation by solving instances of a discrete time-domain wave equation. In this work we study a new method of this type. Given an indefinite linear system, a matrix recurrence relation is constructed, such that in the limit of infinitely many time steps the exact discrete solution is obtained, i.e. unaffected by time-discretization errors. Using a large, finite number of time steps, an approximate solution is obtained. To improve the convergence, the process is used as a preconditioner for GMRES, and the time-harmonic forcing term is multiplied by a smooth window function. We study the convergence of the method analytically and numerically, and conclude with some initial results about the performance on the GPUs.

Keywords: Helmholtz equation, fast solvers, wave equation, parallel computing

1 Introduction

High-frequency Helmholtz equations, i.e. when a large number of wave lengths fit inside the domain, are still difficult to solve, even though a large number of approaches has been studied.

The cost of these different methods may be of different forms. Domain decomposition methods typically require the factorization of sparse matrices from discretized Helmholtz equation on the subdomains. Such factorizations require large amounts of memory and are often (for larger problems) not so easy to parallellize. Other methods, including time-domain Helmholtz solvers, generally require many applications of sparse matrices like those from a discrete Helmholtz or time-domain wave equation. While requiring many computations, these methods can be much less memory intensive.

In all cases, the question is how to optimally make use of modern hardware such as Graphics Processing Units (GPUs). GPUs have many compute cores (1000s per GPU unit), a large memory bandwidth (between compute cores and GPU memory) and are suitable to execute structured and "local" computations with high efficiency. Relatively few works study how to exploit these capabilities. Our study of time-domain Helmholtz solvers is motivated by the possibilities of such modern hardware, see also [2,3].

2 Method

The method takes as a starting point a linear system

$$HU = F, \tag{1}$$

where H is a complex $N \times N$ matrix, such that $\operatorname{Re} H$ is symmetric and $\operatorname{Im} H$ is symmetric positive semidefinite. For simplicity we also assume $\operatorname{Im} H$ is diagonal. This is written in the form

$$(-\omega^2 I + i\omega B + A)U = F. \tag{2}$$

where $A, B \in \mathbb{R}^{N\times N}$ and $\omega \in \mathbb{R}$ are such that $H = -\omega^2 I + i\omega B + A$ (here ω is a frequency parameter, but need not be equal to the physical frequency).

Equation (2) is related to an ordinary differential equation (ODE) via the substitution $\frac{d}{dt} \leftarrow i\omega$. This scheme will be solved with a time-harmonic forcing term f_n. To obtain exact time-harmonic solutions at the choice of frequency ω, the ODE is discretized using a modified leap frog scheme

$$\begin{aligned} &\frac{1}{\Delta t^2}\left(u_{n+1} - 2u_n + u_{n-1}\right) \\ &+ \frac{1}{2\Delta t}\tilde{B}\left(u_{n+1} - u_{n-1}\right) + \tilde{A}u_n = \alpha^{-1} f_n, \end{aligned} \tag{3}$$

where $\tilde{A} = \alpha^{-1}A$, $\tilde{B} = \alpha^{-1}\beta B$, with

$$\begin{aligned} \alpha &= \frac{(\Delta t\,\omega)^2}{2 - 2\cos(\omega\Delta t)}, \\ \beta &= \frac{\omega\Delta t}{\sin(\omega\Delta t)}. \end{aligned} \tag{4}$$

The choice of A and B, ω and the time-step Δt can be made such that (3) is a stable scheme.

To approximate the solution U to (1) one can solve (3) for u_n, during a long time interval $[0, T]$, with n up to $T/\Delta t$ and with

$$f_n = \chi(n\Delta t)e^{i\omega n\Delta t}F \qquad (5)$$

with χ a suitable window function. Then

$$U \approx e^{-i\omega T}u_{T/\Delta t} =: P_T(F). \qquad (6)$$

The resulting map from F to the result of (6) is called a time-domain preconditioner and denoted $P_T(F)$.

3 Results and discussion

Theoretical and numerical results were obtained, to be summarized here. Some of the results have been published in [5].

The convergence was studied theoretically and indeed the approximate solution $P_T(F) \to H^{-1}F$ as $T \to \infty$.

The cost of this approach depends strongly on the size of the eigenvalues, e.g. on the "gap" defined by

$$\text{gap} = \min\{|\lambda_1|, \ldots, |\lambda_N|\}. \qquad (7)$$

A study of the speed of convergence reveals that the error from a preconditioned iterative method can be estimated by

$$\text{error} < Ce^{-(\#\text{timesteps})\cdot\text{gap}}. \qquad (8)$$

(The idea of studying the dependence on the gap was take from [1].)

Numerical results obtained so far are encouraging. For example, a 3-D problem with about 2.54e7 degrees of freedom (the SEG/EAGE salt model, discretized using the method of [4] using 6 points per wavelength) was solved in 601 seconds on Macbook Pro (2019) with 16 GB of memory. Numerical results for different discretizations and using GPUs are forthcoming.

Overall, the numerical examples show good results for finite-difference discretized Helmholtz equations. For finite-element discretizations, both this method and alternative methods will be more costly and further research is necessary to analyse the benefits in this case.

References

[1] D. Appelo, F. Garcia and O. Runborg, Waveholtz: Iterative solution of the Helmholtz equation via the wave equation, *SIAM Journal on Scientific Computing*, **42**(2020), pp. A1950–A1983.

[2] M.J. Grote, F.Nataf, J.H. Tang, and P.-H. Tournier, Parallel controllability methods for the helmholtz equation. *Computer Methods in Applied Mechanics and Engineering*, **362**(2020), 112846.

[3] H. Knibbe, C.W. Oosterlee and C. Vuik, GPU implementation of a Helmholtz Krylov solver preconditioned by a shifted Laplace multigrid method, *Journal of Computational and Applied Mathematics*, **236**(2011) pp. 281–293.

[4] C.C. Stolk, A dispersion minimizing scheme for the 3-d Helmholtz equation based on ray theory. *Journal of computational Physics*, **314** (2016) pp. 618–646.

[5] C. C. Stolk, A time-domain preconditioner for the Helmholtz equation, *SIAM Journal on Scientific Computing* **43**(2021): pp. A3469–A3502.