# PARALLEL COMPUTING SPEEDUPS FOR ALTERNATING DIRECTION COLLOCATION

Mark C. Curran Myron B. Allen III

1989
Proceedings WWRC-89-47

In

Finite Element Analysis in Fluids: Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems

Submitted by

Mark C. Curran and Myron B. Allen III
Department of Mathematics
University of Wyoming
Laramie, Wyoming

# FINITE ELEMENT ANALYSIS IN FLUIDS

*Proceedings of the Seventh International Conference on Finite Element Methods in Flow Problems*

APRIL 3 – 7, 1989

The University of Alabama in Huntsville
Huntsville, Alabama

*T. J. Chung, and Gerald R. Karr, Editors*

UAH PRESS
DEPARTMENT OF MECHANICAL ENGINEERING
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE
HUNTSVILLE, AL 35899

MATHEMATICS OF FINITE ELEMENTS



# PARALLEL COMPUTING SPEEDUPS FOR ALTERNATING DIRECTION COLLOCATION

Mark C. Curran and Myron B. Allen III
Department of Mathematics, University of Wyoming
Laramie, WY 82070 U.S.A.

## ABSTRACT

We apply finite-element collocation to the two-dimensional advection-diffusion equation. Collocation offers savings over other finite-element techniques in that matrix elements are found by point evaluations rather than integrations. Additional computer time and storage is saved by application of an alternating direction process, which allows a multidimensional problem to be solved as a sequence of one-dimensional problems. Since these one-dimensional problems are independent, the speed of the method is enhanced further through use of a parallel computing architecture.

## 1. INTRODUCTION

Alternating direction (AD) methods have been formulated for the numerical solution of partial differential equations since their introduction in 1955 by Peaceman and Rachford [1]. In 1970 Douglas and Dupont [2] developed the alternating direction Galerkin method. More recently, the alternating direction collocation (ADC) method has appeared in several formulations by Bangia et al. [3], Chang and Finlayson [4], Hayes [5], Celia et al. [6], Celia [7], and Celia and Pinder [8].

We examine Celia's ADC for the two-dimensional advection-diffusion equation. Of special interest here is the amenability of the procedure to implementation on parallel-architecture computers. The paper has the following structure: Section 2 briefly reviews finite-element collocation using a bicubic Hermite basis; Section 3 discusses the AD method applied to collocation; Section 4 concludes the paper with an examination of the method's performance on a parallel computer.

## 2. REVIEW OF FINITE-ELEMENT COLLOCATION

Consider the following problem posed on the spatial domain $\Omega = (a,b) \times (c,d)$:

$$
\begin{aligned}
&(a) \quad \partial_t u + \mathbf{v} \cdot \nabla u - \nabla \cdot (D \nabla u) = 0, \quad (x,y,t) \in \Omega \times (0,\infty), \\
&(b) \quad u(x,y,0) = u_I(x,y), \quad (x,y) \in \Omega, \\
&(c) \quad u(x,y,t) = u_B(x,y,t), \quad (x,y) \in \partial\Omega, \ t \geq 0.
\end{aligned} \tag{1}
$$

Here $\mathbf{v} = \mathbf{v}(x,y)$ represents fluid velocity; $D = D(x,y)$ is a diffusion coefficient, and $u = u(x,y,t)$ represents solute concentration. We apply finite-element collocation to the following semidiscrete analog:

$$u^{n+1} - u^n + k[\mathbf{v}\cdot\nabla u^{n+\theta} - \nabla\cdot(D\nabla u^{n+\theta})] = 0, \tag{2}$$

where integer superscripts indicate time level, $(\cdot)^{n+\theta} \equiv \theta(\cdot)^{n+1} + (1-\theta)(\cdot)^n$, $0 \le \theta \le 1$, and $k$ signifies the time step.

We begin by establishing a grid on $\Omega$. Let $\Delta_x = \{x_i = a + ih_x, i = 0,\ldots,N_x\}$ and $\Delta_y = \{y_j = c + jh_y, j = 0,\ldots,N_y\}$, where $h_x = (b-a)/N_x$ and $h_y = (c-d)/N_y$. The Hermite piecewise cubics on these grids are

$$M_1^3(\Delta_x) = \{f \in C^1(\bar{\Omega}) \mid f|_{[x_{i-1},x_i]} \text{ is cubic, } i = 1,\ldots,N_x\},$$

and similarly for $M_1^3(\Delta_y)$. As Prenter [9] shows, each of these spaces has an interpolating basis $\{h_{0i}, h_{1i}\}_{i=0}^{N_x \text{ or } N_y}$, every element of which has support confined to at most two adjacent subintervals $[x_{i-1},x_i]$ or $[y_{j-1},y_j]$.

At each time level $n$ we compute an approximate solution $\hat{u}^n(x,y)$ belonging to the tensor-product trial space

$$M = \{v \in M_1^3(\Delta_x) \otimes M_1^3(\Delta_y) \mid v(x,y) = u_B(x,y) \text{ for } (x,y) \in \partial\Omega\}.$$

Each function in $M$ obeys the boundary conditions (1c) and has the form

$$\begin{aligned}\hat{u}^n(x,y) \;=\; \sum_{i=0}^{N_x}\sum_{j=0}^{N_y}\Bigg[&\hat{u}^n(x_i,y_j)H_{00ij}(x,y) + \frac{\partial\hat{u}^n}{\partial x}(x_i,y_j)H_{10ij}(x,y)\\ &+\frac{\partial\hat{u}^n}{\partial y}(x_i,y_j)H_{01ij}(x,y) + \frac{\partial^2\hat{u}^n}{\partial x\partial y}(x_i,y_j)H_{11ij}(x,y)\Bigg],\end{aligned}$$

where $H_{\ell mij}(x,y) = h_{\ell i}(x)h_{mj}(y)$. At $t = 0$ we form $\hat{u}^0$ by projecting the initial function $u_I$ onto $M$. These criteria specify $\hat{u}^0$ completely and determine $4(N_x + N_y + 1)$ of the $4(N_x+1)(N_y+1)$ nodal coefficients for $\hat{u}^1, \hat{u}^2, \ldots$.

To determine the remaining $4N_xN_y$ degrees of freedom at each time level $n+1$, we first form the residual

$$R^{n+1} = \hat{u}^{n+1} - \hat{u}^n + k\left[\mathbf{v}\cdot\nabla\hat{u}^{n+\theta} - \nabla\cdot(D\nabla\hat{u}^{n+\theta})\right].$$

We then pick a collection $\{(\bar{x}_1,\bar{y}_1),(\bar{x}_1,\bar{y}_2),\ldots,(\bar{x}_{2N_x},\bar{y}_{2N_y})\}$ of collocation points and force $R^{n+1}(\bar{x}_k,\bar{y}_\ell) = 0$ at each. To obtain optimal $O((h_x+h_y)^4)$ error estimates, we choose $\bar{x}_k$ and $\bar{y}_\ell$ to be the two-point Gauss-quadrature abscissae on each subinterval $[x_{i-1},x_i]$ or $[y_{j-1},y_j]$.

## 3. THE ALTERNATING DIRECTION METHOD

To obtain a matrix that can be factored into AD form, we first perturb Equation (2) by a term that is $O(k^2)$ to get

$$u^{n+1} - u^n + k(\mathcal{L}_x + \mathcal{L}_y)u^{n+\theta} + k^2\theta^2(\mathcal{L}_x\mathcal{L}_y)(u^{n+1} - u^n) = 0, \tag{3}$$



where

$$\mathcal{L}_x = v_x\partial_x - \partial_x(D\partial_x) \text{ and } \mathcal{L}_y = v_y\partial_y - \partial_y(D\partial_y).$$

Rearranging Equation (3) and factoring gives

$$(1 + k\theta\mathcal{L}_y)(1 + k\theta\mathcal{L}_x)(u^{n+1} - u^n) = -k(\mathcal{L}_x + \mathcal{L}_y)u^n.$$

Now we can solve $(1 + k\theta\mathcal{L}_y)z = -k(\mathcal{L}_x + \mathcal{L}_y)u^n$, followed by $(1 + k\theta\mathcal{L}_x)(u^{n+1} - u^n) = z$.

When we substitute Hermite bicubic trial functions for $\hat{u}$, we get a matrix equation $K\mathbf{u}^{n+1} = \mathbf{r}^n$, where $\mathbf{u}^{n+1}$ is the vector of time increments for the unknown nodal coefficients of $\hat{u}$. Consider a typical entry of $K$:

$$\left\{\left[1 + k\theta(\mathcal{L}_x + \mathcal{L}_y) + k^2\theta^2(\mathcal{L}_x\mathcal{L}_y)\right]H_\sigma\right\}(\bar{x}_k,\bar{y}_\ell), \tag{4}$$

where $H_\sigma$ is shorthand for some basis function $H_{\ell mij}$. Each $H_\sigma(x,y) = h_\alpha(x)h_\beta(y)$, with $\alpha = (i,r)$ and $\beta = (j,s)$, so we can expand the expression (4) and factor it to get

$$[h_\alpha(\bar{x}_k) + k\theta(\mathcal{L}_x h_\alpha)(\bar{x}_k)]\cdot[h_\beta(\bar{y}_\ell) + k\theta(\mathcal{L}_y h_\beta)(\bar{y}_\ell)].$$

If we number the nodes along the lines $x = \bar{x}_k$, we can use this observation to factor the $4N_xN_y \times 4N_xN_y$ matrix $K$ as follows:

$$K = Y\cdot X = \begin{bmatrix} Y_{1,1} & & \\ & \ddots & \\ & & Y_{2N_x,2N_x}\end{bmatrix}\cdot\begin{bmatrix} X_{1,1} & \cdots & X_{1,2N_y}\\ \vdots & & \vdots \\ X_{2N_y,1} & \cdots & X_{2N_y,2N_y}\end{bmatrix}.$$

Each $2N_y \times 2N_y$ block $Y_{j,j}$ has the five-band structure of a one-dimensional collocation matrix, and its entries depend only on the $y$-coordinates of collocation points.

We can solve the matrix equation $K\mathbf{u}^{n+1} = \mathbf{r}^n$ by the following procedure.

1. Order the nodes vertically and solve $Y\mathbf{z} = \mathbf{r}^n$ by solving the independent problems $Y_{j,j}\mathbf{z}_j = \mathbf{r}_j^n$, $j = 1,\ldots,2N_x$.
2. Reorder the nodes horizontally to convert $\mathbf{z}$ to $\mathbf{z}^*$. This operation transforms $X$ to a block-diagonal form $X^*$ whose blocks $X_{i,i}^*$ have one-dimensional structure.
3. Solve $X^*\mathbf{u}^{n+1} = \mathbf{z}^*$ by solving the independent systems $X_{i,i}^*\mathbf{u}_i^{n+1} = \mathbf{z}_i^*$, $i = 1,\ldots,2N_y$.

Each of the "one-dimensional" systems in steps 1 and 3 is independent of any other. Therefore these steps can be done concurrently.

## 4. IMPLEMENTATION ON A PARALLEL COMPUTER

We have implemented ADC on an Alliant FX/8 parallel processing computer. The Alliant is an eight-processor, shared-memory machine with optimization capability for both concurrent and vector programming. The machine allows users to control concurrency within a Fortran code through the use of compiler directives. The following is a description of the code outlined in Steps 1–3 of Section 3. The compiler directives themselves begin with the flag CVD$ starting in the first column of code.

```
      Initialize û^0, set n = 0
      Begin time level n + 1
CVD$L   CNCALL (Compiler directive to permit the concurrent execution of the following loop
                containing a reference to an external procedure.)
        DO for each j = 1,...,2N_x
          CALL YSWEEP (Constructs the system Y_j z_j = r_j^n, solves it and saves the results.)
        END DO
        CALL RENUM (Reorders z to get z^n)
CVD$L   CNCALL
        DO for each i = 1,...,2N_y
          CALL XSWEEP (constructs the system X_i^n u_i^{n+1} = z_i^n, solves it and updates the
                       appropriate coefficients of û to time level n + 1.)
        END DO
      End time step
        ⋮
CVD$R NOCONCUR (Directive to supress concurrency until the end of the routine.)
      SUBROUTINE YSWEEP
CVD$R NOCONCUR
      SUBROUTINE XSWEEP
```

One measure of how well the algorithm makes use of the machine's parallel capabilities is the *speedup*. Speedup for $n$ processors is the ratio of the time needed by one processor to the time used by $n$ processors to perform a set of tasks in parallel. For a perfectly parallel algorithm requiring no overhead to monitor and schedule the various processors, the speedup for $n$ processors would be $n$. Figure 1 shows the speedup curve for this algorithm, excluding initialization. The speedup for eight processors is 7.27. Clearly, ADC makes very good use of the shared-memory parallel architecture.

To confirm that ADC gives useful approximations, Figures 2 and 3 show solution plots for two different problems. Figure 2 shows the results of a rotating plume problem on $\Omega = (-1,1) \times (-1,1)$, with $N_x = N_y = 40$ and $k = 0.004$. Here, $\mathbf{v} = 2\pi(-y,x)$ is a circular velocity field, $D = 0$, and the initial concentration plume $u_I(x,y)$ is a "Gauss hill" with center at $(0,-0.6)$ and standard deviation $\sigma = 0.066$. Figure 3 displays the results of an advection-diffusion problem on $\Omega = (0,1) \times (0,1)$, with $N_x = N_y = 20$ and $k = 0.004$. The diffusion coefficient is $D = 0.00385$, and $\mathbf{v}(x,y) = 2e^{xy}(-y,x)$. Here, $u_I$ is a "Gauss hill" with $\sigma = 0.05$ centered at $(0.75, 0.25)$. In both problems the global error is less than $.02\|u\|_\infty$.

## ACKNOWLEDGMENTS

The Wyoming Water Research Center supported this work. We also received support from NSF grant RII-8610680 and ONR contract 0014-88-K-0370.

## REFERENCES

1. Peaceman, D.W. and H.H. Rachford, "The Numerical Solution of Parabolic and Elliptic Equations," *SIAM J.*, *3*, 28–41 (1955).



2. Douglas, J., Jr. and T. Dupont, "Alternating-Direction Galerkin Methods on Rectangles," in *Numerical Solution of Partial Differential Equations, Vol. 2*, Synspade 1970, B. Hubbard, Ed., Academic, New York, 1971, pp. 133–214.
3. Bangia, V.K., C. Bennett, and A. Reynolds, "Alternating Direction Collocation for Simulating Reservoir Performance," presented at the 53rd Annual Fall Conference, Society of Petroleum Engineers, Houston, 1978.
4. Chang, P.W. and B.A. Finlayson, "Orthogonal Collocation on Finite Elements for Elliptic Equations," *Math. Comp. Simulation*, 83–92, (1978).
5. Hayes, L.J. "An Alternating-Direction Collocation Method for Finite Element Approximations on Rectangles," *Comput. Math. Appl.*, *6*, 45–50, (1980).
6. Celia, M.A., G.F. Pinder, and L.J. Hayes, "Alternating Direction Collocation Simulation of the Transport Equation," *Proceedings Third Int. Conf. Finite Elements in Water Resources*, S.Y. Wang *et al.*, Eds., University of Mississippi, Oxford, MS, 1980, pp. 3.36–3.48.
7. Celia, M.A., *Collocation on Deformed Finite Elements and Alternating Direction Collocation Methods*, Ph.D. Dissertation, Princeton University, 1983.
8. M.A. Celia and G.F. Pinder, "Generalized Alternating-Direction Collocation Methods for Parabolic Equations: 1. Spatially Varying Coefficients," (1984).
9. Prenter, P.M., *Splines and Variational Methods*, New York: Wiley, 1975, Chapter 3.

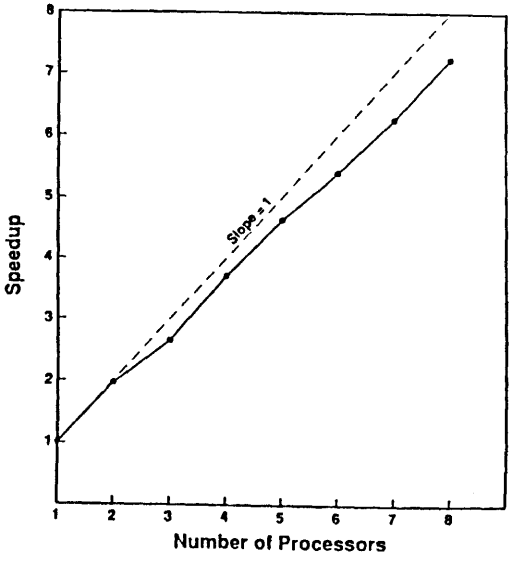


**Figure 1.** Speedup curve for ADC using the Alliant FX/8 shared-memory architecture.

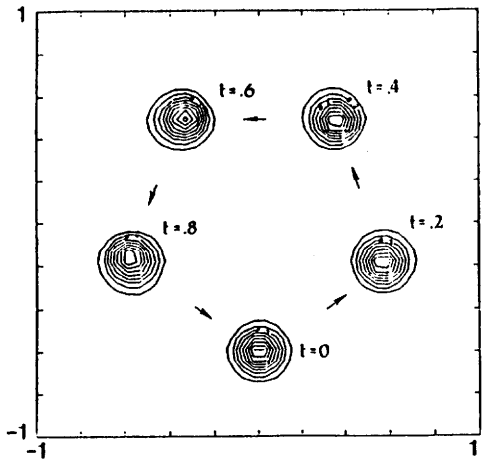


**Figure 2.** Concentration contours for the purely advective rotating plume problem at various time levels. Contour interval is 0.1.

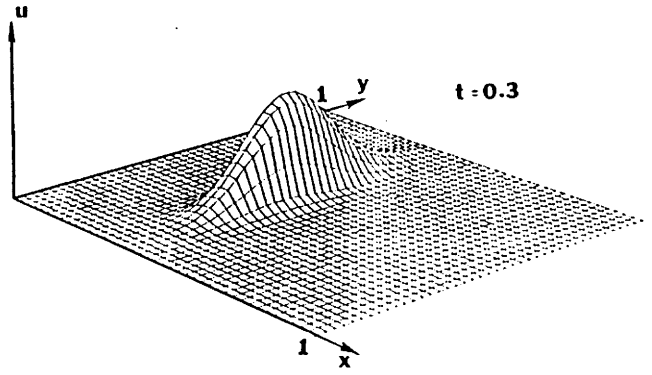


**Figure 3.** Plot of concentration distribution at $t = 0.3$ for an advection-diffusion problem with potential flow.