



Three-dimensional simulations of spiral waves in reacting and diffusing media on DSM computers

E. M. Ortigosa¹, L. F. Romero¹ and J. I. Ramos²

¹ *Dept. de Arquitectura de Computadores,
Universidad de Málaga, Spain*

² *E.T.S. Ingenieros Industriales,
Universidad de Málaga, Spain*

Abstract

Parallel ADI and CG methods based on the time-linearization of the three-dimensional, reaction-diffusion equations which govern the propagation of spiral waves in excitable media, have been implemented on both shared- and distributed-memory computers. ADI has been implemented by means of a dynamic block cartesian decomposition and its efficiency tends to that of the CG technique as the mesh is refined. The largest efficiency of CG was obtained on an Origin-2000 with a message passing model, private memory and 16 processors, and the efficiency of CG is larger than that of ADI.

1 Introduction

Reaction-diffusion equations are ubiquitous in biology, combustion, ecology, etc., because of their relevance in pattern formation, ignition and extinction phenomena, etc. [1-3]. Many studies of reaction-diffusion equations have been concerned with equations for activators and inhibitors in one or two spatial dimensions, e.g., the Belousov-Zhabotinskii, Brusselator and Oregonator models, with and without extinction [3]. Of special interest to the study presented in this paper are the analytical and numerical analyses of the propagation of spiral waves in two-dimensional domains where it has been observed that these waves have a periodic pattern in the absence of heterogeneities, may exhibit breathing motions in the presence of obstacles or may simply be extinguished if a control parameter is activated in a sufficiently large region of the computational domain. In many of these cases, e.g.,



12 Applications of High-Performance Computers in Engineering VI

spiral waves in nonhomogeneous media, spiral waves are characterized by steep gradients in space and relaxation-type oscillations whose accurate simulation demands small spatial and temporal steps.

In three dimensions, there have been very few analytical and numerical studies of spiral waves, presumably because of both the large difficulties in examining wave propagation in three-dimensional space and the cost of such simulations [4]. As a consequence, filament models based on Fréchet formulae and differential geometry have been developed; these models are analogous to those of vortex filaments in theoretical fluid mechanics.

In this paper, three-dimensional simulations of the propagation of spiral waves in cubes without obstacles, and in the presence and absence of extinction sources are presented. These numerical simulations have been carried out by means of both time-linearized and nonlinearized techniques with and without operator splitting, i.e., with and without approximate factorization of the three-dimensional operator into one-dimensional ones. The numerical methods employed in the discretization of the governing partial differential equations have been implemented in a parallel fashion in both shared- and distributed-memory computers, and their performance is reported in this paper.

2 Governing equations and their discretization

Consider the following Belousov-Zhabotinskii system of reaction-diffusion equations

$$\frac{\partial \mathbf{U}}{\partial t} = \mathbf{D} \left(\frac{\partial^2 \mathbf{U}}{\partial x^2} + \frac{\partial^2 \mathbf{U}}{\partial y^2} + \frac{\partial^2 \mathbf{U}}{\partial z^2} \right) + \mathbf{S}(\mathbf{U}). \quad (1)$$

where $\mathbf{D}$ is a diagonal diffusivity tensor, $\mathbf{U} = (u, v)^T$, T denotes transpose, t is time, x , y and z denote spatial coordinates, $\mathbf{S} = \left(\frac{u(u-1) - (f v + \phi) \frac{u-q}{u+q}}{\epsilon}, u - v \right)^T$, f , q and ϵ are constants, and ϕ is a control parameter which may be a function of space and/or time. Equation (1) has been solved in a cube of side equal to 15 nondimensional units. Unless otherwise stated, $f = 1.4$, $q = 0.002$, $\phi = 0$ and $\epsilon = 0.01$, and the diagonal terms of the diffusivity tensor are $d_1 = 1$ and $d_2 = 0.6$.

By discretizing the time variable in eqn (1) by means of a Crank-Nicolson method, one can obtain a nonlinear elliptic equation which can be solved at each time step. This equation can be further linearized with respect to time in order to obtain a linear elliptic equation at each time step. In both the nonlinearized and time-linearized discretizations of the time variable, the second-order spatial derivatives were discretized by means of three-point, second-order accurate finite difference methods that provide a large system of nonlinear and linear, respectively, algebraic equations at each time step. The Newton-Raphson method was employed to solve the resulting nonlinear system of algebraic equations, and it should be noted that the linearized equations correspond to only one iteration of the Newton-Raphson technique. In addition to these techniques, an approximate factorization of the three-dimensional operator into a sequence of one-dimensional



ones (here referred to as ADI) was also used; this technique reduces the solution of the linear elliptic equation in three dimensions to the solution of one-dimensional, linear, two-point boundary value problems in each spatial dimension, but introduces second-order (in time) approximate factorization errors which can be eliminated in an iterative fashion; the approximate factorization errors are of the same order of magnitude as those introduced by the time discretization but may be large where the norm of the Jacobian matrix of the source terms is large, i.e., at the edges of the spiral wave. In the next paragraphs, some details about the methods employed to solve the discrete equations corresponding to eqn (1) are presented.

2.1 ADI method

The discrete operators corresponding to eqn (1) can be written as

$$\begin{aligned} & \left(\mathbf{I} - \frac{k}{2\Delta x^2} \mathbf{D} \delta_x^2 - \frac{1}{2} k \mu_1 \mathbf{J}^n \right) \Delta \mathbf{U}^{**} = \\ & \mathbf{D} \left(\frac{k}{\Delta x^2} \delta_x^2 + \frac{k}{\Delta y^2} \delta_y^2 + \frac{k}{\Delta z^2} \delta_z^2 \right) \mathbf{U}^n + k \mathbf{S}^n, \\ & \left(\mathbf{I} - \frac{k}{2\Delta y^2} \mathbf{D} \delta_y^2 - \frac{1}{2} k \mu_2 \mathbf{J} \right) \Delta \mathbf{U}^* = \Delta \mathbf{U}^{**}, \\ & \left(\mathbf{I} - \frac{k}{2\Delta z^2} \mathbf{D} \delta_z^2 - \frac{1}{2} k \mu_3 \mathbf{J} \right) \Delta \mathbf{U} = \Delta \mathbf{U}^*. \end{aligned} \quad (2)$$

where the approximate factorization errors have been neglected, k is the time step, Δx is the spatial step size in the x -direction, $0 \leq \mu_i \leq 1$, $i = 1, 2, 3$, $\mu_1 + \mu_2 + \mu_3 = 1$, the superscript n denotes the n -th time level, $\mathbf{J}$ denotes the Jacobian matrix of the mapping $\mathbf{U} \rightarrow \mathbf{S}(\mathbf{U})$, $\delta_x^2 v_i = v_{i+1} - 2v_i + v_{i-1}$, and $\Delta \mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$.

2.2 CG methods

The following linear system of equations was solved with the (iterative) conjugate gradient method

$$\begin{aligned} & \left(\mathbf{I} - \mathbf{D} \left(\frac{k}{2\Delta x^2} \delta_x^2 + \frac{k}{2\Delta y^2} \delta_y^2 + \frac{k}{2\Delta z^2} \delta_z^2 \right) - \frac{1}{2} k \mathbf{J}^n \right) \Delta \mathbf{U} = \\ & \mathbf{D} \left(\frac{k}{\Delta x^2} \delta_x^2 + \frac{k}{\Delta y^2} \delta_y^2 + \frac{k}{\Delta z^2} \delta_z^2 \right) \mathbf{U}^n + k \mathbf{S}^n. \end{aligned} \quad (3)$$

which corresponds to the time linearization and discretization of eqn (1) and, therefore, does not contain factorization errors; thus a comparison between the numerical results obtained with eqns (2) and (3) will indicate the magnitude of the approximate factorization errors.

The CG method contains two inner products, three $\alpha \mathbf{x} + \mathbf{y}$ operations and one matrix vector product whose computations require at least three synchronization



14 Applications of High-Performance Computers in Engineering VI

points, but it is not possible to merge messages. However, an optimized rearrangement of CG for parallel computations can be found in [5]; unfortunately, this optimized method is only useful when an incomplete factorization is employed as a preconditioner. As it will be shown below, this kind of preconditioner does not result in any acceleration in the convergence of the problem considered here.

Another modified CG method has been suggested by Aykanat et al. [6], as well as by Chronopoulos and Gear [7] whose main difference with the original one is that, if each iteration is performed in parallel on a distributed-memory system with the modified algorithm, the local values of the scalar products can be included in only one message in order to compute the global ones. Experiments with this modified CG method have shown that there is only a small increase in the sequential computational time but its parallel performance is excellent.

A main issue when solving linear systems of algebraic equations is the ordering of the equations. Blocking of the equations results in a small number of very large blocks, while blocking of the variables per node results in many small blocks. Here, we have tested both types of blocking and found that blocking of the variables results in faster simulations due mainly to a 3 and 5 % reduction in primary and secondary, respectively, cache missings.

We have tested the Jacobi, block Jacobi, incomplete Cholesky and incomplete-block Cholesky preconditioners in our studies. In the first and third preconditioners, the two dependent variables per node have been used to form 2×2 blocks; calculations have also been performed without any preconditioner (N.P.), but the number of iterations of the CG method becomes extremely large. In Table 1, the average number of iterations required by each method to converge is presented as a function of the grid size; the values shown in this table correspond to $k = 0.0004$ and 20 time steps. The above table clearly shows that the incomplete-block

Table 1: Average number of iterations of CG for convergence.

Grid	N.P.	Jacobi	B. Jacobi	I.Cholesky	I.B.Cholesky
51^3	5.95	2.25	2.25	2.3	2.20
101^3	6.00	3.1	3.20	3.2	3.1
201^3	n.a.	5.05	5.15	5.2	n.a.

Cholesky factorization is the most efficient preconditioner; however, the cost associated with this incomplete factorization is larger than that associated with the decrease in the number of iterations, even when using the Eisenstat "trick" [5]. Furthermore, calculations performed using a frozen preconditioner for several time steps were found to increase substantially the number of iterations of the CG method owing to the large rotational speeds of the spiral waves considered in this paper. For these reasons, a Jacobi preconditioner has been used in all the simulations presented below.



3 Parallel implementation

The ADI technique and the linearized CG technique discussed above have been implemented on shared-memory and shared- and distributed-memory models, respectively, as discussed below.

3.1 Parallelization of ADI

In this work, the dynamic block cartesian decomposition (DBCD) has been used for parallel implementation. In this technique, each processor has a contiguous grid block, but the decomposition changes as the one-dimensional operator is changed, i.e., it changes according to the spatial direction. As stated in [8], the optimal solution is reached when the data is partitioned in the z -direction while solving the algebraic equations in the x - and y -directions, whereas a partition in the x - or y -directions may be employed when solving the algebraic equations in the z -direction. Therefore, only prior to the solution of the z -direction, a communication step is required in order to have the data available to the different processors, and this communication results in cache missings in shared-memory computers and represents a major drawback for the parallel implementation of the ADI method.

The results shown in Figure 1 indicate that the parallel efficiency of DBCD is very near to 1, even with a high number of processors, but other decomposition strategies might yield better results. For example, the Bruno-Cappello algorithm [8] partitions the domain in blocks in such a manner that there is no more than one block per processor in each plane coordinate; however, this algorithm imposes some limitations to the parallel system because of its requirements for a number of processors equal to the square of a natural number. Moreover, the parallel efficiency shown in Figure 1 leaves little room for improvement and, as will be illustrated below, the main drawback of the ADI method is its sequential execution time compared with that of the CG method.

3.2 Parallelization of CG

The parallelization of the CG method is very simple since every vector operation of this algorithm can be parallelized separately, i.e., each processor executes scalar operations on a subset of the components of a vector, and a block partition of the vectors is sufficient to obtain a good performance. Some computations such as the scalar product require a communication step in order to obtain the final result. This step as well as other synchronization points imposes severe limitations to the parallel efficiency of the CG algorithm. In this paper, a reorganization, i.e., overlapping, of the computations in the CG algorithm has been implemented, in order to hide the latency of the communications. This overlapping of the communications with computations has been implemented by using asynchronous messages in the message passing model described below, and prefetch directives in a shared-memory environment. It must be noted that the parallelization of the sparse matrix-



16 Applications of High-Performance Computers in Engineering VI

times-vector product for block diagonal matrices can be performed with a nearly perfect overlap between the messages and the computations. CG has been par-

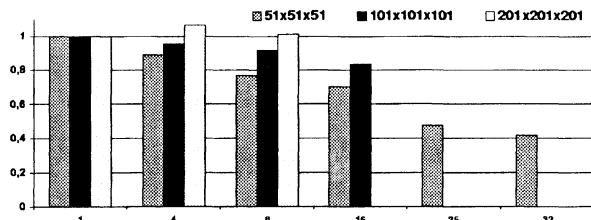


Figure 1: Efficiency of the ADI method as a function of the number of processors and grid size.

allelized by using a shared-memory model on an Origin-2000 computer, and the communications amongst processors have been made by using a shared arena and the synchronization has been implemented by means of lock variables. Figure 2 shows the efficiency of ADI and CG as a function of both the number of processors and the grid size using a shared-memory model, and indicates that the differences between the efficiency of CG and that of ADI decrease as the grid is refined, although the efficiency of ADI is lower than that of CG. It must be emphasized, however, that ADI was parallelized and implemented in a directional manner, i.e., one-dimensional operators were solved at each time step, and, perhaps, the efficiency of this method might increase substantially if it were implemented in a planar fashion, i.e., the three-dimensional operator is approximated by a sequence of two- and one-dimensional operators.

Calculations were also tried with a $201 \times 201 \times 201$ mesh on an Origin-2000, but we found that this computer has a bug on the virtual memory management. For this reason, a message-passing model with private memory was implemented on

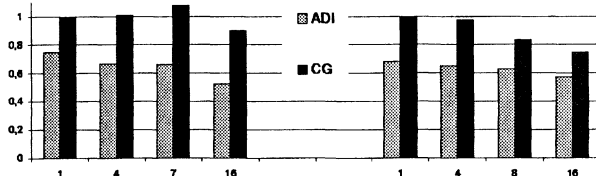


Figure 2: Efficiency of the ADI and CG methods as a function of the number of processors. Left: $51 \times 51 \times 51$ grid; right: $101 \times 101 \times 101$ grid.



Table 2: CPU time (in seconds) for one time step as a function of the number of processors on the CRAY T3E and Origin-2000 for with (O) and without (NO) overlapping between communications and computations.

NPES	CRAY T3E		Origin: O		Origin: NO	
	Grid 51	Grid: 101	Grid 51	Grid 101	Grid 51	Grid 101
1	1.003	n.a.	0.876	9.87	0.869	9.981
2	0.516	n.a.	0.460	4.43	0.513	4.517
4	0.264	2.656	0.207	2.39	0.254	2.443
8	0.143	1.396	0.0990	1.424	0.150	1.488
16	0.0820	0.745	0.0466	0.791	0.0670	0.799

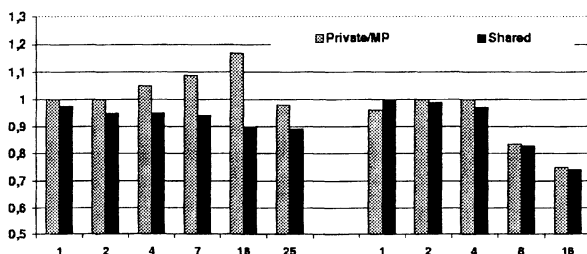


Figure 3: Efficiency of the ADI and CG methods as a function of the number of processors with shared-memory and message-passing models with private memory (MP). Left: $51 \times 51 \times 51$ grid; right: $101 \times 101 \times 101$ grid.

the same computer; this model did not overcome the failure on the virtual memory management but resulted in a large increase in efficiency as illustrated in Figure 3, i.e., an efficiency of 117% was obtained with 16 processors. This figure also shows that the efficiency of the two parallel implementations of CG on a shared-memory computer (Origin-2000) degrades as the mesh size is decreased owing to the number of cache hits for coarse grids.

The message-passing model has also been implemented on a CRAY T3E computer in order to both avoid the virtual-memory management problem found in the Origin-2000 and execute the same code with much finer grids. The CPU time for one time step obtained with the CG method on the CRAY T3E and Origin-2000 computers are presented in Table 2. In this table, the results obtained with and without overlapping between communications and computations are also presented. The overlapping was mainly based on the fact that the number of iterations of CG was nearly constant; for this reason, the assessment of convergence was delayed in order to hide the latency of communications, and resulted in a more efficient computation than those without overlapping on the Origin-2000. More-



18 Applications of High-Performance Computers in Engineering VI

over, we observed that the CPU time on the CRAY was almost independent of the overlapping.

4 Results

Calculations have been performed 1) without extinction, i.e., $\phi = 0$, 2) with an extinction barrier, i.e., $\phi = 0.2$ for $7.2 \leq y \leq 7.8$ and $\phi = 0$ otherwise, and 3) with a localized extinction zone which is a cube with center located at $(0, 0, 0)$ and length equal to 1.65 where $\phi = 0.2$. In all cases, homogeneous Neumann boundary conditions were employed at the faces of the computational domain, $k = 0.0004$, and the calculations were performed until $t = 60$ starting from initial conditions corresponding to a wedge for u . In all the calculations considered in this paper, it has been observed that the solution became periodic at about $t = 25, 30$, and 28 for the first, second and third cases, respectively. In the first case, i.e., without extinction, it has been observed that the spiral wave rotates around an axis parallel to the z -direction which passes through the center of the computational domain and remains anchored there, the u solution is symmetric about the $z = 0$ plane, and the rotational speed of the spiral wave decreases as it approaches the planes $z = -7.5$ and $z = 7.5$. Some sample results illustrating the u solution at $t = 41.6$ are presented in Figure 4 which indicates that, at this time, the spiral wave is nearly absent near the bottom and top planes, whereas the influence of the initial conditions can still be observed near these planes. The u -solution also shows that at $t = 41.6$, near to the top and bottom planar boundaries, the spiral wave is thicker and much shorter than that near $z = 0$.

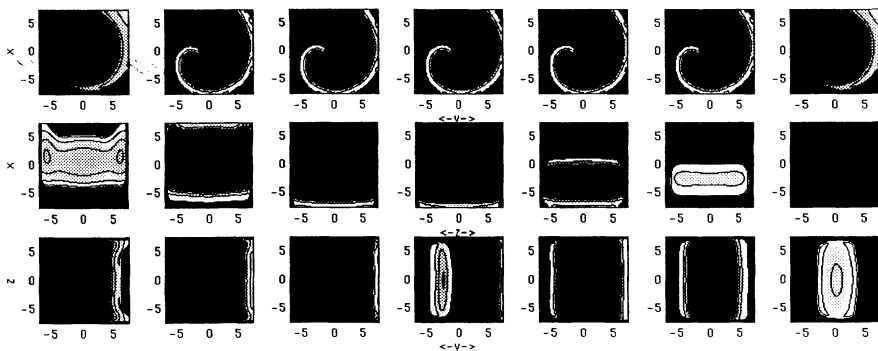


Figure 4: u -solution for a $101 \times 101 \times 101$ grid at $t = 41.6$. From left to right: a) top: $k=100, 84, 68, 51, 34, 18, 2$, b) middle: $j=100, 84, 68, 51, 34, 18, 2$, c) bottom: $i=100, 84, 68, 51, 34, 18, 2$.

In the second case, the extinction barrier is a slab parallel to the $x - z$ plane where the value of ϕ is different from zero; therefore, the source term for u de-

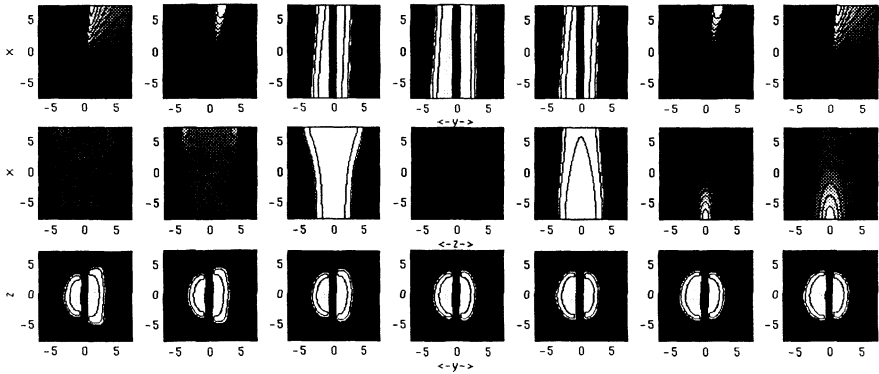


Figure 5: u -solution for a $101 \times 101 \times 101$ grid at $t = 41.6$. From left to right: a) top: $k=100, 84, 68, 51, 34, 18, 2$, b) middle: $j=100, 84, 68, 51, 34, 18, 2$, c) bottom: $i=100, 84, 68, 51, 34, 18, 2$.

creases exponentially whenever $u > q$ by an amount that is proportional to ϕ , i.e., the spiral wave may be extinguished as indicated in Figure 5. The u -solution in $x = \text{constant}$ planes indicates that almost planar fronts propagate initially from the left to the right boundaries but, on encountering the extinction barrier, they decelerate and emerge from this barrier until they reach the right boundary. Afterward, the u -solution is almost nearly homogeneous through $x = \text{constant}$ planes for a long period of time and planar fronts appear on the right and left sides of the extinction barrier until the former reaches the right boundary. Thereafter, the motion is repeated in a periodic fashion. The u -solution in the $y = 0$ plane is nearly nil, that for $y > 0$ shows curved fronts propagating from the top to the bottom boundaries that emerge, after some period of quieteness, from $y < 0$ propagating from the bottom to the top boundaries; this behaviour seems to correspond to a spiral wave that rotating in a clockwise manner encounters an extinction source, turns about and changes its direction of rotation and its size. The u -solution in $z = \text{constant}$ planes shows that the tip of the spiral wave disappears due to the extinction barrier and that the solution in $x > 0$ rotates clockwise until the advancing edge of the spiral wave encounters the extinction barrier. Afterward, the localized spiral wave is transformed in two fronts propagating upwardly in $x < 0$ and downwardly in $x > 0$ until, after some time, a thick spiral wave front is formed in $x < 0$. This front also rotates in a clockwise manner until its advancing edge encounters the extinction barrier; thereafter, an elongated, large-radius spiral wave appears in $x > 0$, and the whole process is repeated in a periodic fashion. The results with an extinction source also indicate that the effect of the initial conditions are felt only near the top and bottom boundaries, and that the presence of an extinction barrier causes the spiral wave to tumble about and change its rotational speed and direction.



20 Applications of High-Performance Computers in Engineering VI

In the third case, the results are similar to those of the first one except for the fact that the tip of the spiral wave rotates around the extinction source, and the wave is shorter and has a thicker tip. In the first case, the tip is anchored on the vertical axis passing through the center of the cube, while, in the third one, the anchoring point describes a trajectory which corresponds to the projection of the extinction cube into the $x - y$ -plane.

5 Conclusions

Linearized approximate factorization and conjugate methods for the numerical solution of three-dimensional reaction-diffusion equations corresponding to spiral wave propagation in excitable media have been parallelized and implemented in both shared- and distributed-memory computers. The parallelization of the approximate factorization technique has been carried out with a block dynamic cartesian decomposition and its efficiency is very near to one; however, this parallelization leaves little room for improvement. The parallelization of conjugate gradient methods has been performed in terms of both the overlapping of communications and computations. The overlapping of communications and computations has been carried out by using asynchronous messages in the message-passing version and prefetch directives in shared-memory computers.

References

- [1] Murray, J.D. *Mathematical Biology*, Springer-Verlag: New York, 1989.
- [2] Williams, F.A. *Combustion Theory*, second edition, Addison-Wesley: New York, 1985.
- [3] Holden, A.V., Markus, M. & Othmer, H.G. (eds). *Nonlinear Wave Propagation in Excitable Media*, Plenum Press: New York, 1991.
- [4] Keener, J. & Sneyd, J. *Mathematical Physiology*, Springer-Verlag: New York, 1998.
- [5] Barrett, R., Berry, M., Chan, T.F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C. & Van de Vorst, H., *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM: Philadelphia, 1994.
- [6] Aykanat, C., Özgüner, F. & Scott, D.S. Vectorization and parallelization of the conjugate gradient algorithm on hypercube-connected vector processors, *Microprocess. Microprogram.*, **29**, pp. 67–82, 1990.
- [7] Chronopoulos, A. T. & Gear C. W. S-step iterative methods for symmetric linear systems, *J. comput. Appl. Math.*, **25**, pp. 153–168, 1989.
- [8] Van der Wijngaart, R.F. Efficient Implementation of a 3-Dimensional ADI Method on the iPSC/860, *Supercomputing'93*, pp. 102–111, 1993.