Proceedings of the XXVI Iberian Latin-American Congress on Computational Methods in Engineering CILAMCE 2005
Brazilian Assoc. for Comp. Mechanics & Latin American Assoc. of Comp. Methods in Engineering
Guarapari, Espírito Santo, Brazil, 19^{th}–21^{st} October 2005

Paper CIL16-692

NUMERICAL EVALUATION OF THE LCD METHOD IMPLEMENTED IN THE LibMesh LIBRARY

José J. Camata
Lucia Catabriga
Andrea M. P. Valli
jcamata@inf.ufes.br
luciac@inf.ufes.br
avalli@inf.ufes.br
Computer Science Department, Federal University of Espirito Santo
Av. Fernando Ferrari, s/n, 29060-970 ,Vitória - ES - Brazil

Alvaro L.G.A. Coutinho
alvaro@nacad.ufrj.br
Department of Civil Engineering, COPPE, Federal University of Rio de Janeiro
PO Box 68506, 21945-970, Rio de Janeiro - RJ - Brazil

Graham F. Carey
carey@cfdlab.ae.utexas.edu
Institute for Computational Engineering and Sciences, The University of Texas at Austin
201 E. 24th St., ACES 6.430, 1 University Station C0200, 78712-0227, Austin - TX - USA

Abstract. *In this work we evaluate the performance of the left conjugate direction method (LCD) for the solution of non-symmetric systems of equations arising from finite element simulation of the steady convection-diffusion equation. The restarted LCD algorithm is implemented in the LibMesh library and computational efficiency of this new method and two Krylov solvers available in the library (GMRES and Bi-CGSTAB) are compared. The test problems considered correspond to a low speed flow problem with diffusion and a convection dominated problem.*

Keywords: *Left conjugate direction method, LibMesh library, Finite element*

1. INTRODUCTION

Numerical strategies for flow problems in science and engineering often require repeated solution of large sparse nonlinear systems of equations. The associated linearized subproblems are usually solved by Krylov subspace iterative methods (e.g. see Saad (1996)). Yuan et al. (2004) introduced a new algorithm for solving nonsymmetric, nonsingular linear systems, the Left Conjugate Direction (LCD) method. This method is based on the concept of left and right conjugate vectors for nonsymmetric and nonsingular matrices and possesses several theoretical advantages: (i) it has a finite termination property; (ii) breakdown for general matrices can be avoided and (iii) there is a connection between LCD and LU decomposition. Initial experiments to solve some sample linear systems arising from linear partial differential equations are presented in Yuan et al. (2004). Using a MATLAB implementation, they have shown that the LCD method has attractive convergence rates when compared to Bi-CGSTAB, QMR and GMRES methods.

Catabriga et al. (2004) evaluated the performance of the original LCD algorithm for the solution of nonsymmetric systems of linear equations arising from the implicit semi-discrete SUPG finite element formulation for inviscid compressible flows described in Catabriga and Coutinho (2002). They extended the original algorithm to accommodate restarts and typical finite element preconditioners. In these studies, for that application class, comparisons with other Krylov space methods with or without preconditioning unfortunately did not favour the LCD method. Although requiring usually fewer iterations, CPU times and memory were larger than GMRES, Bi-CGSTAB and TFQMR algorithms. The main reason is the need to compute two matrix-vector products per iteration, one with the coefficient matrix and the other with its transposed matrix.

Recently, Dai and Yuan (2004) proposed a new technique to overcome the breakdown problem appearing in the semi-conjugate direction method and a memory limitation scheme similar to the limited-memory BFGS method to minimize memory requirements of the original algorithm. Catabriga et al. (2005) introduced a restart strategy for the new LCD algorithm given by Dai and Yuan (2004) and compared it with the restarted LCD algorithm given by Catabriga et al. (2004) and the restarted GMRES method for the solution of linear and nonlinear problems discretized by finite element and finite difference methods. They observed that in some cases the LCD algorithm was faster than GMRES. In this work we evaluate the performance of the LCD scheme for the solution of non-symmetric systems of equations arising in finite element simulation of that steady convection-diffusion problem. This scheme in implemented in the LibMesh library and compared with GMRES and Bi-CGSTAB with ILU(0) preconditioning. The present study is confined to serial simulations on structured meshes (LibMesh permits parallel adaptive unstructured mesh simulations).

The remainder of this work is organized as follows. In the next section we briefly comment on LibMesh and associated solvers. Section 3 introduces the LCD algorithm with particular emphasis on the restart capability. This is followed by several numerical experiments, where we compare the performance of LCD with GMRES and Bi-CGSTAB methods.

2. LibMesh LIBRARY

The LibMesh (2005) library is a tool for numerical simulation of partial differential equations on serial and parallel platforms, using the finite element method and developed in C++. It provides a C++ interface to the user, simplifying many programming details. LibMesh allows discretization of one, two and three dimensional transient problems using several element types. A major goal of LibMesh is to provide support for Adaptive Mesh Refinement (AMR). LibMesh includes interfaces for standard high performance linear equation solvers libraries such as PETSc (2005) and LASPack (2005). The choice of appropriate solvers is made by the user at runtime.

PETSc and LASPack packages are integrated into LibMesh providing several linear equation solvers such as GMRES, CG, Bi-CGSTAB, QMR. The library allows the combination of a Krylov subspace iterative method and a preconditioner. Jacobi, Incomplete LU factorization and Incomplete Cholesky factorization are examples of preconditioners found in the library.

3. GOVERNING EQUATIONS AND DISCRETE FORMULATION

We consider the following linear convection-diffusion equation defined in a domain Ω with boundary Γ:

$$\boldsymbol{\beta} \cdot \nabla u - \nabla.(\boldsymbol{\kappa} \nabla u) = f, \tag{1}$$

$$u = g \quad \text{on} \quad \Gamma_g, \tag{2}$$

$$\boldsymbol{n} \cdot \kappa \nabla u = h \quad \text{on} \quad \Gamma_h, \tag{3}$$

where u represents the quantity being transported (e.g. temperature, concentration), $\boldsymbol{\beta}$ is the divergence-free flow velocity and $\boldsymbol{\kappa}$ is the volumetric diffusivity. Equations (2) and (3) are the essential and natural boundary conditions, respectively, g and h are given functions of $\boldsymbol{x} = (x, y)$, and $\boldsymbol{n}$ is the unit outward normal vector at the boundary, Γ_g and Γ_h are the complementary subsets of Γ where boundary conditions are prescribed.

Consider a finite element discretization of Ω into elements Ω_e, $e = 1, \ldots, n_{el}$, where n_{el} is the number of elements. The stabilized finite element formulation of Brooks and Hughes (1982), leads to a system of linear equations,

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4}$$

where $\boldsymbol{x}$ is the vector of nodal values of u, $\boldsymbol{A}$ is a nonsymmetric sparse matrix and $\boldsymbol{b}$ is a vector accounting for sources/sinks and boundary contributions.

The nonsymmetric linear systems are solved using the LCD method implemented in LibMesh and two linear solvers available there (GMRES and Bi-CGSTAB methods). The LCD algorithm introduced in the library is the algorithm proposed by Catabriga et al. (2005) which accommodates restarts and typical finite element preconditioners. In this LCD version, we need only one matrix vector product as in the GMRES algorithm, minimizing memory requirements of the original algorithm in Yuan et al. (2004). The LCD algorithm for nonsymmetric system (4) can be written as follows:

1. *Given* x, A, b, l_{max}, k_{max} *and* η
2. $r = b - Ax$
3. $\epsilon = \eta \|r\|$
4. *Choose* p_1 *such that* $p_1^T A p_1 \neq 0$
5. *For* $l = 1, \ldots, l_{max}$ *do*
    - 5.1. $q_1 = Ap_1$
    - 5.2. *For* $i = 1, \ldots, k_{max}$ *do*
        - 5.2.1. $\alpha_i = \frac{p_i^T r}{p_i^T q_i}$
          $x = x + \alpha_i p_i$
          $r = r - \alpha_i q_i$
        - 5.2.2. *if* $\|r\| < \epsilon$ *then exit loops* i *and* l, x *is the solution; else*
        - 5.2.3. $p_{i+1} = r$
          $q_{i+1} = Ap_{i+1}$
          For $j = 1, \ldots, i$ *do*
            $\beta_j = -\frac{p_j^T q_{i+1}}{p_j^T q_j}$
            $p_{i+1} = p_{i+1} + \beta_j p_j$

$$q_{i+1} = q_{i+1} + \beta_j q_j$$

5.3. *choose the new* p_1 *such that* $p_1^T A p_1 \neq 0$

where l_{max} is the maximum number of iterations, k_{max} is the number of left conjugate directions considered in the restart and η is user a supplied tolerance. To start LCD, we have to choose p_1 in step 4 and the subsequent values of p_1, in step 5.3, for each iteration. Catabriga et al. (2004) report numerical experiments concerning this choice for the compressible flow simulations. The best results were $p_1 = r$ for $l = 1$ and $p_1 = p_{k_{max}+1}$ for $l + 1$ so in this work we adopt this choice. For all solvers we use an ILU(0) preconditioner and the LCD method was implemented using the same functions (for instance, matrix-vector products, inner products and preconditioner calculations) used in GMRES and Bi-CGSTAB algorithms and available on the LASPack and PETSc libraries.

4. NUMERICAL RESULTS

4.1 Linear convection-diffusion problem

Let us consider the linear convection-diffusion equation (1)-(2) defined in a unit square domain Ω = (0,1) × (0,1) with boundary Γ. The diffusivity coefficients are $k_x = 1$ and $k_y = 1$ and $\boldsymbol{\beta}$ is a specified velocity vector given by

$$\beta_x = x^2(1-x)^2(2y - 6y^2 + 4y^3) \quad (5)$$

$$\beta_y = y^2(1-y)^2(-2x + 6x^2 - 4x^3) \quad (6)$$

and body force f is constructed such that $u(x, y) = 100xy(x - 1)(y - 1)$ is the exact solution with $g = 0$ on boundary Γ. The domain is discretized by grids of three-node (TRI3) and six-node (TRI6) triangular elements (with 64×64, 128×128 and 256×256 cells, and each cell is subdivided into four triangles). For all cases, we consider a relative residual tolerance of 10^{-10}.

First, we solve the problem using different numbers of restart vectors in the GMRES and LCD algorithms to observe the number of iterations and CPU time (in seconds) needed for convergence. We consider 5, 10, 20, 30 and 40 restart vectors in both algorithms. In most of the cases, the best performance of GMRES and LCD methods occur when 40 and 10 vectors are used for restart, respectively. As an example of the numerical experiments performed in this analysis, we show in Table 1 the number of iterations and CPU time (in seconds) to solve the convection-diffusion problem on the mesh with 256 × 256 cells. Bi-CGSTAB has an order of magnitude fewer iterations and, in the case shown, gives the best results in terms of CPU time. Depending on the type of element used, Bi-CGSTAB is up to 4 times faster than GMRES and LCD. When we compare the LCD and GMRES methods, we can see that the LCD method presents better results in most of the cases. The difference between the performance of the methods increases when we increase the number of cells in the mesh.

Now, for 40 restart vectors we calculate the dominant computational costs in the algorithms. Table 2 shows the cost of *inner products*, *matrix vector products* and *preconditioner calculations* on the mesh with 256 × 256 cells with linear (TRI3) and quadratic (TRI6) triangular elements. The preconditioner calculations are the most costly task in Bi-CGSTAB and GMRES, and the inner products are the most costly task in LCD. Figure

Table 1: Performance of GMRES, LCD and Bi-CGSTAB methods for the convection-diffusion problem - Mesh 256×256 cells

	GMRES				**LCD**			
	TRI3		***TRI6***		***TRI3***		***TRI6***	
Vec.	*CPU Time*	*Iter.*	*CPU Time*	*Iter.*	*CPU Time*	*Iter.*	*CPU Time*	*Iter.*
5	109.17	5037	1146.5	10000	10.124	328	321.54	2107
10	57.673	2604	981.27	8657	11.2	296	302.58	1718
20	36.202	1403	486.67	3843	13.349	251	362.57	1525
30	24.504	812	464.29	3243	17.28	251	411.72	1368
40	20.994	612	402.01	2506	21.277	251	506.69	1381

	Bi-CGSTAB - *TRI3*		**Bi-CGSTAB - *TRI6***	
	CPU Time	*Iter.*	*CPU Time*	*Iter.*
	5.3867	150	77.983	407

1 shows the residual behavior for TRI3 and TRI6 elements considering the mesh with 256×256 cells and 40 vectors for the restart. The relative residual in LCD and GMRES decreases more slowly than Bi-CGSTAB. LCD converges with fewer iterations than GMRES. The behavior of the residuals for the other cases are similar.

Table 2: Computational cost for the convection-diffusion problem - Mesh 256×256 cells

GMRES(40)	**TRI3**		**TRI6**	
Operations	**CPU time**	% cost	**CPU time**	% cost
Inner product	5.0166	23.9%	81.199	20.2%
Matrix-Vector	2.9444	14.0%	65.166	16.2%
Preconditioner	5.2279	24.9%	125.060	31.1%
LCD(40)	**TRI3**		**TRI6**	
Operations	**CPU time**	% cost	**CPU time**	% cost
Inner product	7.2909	34.26%	145.710	28.75%
Matrix-Vector	1.1585	5.40%	35.588	7.00%
Preconditioner	2.1926	10.30%	69.130	13.60%
Bi-CGSTAB	**TRI3**		**TRI6**	
Operations	**CPU time**	% cost	**CPU time**	% cost
Inner product	0.31934	5.9%	3.5611	4.5%
Matrix-Vector	1.3786	25.6%	20.375	26.1%
Preconditioner	2.4801	46.0%	39.686	50.9%

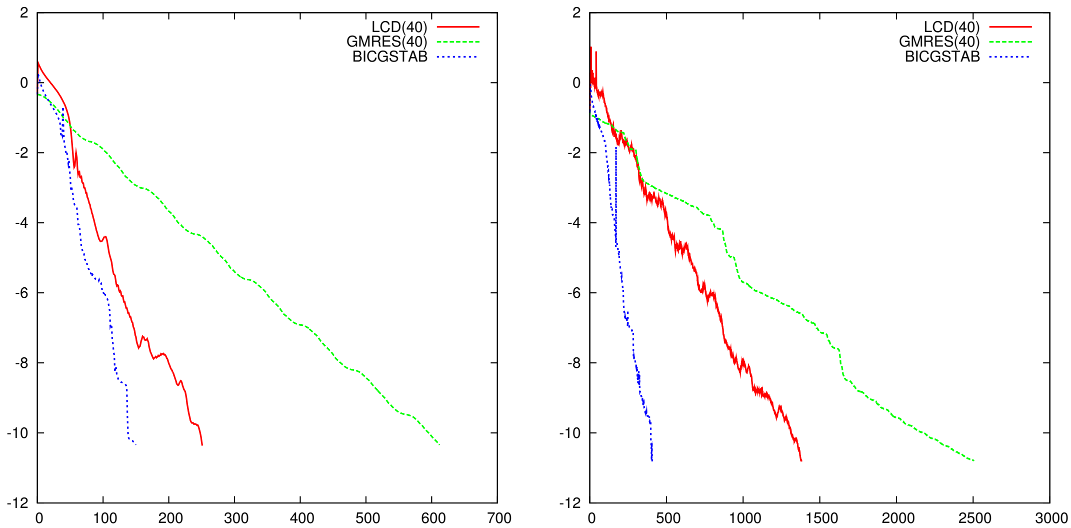

Figure 1: Relative residual evolution (Iterations $\times$ $log(||r||)$) for TRI3 (left) and TRI6 (right) elements, mesh 256 $\times$ 256 cells - convection-diffusion problem

4.2 Convection-dominated problem

Consider the common test problem of convection-dominated transport of a scalar on a unit square domain, where convection is skew to the mesh and diffusivity is negligible. The problem set up is given in Fig. 2 for boundary conditions $u = 0$ along $y = 0$, $u = 0$ along $x = 0$ and $0 < y < 0.25$, and $u = 1$ along $x = 0$ and $0.25 < y < 1.0$. The flow direction is 45^o from the x-axis, constant ($||\beta||$ = 1), the diffusivity coefficients

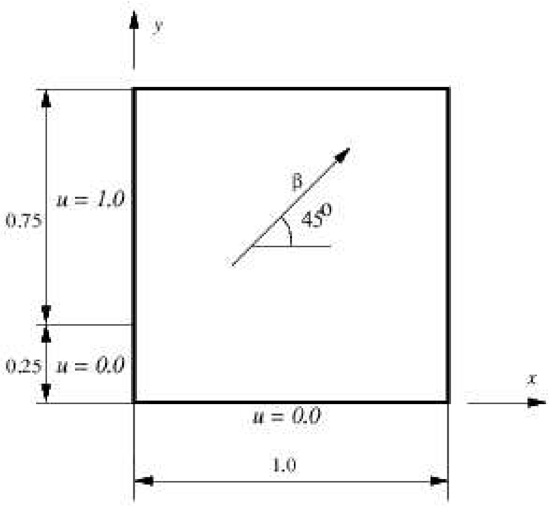

Figure 2: Problem set up - Convection-dominated problem

are $k_x = 10^{-7}$ and $k_y = 10^{-7}$ and the solution has an interior layer emanating. The stabilization parameter for the SUPG finite element discretization is determined as in Brooks and Hughes (1982). As in the first example, the domain is discretized by linear

(TRI3) and quadratic (TRI6) triangular elements (with 64×64, 128×128 and 256×256 cells, and each cell is subdivided into four triangles). For all cases, we consider a relative residual tolerance of 10^{-10}.

Table 3 shows the CPU time (in seconds) and the number of iterations to solve this problem for the mesh with 128 × 128 cells. The results for the other meshes are similar and therefore are not shown here for conciseness. In this second example, we observe that the GMRES method presents the smallest CPU time for all cases. Bi-CGSTAB has better performance than LCD for linear elements.

Table 3: Performance of GMRES, LCD and Bi-CGSTAB methods for convection-dominated problem - Mesh 128 × 128 cells

	GMRES				**LCD**			
	TRI3		***TRI6***		***TRI3***		***TRI6***	
Vec.	*CPU Time*	*Iter.*	*CPU Time*	*Iter.*	*CPU Time*	*Iter.*	*CPU Time*	*Iter.*
5	0.058874	9	8.3921	325	0.07265	9	10.875	327
10	0.061265	9	10.435	374	0.08657	9	18.133	430
20	0.065397	9	14.856	480	0.091874	9	27.969	481
30	0.066153	9	19.729	564	0.10035	9	41.007	569
40	0.066056	9	25.562	657	0.1037	9	58.142	664

	Bi-CGSTAB - *TRI3*		**Bi-CGSTAB - *TRI6***	
	CPU Time	*Iter.*	*CPU Time*	*Iter.*
	0.061741	5	9.8254	217

Table 4 shows the computational cost of *inner products*, *matrix vector products* and *preconditioner calculations* for the case studied above. In this example, preconditioner calculations are the most costly task for all methods, followed by matrix-vector products and inner products. This behavior is mainly due to the number of restart vectors (5) used in this case. Figure 3 shows the residual behavior for TRI3 and TRI6 elements using 5 restart vectors and the mesh with 128×128 cells. For linear triangular elements, we observe that the relative residual in GMRES and LCD are similar and Bi-CGSTAB converges with the smallest number of iterations. For quadratic triangular elements, the relative residual in Bi-CGSTAB oscillates and increases in the beginning of the process. However, just before 200 iterations the relative residual falls abruptly.

5. CONCLUDING REMARKS

In this work we include the restarted left conjugate direction method (LCD) in the LibMesh library and perform comparison studies between the computational efficiency of this new method and two Krylov solvers available in the library, GMRES and Bi-CGSTAB. For all solvers we use the incomplete LU factorization preconditioner available in the library and the LCD method was implemented using the same functions as in the GMRES algorithm and available on the PETSc library. We solve a convection-diffusion transport problem and a convection-dominated problem implemented in the LibMesh li-

Table 4: Computational cost for convection-dominated problem - Mesh 128×128 cells

GMRES(5)	**TRI3**		**TRI6**	
Operations	**CPU time**	% cost	**CPU time**	% cost
Inner product	0.00233	3.5%	0.3780	4.27%
Matrix-Vector	0.01138	17.3%	2.3727	26.80%
Preconditioner	0.02700	41.0%	4.2993	48.60%
LCD(5)	**TRI3**		**TRI6**	
Operations	**CPU time**	% cost	**CPU time**	% cost
Inner product	0.00704	9.6%	1.6585	14.30%
Matrix-Vector	0.01266	17.3%	2.2829	19.70%
Preconditioner	0.02146	41.0%	4.3327	37.45%
Bi-CGSTAB	**TRI3**		**TRI6**	
Operations	**CPU time**	% cost	**CPU time**	% cost
Inner product	0.00213	3.4%	0.4367	4.6%
Matrix-Vector	0.01708	27.1%	2.5359	26.8%
Preconditioner	0.02648	35.9%	4.7840	50.6%

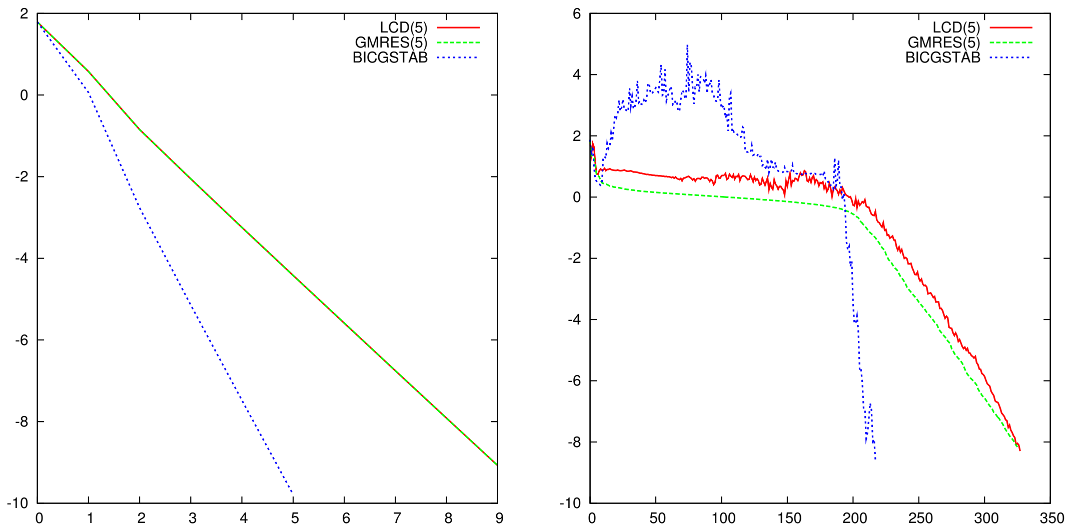

Figure 3: Relative residual evolution (Iterations $\times$ $log(||r||)$) for TRI3 (left) and TRI6 (right) elements, mesh 128×128 cells - Convection-dominated problem

brary on a serial platform. Future plans are to solve more complex 3D problems using parallel adaptive mesh refinement (AMR) with LibMesh.

In the first problem, the best performance of GMRES and LCD methods occurs when 40 and 10 vectors are used for restart, respectively. Bi-CGSTAB has the best results in terms of CPU time. Depending on the type of element used, Bi-CGSTAB is up to 4 times faster than GMRES and LCD. When we compare the LCD and GMRES methods, we see that the LCD method yields better results in most of the cases. For the case of 40 restart vectors, the preconditioner calculations are the most costly task in Bi-CGSTAB and GMRES, and the inner products are the most costly task in LCD. In this case, the relative residuals in LCD and GMRES decrease more slowly than Bi-CGSTAB, and LCD converges with fewer iterations than GMRES.

The second example is a convection dominated problem. In this case, both GMRES and LCD methods present better performance using 5 restart vectors and the GMRES method presents the smallest CPU time for all cases. Bi-CGSTAB has better performance than LCD for linear elements. For 5 restart vectors, preconditioner calculations are the most costly task for all methods, followed by matrix vector products and inner products. This behavior is mainly due to the number of restart vectors used in this case. The relative residual in GMRES and LCD are similar and Bi-CGSTAB converges with the smallest number of iterations.

Acknowledgments

This work is partly supported by CAPES, under Grant CAPES/UT N° 11/04 and by CNPq, under Grant 200649/2005-1.

REFERENCES

Brooks, A. & Hughes, T., 1982. Streamline Upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. and Engrg.*, vol. 32, pp. 199–259.

Catabriga, L., Coutinho, A., & ca, L. F., 2004. Evaluating the lcd algorithm for solving linear systems of equations arising from implicit SUPG formulation of compressible flows. *International Journal for Numerical Methods in Engineering*, vol. 60, pp. 1513–1534.

Catabriga, L. & Coutinho, A. L. G. A., 2002. Implicit SUPG solution of Euler equations using edge-based data structures. *Comput. Methods Appl. Mech. and Engrg.*, vol. 191, pp. 3477–3490.

Catabriga, L., Valli, A., Melotti, B., Pessoa, L., & Coutinho, A., 2005. Performance of lcd iterative method in the finite element and finite difference solution of convection-diffusion equations. *Communications in Numerical Methods in Engineering (to appear)*, vol. .

Dai, Y. & Yuan, J. Y., 2004. Study on semi-conjugate direction methods for non-symmetric systems. *International Journal for Numerical Methods in Engineering*, vol. 60, pp. 1383–1399.

LASPack, visited 08-02-2005. Laspack home page. http://www.mgnet.org/mgnet/Codes/laspack/html/laspack.htm.

LibMesh, visited 08-02-2005. Libmesh home page. http://libmesh.sourceforge.net/.

PETSc, visited 08-02-2005. Pestc home page. http://www-unix.mcs.anl.gov/petsc/petsc-2/.

Saad, Y., 1996. *Iterative Methods for Sparse Linear Systems*. PWS Publishing, Boston.

Yuan, J., Golub, G., Plemmons, R., & Cecílio, W. A. G., 2004. Semiconjugate direction methods for real positive definite systems. *BIT Numerical Mathematics*, vol. 44(1), pp. 189–207.