The data-parallel iterative solution of the finite element discretization of stream-function models for global ocean circulation

M.B. van Gijzen, G.L.G. Sleijpen
Department of Mathematics, Utrecht University
P.O. Box 80.010, 3508 TA Utrecht, The Netherlands
E-mail: mbvg@xs4all.nl, sleijpen@math.ruu.nl

A.J. van der Steen
Department of Physics and Astronomy, Utrecht University
P.O. Box 80.000, 3508 TA Utrecht, The Netherlands
E-mail: A.vanderSteen@fys.ruu.nl

Keywords: Finite Elements, Stream Function Problems, Iterative Solution Methods, Parallel Computing, BiCGstab(ℓ), Polynomial Preconditioning

ABSTRACT

This paper describes the iterative solution of linear systems that arises after discretization of stream-function problems on multiply-connected domains by a certain finite element method. We apply our ideas to global ocean circulation and describe experiments that are performed on the Cray C90.

INTRODUCTION

The behaviour of 2-D incompressible flow can be described by a set of two 2nd order partial differential equations plus boundary conditions for the scalar quantities vorticity and stream-function. The no-slip boundary condition, i.e. no flow at solid boundaries, states that at solid boundaries the normal derivative of the stream-function is zero, and that the stream-function itself is constant. In multiply-connected domains there are several internal boundaries. The constant values of the stream function differ per boundary and are not explicitly known. To be able to determine them, extra integral conditions have to be imposed.

A usual way to discretize a set of partial differential equations is to define a grid of nodal points and to derive explicit relations between a nodal quantity and the nodal quantities in neighbouring nodes. The explicit relations can e.g. be derived by the Finite Element Method. This approach yields a sparse and often very large system of equations. If the grid is structured, the resulting matrix will also be structured (if the nodes are properly ordered); only a few diagonals will contain nonzero entries. This feature is important since indirect addressing can then often be avoided when operations with the matrix are performed. Indirect addressing may be inefficient and may hamper parallelization and vectorization.

The stream-function has a constant value at each internal boundary. To determine this value, formula's have to be derived to relate it to the values of the stream-function and vorticity in neighbouring nodes. An internal boundary may have many neighbouring nodes, and because of this the structure in the matrix will be lost, even if the grid is structured.

The subject that inspired us for this research is simulation of global ocean circulation. The internal boundaries are the islands and continents. The simulation of global ocean circulation is extremely computationally intensive, it is one of the so called "grand challenges", and all possibilities to reduce the computational effort should be exploited. To avoid indirect addressing and to facilitate parallelization and vectorization, we must exploit the structure in the grid, despite the presence of islands and continents. This can to a large extent be achieved by a Finite Element based discretization, described in [8].

In this paper we describe how the resulting system can be solved by the iterative solution technique BiCGstab(ℓ), combined with a Polynomial Preconditioner.

A SIMPLE OCEAN CIRCULATION MODEL

Steady barotropic flow in a homogeneous ocean with constant depth and in nearly equilibrium can be described by the following set of partial differential equations:

$$-r\,\nabla^2\psi - \beta\,\frac{\partial\psi}{\partial x} - A\,\nabla^2\zeta = \nabla\times\mathbf{F} \quad \text{in } \Omega,$$

$$\nabla^2\psi + \zeta = 0\,. \tag{1}$$

The variables in these equations are the stream-function ψ and the vorticity ζ. $\mathbf{F}$ is the external force field and is equal to the wind stress τ divided by the average depth of the ocean H times the water density ρ:

$$\mathbf{F} = \frac{\tau}{\rho H}\,. \tag{2}$$

The other parameters are the bottom friction coefficient r, the Coriolis parameter β and the viscosity coefficient A. The stream-function ψ is related to the velocity component u and v by

$$u = \frac{\partial\psi}{\partial y}, \quad v = -\frac{\partial\psi}{\partial x}, \qquad \zeta = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\,. \tag{3}$$

The no-slip boundary conditions (zero velocity at coasts) state that the normal derivative of the stream function is zero at the coasts

$$\frac{\partial\psi}{\partial n} = 0 \quad \text{on } \Gamma_k, \quad k = 1,\cdots,n_k. \tag{4}$$

and that the stream function itself is constant on each continent (or island, island and continents are the same in this context)

$$\psi = C_k \quad \text{on } \Gamma_k, \quad k = 1,\cdots,n_k, \tag{5}$$

n_k is the number of continents. The values of the constants C_k are not specified. In order to be able to determine them one has to impose integral conditions, stating that the water level is continuous around each island or continent.

$$\oint_{\Gamma_k} A\,\frac{\partial\zeta}{\partial n}\,ds = -\oint_{\Gamma_k} \mathbf{F}\cdot\mathbf{s}\,ds. \tag{6}$$

The equations are commonly expressed in spherical coordinates to map the physical domain onto a rectangular domain. The coordinate transformation causes singularities on the poles. The singularity at the South Pole gives no problem since the South Pole is land. The singularity at the North Pole is solved by imposing a slip boundary condition, i.e. $\psi = \zeta = 0$ on the North Pole.

OBTAINING THE SYSTEM OF EQUATIONS

In [8] a new technique is proposed for discretizing and solving stream-function problems on multiply-connected domains. The procedure is as follows.
Step 1: Define a structured grid on the rectangular domain, disregarding islands and continents.
Step 2: Discretize the set of partial differential equation everywhere, using the Finite Element Method, and again disregarding islands and continents. This yields a structured matrix. Discretization with the Finite Element Method is essential in this setting in order to adhere to the integral conditions.
Step 3: Define Master and Slave nodes on the continents. Contributions to Slave nodes, e.g. of the wind forcing, must be lumped to values in the Master node.
Step 4: Solve the following system of equations:

$$\mathbf{P}^T\mathbf{K}\mathbf{P}\mathbf{x} = \mathbf{P}^T\mathbf{f}. \tag{7}$$

Here, $\mathbf{x}$ contains the values of the stream-function in the Master nodes on the islands and in the sea-nodes, and all nodal values of the vorticity. The vector $\mathbf{f}$ represents the (curl of) the force field, both on land and on sea. The matrix $\mathbf{K}$ describes the physical behavior of incompressible viscous flow, in the presence of the Coriolis force and bottom friction. The matrix $\mathbf{P}$ relates the Master nodes to the Slave nodes and is defined as follows. The columns that correspond to sea-points or to vorticity unknowns are just basis-vectors:

$$\mathbf{P_j} = \mathbf{e_j} \qquad \forall \mathbf{j} \text{ in the sea.} \tag{8}$$

And a column that correspond to the Master value of the stream-function on a continent is the sum of the basis vectors corresponding to all nodal values of the stream-function on the continent:

$$\mathbf{P_k} = \sum \mathbf{e_j} \qquad \forall \mathbf{j} \text{ on same continent.} \tag{9}$$

The matrix $\mathbf{P}^T\mathbf{K}\mathbf{P}$ is unstructured, but $\mathbf{K}$ is structured. We will try to exploit this feature in the solution procedure.

ITERATIVE SOLUTION OF THE LINEAR SYSTEM

The matrix $\mathbf{K}$ is structured, but what have we really gained? If we would apply a direct solution method we would not have to decompose the structured matrix $\mathbf{K}$ but the unstructured matrix $\mathbf{P}^T\mathbf{K}\mathbf{P}$. Moreover, the latter matrix would have a large bandwidth, since continents are literally global phenomena. We therefore solve the system by an iterative solution technique. Krylov-subspace type methods like CG [3] (for symmetric positive definite systems) and GMRES [4] are among the most popular and powerful methods. There are, however many more good methods, in particular for solving non-symmetric systems.

The ideal method should combine the attractive properties of both CG-type methods (fast and memory efficient) and GMRES-type methods (robust). A method that does this to a large extent is BiCGstab(ℓ) [5]. The method combines in each iterative step a low degree GMRES-polynomial with the (Bi-)CG-polynomial that has been build up in all previous iterations. We have used an enhanced implementation [1], that includes special techniques to improve the convergence properties [6] and the accuracy of the solution [7].

Iterative solution techniques should be combined with a preconditioner to speed up convergence. The most commonly used techniques construct a preconditioning matrix by making an incomplete decomposition of the system matrix $\mathbf{A}$, in our situation $\mathbf{A} = \mathbf{P}^T\mathbf{K}\mathbf{P}$. This would mean that the preconditioning matrix would be unstructured. Moreover, applying a preconditioner based on an incomplete decomposition would involve back and forward substitutions, operations that are hard to parallelize or vectorize. For these reasons we propose to apply the Polynomial Preconditioner that is proposed in [9]. The idea of this method is to construct a (low degree) polynomial in the system matrix that mimics the action of its inverse. The coefficients of the preconditioning polynomial are computed by minimizing a residual polynomial on a contour that encloses the spectrum of the system matrix. This spectrum is not known in advance, but can be estimated by making use of the close relation between BiCGstab(ℓ) and Bi-CG (or more precise with the Bi-Lanczos method). The multiplication of a vector by a Polynomial Preconditioner $P_q(\mathbf{A})$ of degree q can be performed by applying Horner's rule. This requires q multiplications with the system matrix and q vector-updates. We combine the preconditioner with diagonal scaling, i.e. the system is scaled by its main diagonal.

The complete algorithm can be parallelized by parallelizing the basic operations in the algorithm. BiCGstab(ℓ) is composed of four basic operations: matrix-vector product, preconditioning operation, vector updates and inner product operations. The system matrix itself is only addressed in the matrix-vector multiplication. The preconditioning operation is in our case composed of matrix-vector products and vector-updates. The vector-update and inner product are trivial to parallelize on data-parallel machines: optimized (BLAS) routines are available on all important computers.

Hence the operation we will have to focus on is the multiplication with the system matrix. The operation $\mathbf{y} = \mathbf{P}^T\mathbf{KPx}$ can be performed in three steps:

- Give all Slave nodes the value of the "master" node: $\bar{\mathbf{x}} = \mathbf{Px}$ (Scatter)
- Multiply: $\bar{\mathbf{y}} = \mathbf{K}\bar{\mathbf{x}}$ (Structured matrix-vector product).
- Add up the values of the stream-function on each continent: $\mathbf{y} = \mathbf{P}^T\bar{\mathbf{y}}$ (Gather+Update)

The second step, the structured matrix-vector product is completely data-parallel if performed diagonal-wise. The Scatter operation is parallelizable, but efficient processing will be hampered by indirect addressing. The Gather operation is not parallelizable and may become dominant in terms of computing time.

Note that multiplying with the unstructured matrix $\mathbf{P}^T\mathbf{KP}$ would require a lot more indirect addressing: a Gather operation for each equation of the system.

NUMERICAL RESULTS

For the numerical experiments we have taken the following realistic values for the physical parameters:

- Wind stress τ: Long term averaged data for January [2],
- Average depth $H = 500\,\mathrm{m}$,
- Water density $\rho = 1000\,\mathrm{kg/m}^2$,
- Earth radius $R = 6.4 \cdot 10^6\,\mathrm{m}$,
- Coriolis parameter $\beta = 2.3 \cdot 10^{-11} \cos\theta\,(\mathrm{ms})^{-1}$,
- Lateral viscosity parameter $A = 500\,\mathrm{m}^2/\mathrm{s}$,
- Bottom friction coefficient $r = 5 \cdot 10^{-6}\,\mathrm{s}^{-1}$.

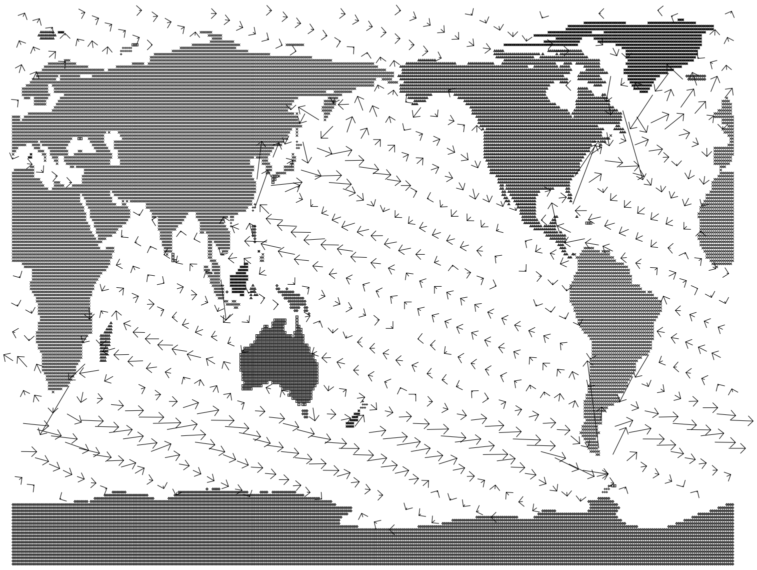

Figure 1: Velocity field.

We have solved the problem on a grid with a 1^o resolution, which yields a system of $\approx$ 130,000 linear equations. Figure 1 shows the computed velocity field: the results are satisfactory, all major surface currents are present.

To study the effect of the Polynomial Preconditioner, we have applied preconditioners of degree 0, 1, 3, and 5 in combination with BiCGstab(4). Table 1 shows the operations that were required to solve the system to the desired accuracy (relative residual norm smaller that 10^{-6}). The Polynomial Preconditioner significantly reduces the numbers of operations. The

Degree Pol.Prec.	# Iterations	# Matvecs	# Inprods	# Updates
0	2254	18032	49588	81144
1	548	8768	12056	20276
3	248	7936	5456	9672
5	179	8592	3938	7339

Table 1: Iterations and operations for BiCGstab(4) plus Pol. Prec.

wall-clock time is reduced from 1018s if only diagonal scaling is applied, to 316 for a Polynomial Preconditioner of degree 3. Hence we may conclude that the Polynomial Preconditioner is quite effective here.

The computations are performed at a rate of approximately 0.1 Gflop. This is about one tenth of the (single processor) peak performance. This slightly disappointing figure is totally due to the Gather operation in the matrix-vector multiplication. This operation is performed at a rate of only 1 Mflops. As a result, about 60% of the time is spend in this routine! The structured matrix-vector product is performed at a satisfactory rate of 400 Mflops. The diagonalwise matrix-vector multiplication requires three load operations and one store for only two floating point operations. This implies that one can not expect a beter performance than half of the peak of 1 Gflops. About 25% of the time is spend in this routine. Hence we may conclude that, although we have limited the number of indirect addressing, the single Gather operation in the matrix-vector product remains the bottle neck.

CONCLUDING REMARKS

In this paper we discussed the question how the type of linear systems that arise in global ocean circulation simulation can be solved efficiently on data-parallel computers, like e.g. Cray vector computers. To make efficient use of these computers it is important to avoid indirect addressing operations, like Gather and Scatter operations, as much as possible. Indirect addressing operations are unavoidable in global ocean circulation simulation due to the presence of islands and continents and the boundary conditions that have to be imposed there.

We used the iterative solution technique BiCGstab(ℓ) combined with a Polynomial Preconditioner. The Polynomial Preconditioner reduced the number of operations significantly.

The most time consuming part in the algorithm is the matrix-vector multiplication. We were able to limit the indirect addressing operations in the matrix-vector multiplication to only one Gather and one Scatter operation. But even then, the Gather operation proved to be the most time-consuming. The overall performance on the Cray C90 was reasonable, 0.1 Glops which is 10% of the single processor peak performance.

Acknowledgments. Diederik Fokkema is greatfully acknowledged for supplying the code for the BiCGstab(ℓ) algorithm. Henk van der Vorst, Kees Vreugdenhil and Hakan Oksuzoglu are acknowledged for their various contributions.

This research is supported by the Netherlands Organization for Scientific Research NWO, project 95MPR04.

REFERENCES

[1] D.R. Fokkema, *Enhanced implementation of BiCGstab(ℓ) for solving linear systems of equations*, Preprint nr. 976, Department of Mathematics, Utrecht University, September 1996.

[2] S. Hellerman and M. Rosenstein, *Normal Monthly Wind Stress Over the World with Error Estimates*, J. of Physical Oceanography, 13, pp. 1093–1104 (1983)

[3] M.R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Natl. Bur. Stand., 49, pp. 409–436 (1954)

[4] Y. Saad and M.H. Schultz, *GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7, pp. 856–869 (1986)

[5] G.L.G. Sleijpen and D.R. Fokkema, *BiCGstab(ℓ) for linear equations involving matrices with complex spectrum*, ETNA 1, pp. 11–32 (1994)

[6] G.L.G. Sleijpen and H.A. van der Vorst, *Maintaining convergence properties of BiCGstab methods in finite precision arithmetic*, Numerical Algorithms, 10, pp. 203–223 (1995)

[7] G.L.G. Sleijpen and H.A. van der Vorst, *Reliable updated residuals in hybrid Bi-CG methods*, Computing 56, pp. 141–163 (1996)

[8] M.B. van Gijzen, C.B. Vreugdenhil, and H. Oksuzoglu, *A finite element discretization for stream-function problems on multiply connected domains*, Preprint nr. 1005, Department of Mathematics, Utrecht University, March 1997

[9] M.B. van Gijzen, *A polynomial preconditioner for the GMRES algorithm*, J. Comp. and Appl. Math. 59, pp. 91–107 (1995)