Lanczos strategies for the transient linear wave equation

E. Rodríguez[1] G. Montero[1]

Abstract

We consider the solution of sparse linear systems $Ax = b$ arising from finite difference discretization of the transient lineal wave equation. Two different strategies are studied: the preconditioned conjugate gradient method for the symmetric case and the preconditioned BiCGSTAB for the nonsymmetric one. Finally some numerical experiment are defined in order to illustrate the convergence behaviour of the linear solvers, as well as the effect of different preconditioners.

Introduction

Sparse linear systems of equations $Ax = b$ arising from the finite difference discretization of the wave equation are considered. We have studied the solution of such systems by using two of the most popular iterative methods based on Krylov subspaces, the Conjugate Gradient (CG) [1] for the cases involving symmetric matrices and BiCGSTAB [2] for nonsymmetric ones. Each of them are implemented in its preconditioned versions which make use of suitable preconditioning matrices.

As we will see, the linear systems to be solved here are generally nonsymmetric since the discretization of the boundary conditions introduces new rows in the matrix which makes it loose the symmetry. However the symmetry is kept in the equations corresponding to the inner nodes. Thus, we first consider the solution of each linear system by using the CG method starting from a null vector as initial guess. This allows to work with the symmetric part of matrix A and the GC algorithm is able to be used. The other strategy is to apply the BiCGSTAB algorithm which is twice more expensive approximately, but in the other hand it may take advantage of the solution obtained in the previous time step as initial guess. Anyway, there are cases where the hard nonsymmetry leads us to apply only nonsymmetric solvers. In both situations we have compared the performance of the convergence of the iterative methods when several preconditioners are used. The main question is to know whether a robust but expensive preconditioner is competitive with another cheaper but less efficient.

In section 2 we present the equations which define our wave problem. The finite difference scheme and the corresponding linear system is developed in section 3. The Lanczos-type algorithms and the preconditioners are described in section 4. In order to compare the different strategies for solving the set of linear systems arising in the whole run-time of any nonsteady problem, some numerical experiments are studied in section 5. Finally, section 6 contains the conclusions obtained in this paper.

State equations

Let consider the second order 2-D wave equation

$$u_{tt} - \alpha\ (u_{xx} + u_{yy}) = f\ in\ \Omega \tag{1}$$

where $\Omega \subset \mathbb{R}^2$ is here a rectangular domain, and α, f, are, in general, functions of

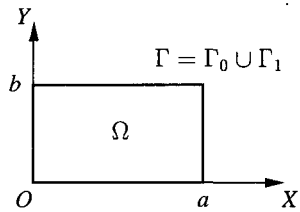


Figure 1: Domain of the problem

x, y, t. If $\alpha = T/\rho$, e.g., the equation (1) may represent the vibration of a wave on a membrane of mass density ρ per unity of surface, subject to a constant stress T. For the well posed of this problem we need to defined suitable boundary and initial conditions. We establish boundary conditions of Dirichlet and Neumann type on Γ_0 and Γ_1, respectively,

$$u(x, y, t) = c(x, y, t)\ on\ \Gamma_0 \tag{2}$$

$$\frac{\partial u}{\partial n}(x, y, t) = d(x, y, t)\ on\ \Gamma_1 \tag{3}$$

and initial conditions given by

$$u(x, y, 0) = u_0(x, y) \tag{4}$$

$$u'(x, y, 0) = v_0(x, y) \tag{5}$$

Finite difference scheme and linear system

In the following, we will refer to a regular mesh of rectangles defined to apply a finite difference scheme to the second order PDE corresponding to the wave problem. The ordering of nodes in the mesh is the lexicographical, as we can see in the example of Figure 2. The values of Δx and Δy define the level of spatial fineness of the mesh in directions x and y, respectively, and Δt is the time step of the process. Our model adjust the input parameter for discretization to the size of the domain. Thus, i.e., we will take,

$$\Delta x = \frac{l_x}{\lceil l_x/\Delta x \rceil}$$

where $\lceil \cdot \rceil$ means the C rounding function to bigger integer.

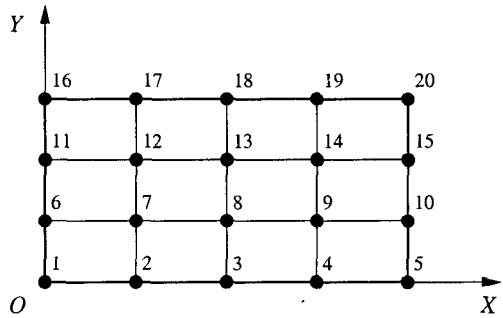


Figure 2: Lexicographical ordering

Here, the finite difference scheme takes into account the classical discretization scheme for second time-derivatives,

$$u_{tt}(x_i, y_j, t_k) = \frac{u_{i,j}^{k+1} - 2u_{i,j}^{k} + u_{i,j}^{k-1}}{\Delta t^2} + O\left(\Delta t^2\right) \tag{6}$$

and for spatial derivatives,

$$u_{xx}(x_i, y_j, t_k) \cong \left[\frac{u_{i+1,j}^{k+1} - 2u_{i,j}^{k+1} + u_{i-1,j}^{k+1}}{2\Delta x^2} + \frac{u_{i+1,j}^{k-1} - 2u_{i,j}^{k-1} + u_{i-1,j}^{k-1}}{2\Delta x^2}\right] + O\left(\Delta x^2\right) \tag{7}$$

$$u_{yy}(x_i, y_j, t_k) \cong \left[\frac{u_{i,j+1}^{k+1} - 2u_{i,j}^{k+1} + u_{i,j-1}^{k+1}}{2\Delta y^2} + \frac{u_{i,j+1}^{k-1} - 2u_{i,j}^{k-1} + u_{i,j-1}^{k-1}}{2\Delta y^2}\right] + O\left(\Delta y^2\right) \tag{8}$$

The substitution of schemes 6, 7 and 8 into equation (1) yields,

$$\begin{aligned}(1 + 2P + 2Q)\, u_{i,j}^{k+1} - P u_{i+1,j}^{k+1} - P u_{i-1,j}^{k+1} - Q u_{i,j+1}^{k+1} - Q u_{i,j-1}^{k+1} = \\ 2u_{i,j}^{k} - (1 + 2P + 2Q) u_{i,j}^{k-1} + P u_{i+1,j}^{k-1} + P u_{i-1,j}^{k-1} + Q u_{i,j+1}^{k-1} + Q u_{i,j-1}^{k-1} + f_{i,j}^{k}\end{aligned} \tag{9}$$

with $P = \Delta t^2 \alpha / 2\Delta x^2$ and $Q = \Delta t^2 \alpha / 2\Delta y^2$. The subscripts i, j correspond to $x_i = x_0 + i\Delta x$, $y_j = y_0 + j\Delta y$, respectively, and the superscript k indicates the updated time $t_k = t_0 + k\Delta t$. The process starts from u^2 and needs to know u^0 and u^1 in any point of Ω. Both of these initial vectors are obtained from the initial condition by using a standard Euler scheme,

$$u_{i,j}^{0} = u_0(i,j)_{i,j} \tag{10}$$

$$u_{i,j}^{1} = u_{i,j}^{0} + \Delta t \cdot v_0(i,j) \tag{11}$$

The implicit scheme (9) takes advantage of its unconditional stability, in front of the explicit ones which are less expensive but their stability is conditioned by some relationship between the discretization parameters. This fact implies that the time-step size affects to the level of discretization of the grid and thus, leads to larger linear

systems of equations. Nevertheless, even if the convergence is assured, the accuracy of the solution will depend on the time and spatial discretization.

In order to simplify the implementation of the matrix construction, we have introduced the boundary nodes corresponding to Dirichlet condition into the linear system instead of removing the corresponding equations and changing the right hand side vector.

$$u_{i,j}^{k+1} = c(x_i, y_j, t_{k+1}) \tag{12}$$

Then, the matrix becomes nonsymmetric, even in its pattern. Furthermore, as if we consider some part of the boundary with Neumann conditions the nonsymmetry of the matrix may not be avoided to guarantee the accuracy order, the use of nonsymmetric solvers seems to be a unquestionable statement. The finite difference schemes for Neumann conditions yield,

$$\left.\frac{\partial u}{\partial x}\right|_{\Gamma_2} = \frac{u_{i-2,j}^{k+1} - 4u_{i-1,j}^{k+1} + 3u_{i,j}^{k+1}}{2h} \tag{13}$$

$$\left.\frac{\partial u}{\partial x}\right|_{\Gamma_4} = \frac{3u_{i,j}^{k+1} - 4u_{i+1,j}^{k+1} + u_{i+2,j}^{k+1}}{2h} \tag{14}$$

$$\left.\frac{\partial u}{\partial y}\right|_{\Gamma_3} = \frac{u_{i,j-2}^{k+1} - 4u_{i,j-1}^{k+1} + 3u_{i,j}^{k+1}}{2k} \tag{15}$$

$$\left.\frac{\partial u}{\partial y}\right|_{\Gamma_1} = \frac{3u_{i,j}^{k+1} - 4u_{i,j+1}^{k+1} + u_{i,j+2}^{k+1}}{2k} \tag{16}$$

Equation (9) leads to a $n \times n$ sparse linear system which must be solved in every time-step, n being the number of nodes in the grid. Since the run-time is generally large, the selection of a suitable iterative solver and a solution strategy for each linear system will define an important part of the efficiency of the numerical model. Let a linear system involving the solution at step k be,

$$A^k u^k = b^k \tag{17}$$

Evidently, if neither of α, Δt, Δx depends on time, matrix $A^k = A$ and it does not vary along the run-time process,

$$A = \begin{pmatrix} N_1 & N_2 & N_3 & & & & & & & \\ & \ddots & & & & & & & & \\ & & D & & & & & & & \\ & & & \ddots & & & & & & \\ -Q & 0 & \cdots & -P & C & -P & 0 & \cdots & -Q & \\ 0 & -Q & 0 & \cdots & -P & C & -P & 0 & \cdots & -Q \\ & & & & & & \ddots & & & \\ & & & & & & & N_1 & N_2 & N_3 \\ & & & & & & & & \ddots & \\ & & & & & & & & & D \end{pmatrix}$$

where $C = 1 + 2P + 2Q$, D represents the diagonal terms due to Dirichlet conditions, N_1, N_2, N_3 correspond to the coefficients in Neumann conditions, respectively, and b^k includes the right hand side of equation (9) and equations 12-16.

Lanczos-based solvers

Herein, we have considered the Conjugate Gradient algorithm for symmetric linear systems and BiCGSTAB for nonsymmetric ones. While the choice of CG for symmetric problems is clearly accepted for the scientific community, the selection of a suitable method for nonsymmetric problems is much more difficult. However, BiCGSTAB has proved to be an efficient solver for linear systems like the ones arose in this paper.

Preconditioned Conjugate Gradient algorithm (CG)

The Conjugated Gradient method [1] has been widely used for solving large and sparse linear systems of equations involving a symmetric and positive definite matrix. Although our linear systems are nonsymmetric, if and only if we consider only Dirichlet conditions, we can work with the symmetric part of A by starting from the zero initial guess. A preconditioned version of CG algorithm is given below.

Algorithm 1. Preconditioned CG.
Let x_0 be an initial guess
$r_0 = b - Ax_0$
$\rho_0 = 1$
$p_0 = 0$
while $\|r_{i-1}\| \,/\, \|r_0\| \geq \varepsilon$ $(i = 1, 2, 3, \ldots)$ **do**
  Solve $Mz = r_{i-1}$
  $\rho_i = r_{i-1}^t z$
  $\beta_i = \rho_i / \rho_{i-1}$
  $p_i = z + \beta_i \rho_{i-1}$
  $v_i = Ap_i$
  $\alpha_i = \beta_i / p_i^t v_i$
  $x_i = x_{i-1} + \alpha_i p_i$
  $r_i = r_{i-1} - \alpha_i v_i$
end while

Preconditioned BiCGSTAB algorithm

The BiCGSTAB algorithm [2] is a stabilized modification of the Biconjugate Gradient method [4] for solving nonsymmetric linear systems. The computational cost of this method is approximately twice of CG since two matrix-vector products are involved. However they will be compared in the near symmetric cases in order to test if the possibility of an arbitrary initial guess in BiCGSTAB is enough to win CG. A preconditioned BiCGSTAB algorithm is as follows [3],

Algorithm 2. Preconditioned BiCGSTAB

Let x_0 be an initial guess
$r_0 = b - Ax_0$
Set $\hat{r}_0$ arbitrary, such that $\hat{r}_0^t r_0 \neq 0$, e.g. $\hat{r}_0 = r_0$
$\rho_0 = \alpha_0 = \omega_0 = 1$
$p_0 = v_0 = 0$
while $\|r_{i-1}\| / \|r_0\| \geq \varepsilon$ $(i = 1, 2, 3, \ldots)$ **do**
  Solve $Mz = r_{i-1}$
  $\rho_i = \hat{r}_0^t z$
  $\beta_i = (\rho_i / \rho_{i-1})\,(\alpha_{i-1} / \omega_{i-1})$
  $p_i = z + \beta_i\,(p_{i-1} - \omega_{i-1} v_{i-1})$
  $y = Ap_i$
  Solve $Mv_i = y$
  $\alpha_i = \frac{\rho_i}{\hat{r}_0^t v_i}$
  $s = r_{i-1} - \alpha_i y$
  $u = z - \alpha_i v_i$
  $t = Au$
  $\omega_i = \frac{t^t s}{t^t t}$
  $x_i = x_{i-1} + \alpha_i p_i + \omega_i u$
  $r_i = s - \omega_i t$
end while

Preconditioners

A linear system of equation $Ax = b$ is ill-conditioned if small variations in the entries of A or b produce high variations in the solution. The condition number of matrix A is used as a measurement of it. In order to improve the conditionement another matrix M is constructed such that $M \simeq A^{-1}$ somehow. Thus, the preconditioned system becomes $MAx = Mb$. In this paper we have implemented the Jacobi and Optimal Diagonal preconditioners for symmetric systems and in addition SSOR(ω) and ILU(0) for nonsymmetric ones. The Jacobi preconditioner is taken to be the diagonal of A, $M = Diag(A)$. The Optimal Diagonal preconditioner corresponds to the best approximate inverse M with diagonal pattern,

$$m_{ii} = \frac{a_{ii}}{\sum_{k=1}^{n} a_{ki}^2}$$

The SSOR(ω) preconditioner arises from comparing the preconditioned Richardson's method with the SSOR(ω) iterative method,

$$M = \left(I + \omega E D^{-1}\right) \left(\frac{D + \omega F}{\omega(2 - \omega)}\right)$$

where $A = D + E + F$ is the standard sum decomposition of matrix A. Finally, the ILU(0) preconditioner is the result of an incomplete LU factorization where each factor keeps the same pattern as A,

$$m_{ij} = 0 \quad if \quad a_{ij} = 0$$

$$\{A - LU\}_{ij} = 0 \quad if \quad a_{ij} \neq 0$$

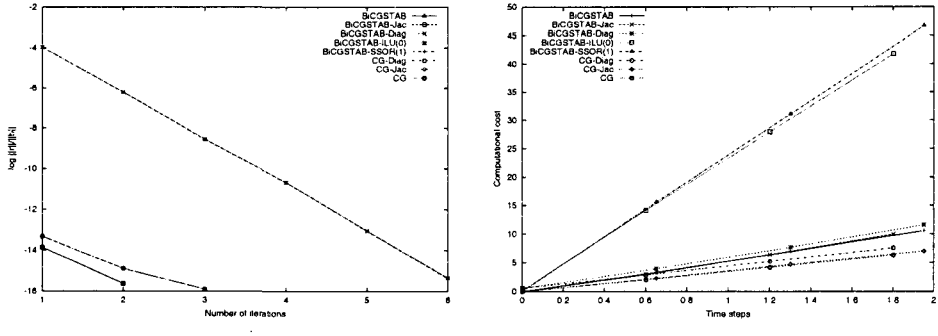
Notice that both SSOR(ω) and ILU(0) have the same sparsity pattern as A and thus, the storage of these preconditioners only requires to keep a double precision matrix array containing the entries of M, since the location of them are given by the same integer matrix array of A.

Numerical experiments

The first numerical example is defined by the equation (1) and a source function,

$$f = -k\, w\, \sin(\pi x)\, \sin(\pi y)\, [w - 2\, \sin(wt)]\, e^{-t}$$

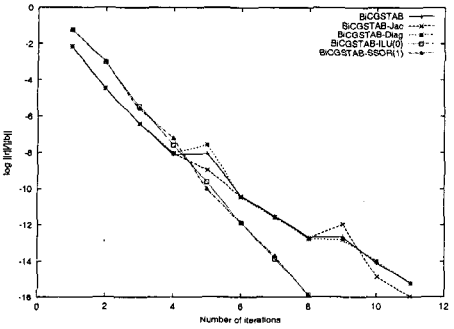
with $\alpha = \frac{T}{\rho}$, $w = \sqrt{1 + \frac{2T}{\rho}}$ and $\rho = k = 1$. Dirichlet conditions $u(x, y, t) = 0$ on Γ and initial conditions $u(x, y, 0) = 0$, $u_t(x, y, 0) = 0$ in Ω are considered. The corresponding matrix A is near symmetric and thus, we have applied CG and BiCGSTAB for solving every linear system along the time-running. Figures 3(a) and (b) show the performance of both algorithms with different preconditioners for $T = 10^2$. The results indicate that CG is preferable to BiCGSTAB if it is possible to be used. Furthermore, in this problem, a simple preconditioner produces a faster resolution even with more iterations than others like ILU(0) or SSOR(1) which are more expensive. In the second experiment we have only changed the boundary conditions on Γ_1 andΓ_3 to be $\frac{\partial u}{\partial n}(x, y, t) = 0$ and $T = 10^4$. Now matrix A is nonsymmetric due to the discretization scheme of the first derivatives. This makes the CG impracticable and convergence is only obtained for BiCGSTAB. Here again the cheaper preconditioners allows to solve the set of linear systems along the time steps corresponding to two seconds of simulation of the model. Now ILU(0) and SSOR(1) produces less iterations (see Figure 4(a)) but at a higher cost which is increased with the simulation time (see Figure 4(b)). The last experiment



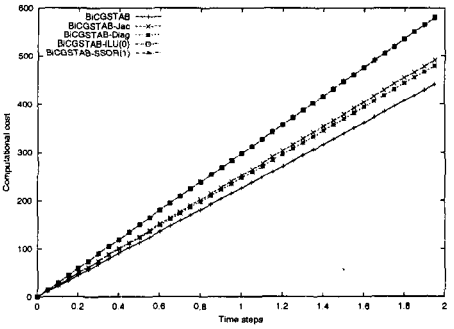
(a) Performance of iterative solvers

(b) Computational cost

Figure 3: First example



(a) Performance of preconditioned BiCGSTAB

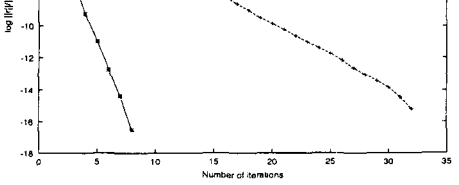


(b) Computational cost

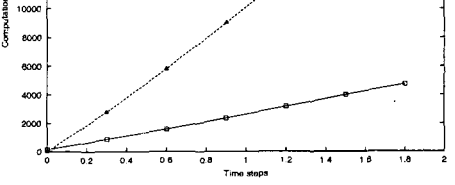
Figure 4: Second example

is also similar to the first. In this case we consider $\alpha = Ke^{(\Delta t+2-x-y)^\gamma}$ and $K = 1$. We have solved the linear systems arising at each time-step with BiCGSTAB. The values of γ have been selected diabolically in order to make harder the convergence of the iterative solver. Figures 5(a) and (b) represent the behaviour of BiCGSTAB with ILU(0) or SSOR(1) in the case of $\gamma = 2$ for the linear system in $t = 1$ and the computational cost versus time steps, respectively. No convergence was reached with other selection.

Finally, Figures 6(a) and (b) show that ILU(0) is a competitive choice in front of other cheaper preconditioner if the same matrix A is involved in the set of linear systems along the process. Notice that the additional cost due to the construction of ILU(0) is neutralized along the time steps by the higher number of BiCGSTAB iterations with the other preconditioners.

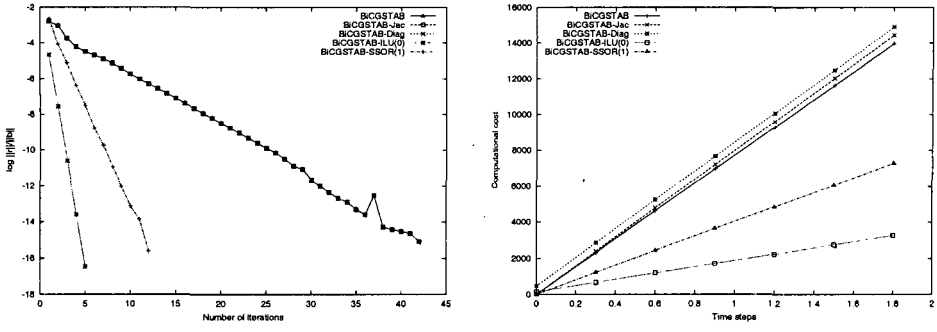


(a) Performance of preconditioned BiCGSTAB



(b) Computational cost

Figure 5: Third example with $\gamma = 2$



(a) Performance of preconditioned BiCGSTAB

(b) Computational cost

Figure 6: Third example with $\gamma = -0.1$

Conclusions

We have implemented an efficient model for the wave equation using an unconditionally stable finite difference scheme and fast solvers for the associated linear systems of equations. For problems where the convergence is hard, the selection of a suitable preconditioner becomes as important as the selection of the iterative solver. In addition, we have experimentally shown that the more complicated preconditioners may be competitive with the simpler ones in such problems. Further research must be carried out on the behaviour of these algorithms when used for solving the set of linear systems arising from the linearization of nonlinear problems.

References

[1] M.R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, *Jour. Res. Nat. Bur. Standards,* 49, 6, (1952), 409-436.

[2] H.A. van der Vorst, Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.,* 13, (1992), 631-644.

[3] G. Montero, R. Montenegro, J.M. Escobar, A 3-D diagnostic model for wind field adjustment, *Journal of Wind Engineering and Industrial Aerodynamics,* 74-76, (1998), 249-261.

[4] R. Fletcher, Conjugate gradient methods for indefinite systems, *Lectures Notes in Math.*, 506, (1976), 73-89.

1 Department of Mathematics. University of Las Palmas de Gran Canaria. Edif. de Informática y Matemáticas, Campus Univ. de Tafira, 35017-Las Palmas de Gran Canaria