ADAPTIVE RITZ BLOCK LANCZOS FOR ALGEBRAIC SYSTEMS ARISING FROM FLUID FLOW PROBLEMS

Pedro Torres

ptorres@pol.una.py

Christian E. Schaerer

cschaer@pol.una.py

San Lorenzo, Paraguay. P.O. Box: 2111 SL

Polytechnic School, National University of Asuncion

Campus Universitario, 2111 SL, San Lorenzo, Central, Paraguay.

Amit Bhaya

amit@nacad.ufrj.br

COPPE, Federal University of Rio de Janeiro

Ilha Universitaria, Rio de Janeiro, Brazil.

Abstract. *Block Methods based on conjugate directions are promising for efficient implementation on new computational architectures, improving the arithmetic intensity and the rate of convergence with respect to classical ones. In this article the Block Lanczos method - BLM, previously introduced for computing eigenvalues, is used for the resolution of algebraic linear systems $Ax = b$ with A symmetric positive definite. A drawback of the method lies in the adequate choice of block size. This article, propose a rule that adaptively updates the size of the block in the BCG algorithm and uses a threshold to determine when the rule acts. The rule is based on indirect measure of the distribution of the eigenvalues by approximating them and their relative clusterization using Ritz values. The resulting method shows an acceleration of the convergence of the algorithm decreasing the computational cost, if an adequate threshold is set. Results for matrices arising from computational fluid problems are presented.*

Keywords: *Block Conjugate Gradient, Ritz Values, Symmetric Positive Definite Problems, Algebraic Linear Systems.*

1 INTRODUCTION

Consider the linear system of equations

$$Ax = b, \tag{1}$$

where A is a large sparse real symmetric positive definite $n \times n$ matrix and x and b are real n-dimensional vectors. A typical algorithm for solving the Eq. (1) is the Lanczos algorithm or its better known version, the Conjugate Gradient (CG) algorithm Hestenes and Stiefel (1952). However, it is well known that due to the necessity of using matrix-vector multiplication routines for building the Krylov subspace, CG the algorithm presents a low arithmetic intensity or, equivalently, a low ratio of computation to memory accesses. This problem gets worse in new computer architectures Gropp et al. (2001, 1999). Consequently, new algorithms are being investigated to optimize the computational resources, as well as, to improve the efficiency of the algorithm in terms of arithmetic intensity for solving linear systems of the form Eq. (1) (see Aktulga et al., 2014; Abu-Sufah and Ahmad, 2014; Ortega et al., 2014; Ghysels et al., 2012; Kislal et al., 2013; Hoemmen, 2010, and references therein).

In this paper, instead of solving Eq. (1) directly , we consider the system

$$AX = B \tag{2}$$

where X, B are $n \times s$ matrices with $s \ll n$. Every column of X and B is denoted by X_i and B_i, respectively. Therefore, if every column $B_i = b$, then solving Eq. (2) is equivalent to solving s linear systems of the form Eq. (1). Convergence of an iterative algorithm for solving Eq. (2) means that $X_i \to x^* = A^{-1}b \; \forall i \in [1, s]$.

Solving Eq. (2) is equivalent to independently solving s linear system of equations. Thus in principle, if the time of resolution of Eq. (2) is comparable with the time of resolution of Eq. (1); then if it is necessary to solve Eq. (1) with s different RHS, there is a clear advantage in using Eq. (2).

Although this kind of problem arises in several engineering applications, in this article we explore the usage of block formulation Eq. (2) for solving Eq. (1) when $B_i = b \; \forall i \in [1, s]$. Several articles have been investigated this approach using block algorithms (see Jianhua and Jing, 2013; Feng et al., 1995; Torres et al., 2015), and it has been observed that increasing the block size s can potentially improve the rate of convergence of X_i to x^*, and additionally reduce the number of accesses to memory (see O'Leary, 1980). However, this strategy can not be used arbitrarily, since a large block size introduces an excessive computational cost. Consequently, it is necessary to establish a compromise between the speed up of the convergence and the computational cost of the algorithm when a block strategy is used. Therefore, a key point consists in choosing a size block s for obtaining a good compromise between the arithmetic intensity and the rate of convergence. This work is framed in the context of understanding how to control the variation of the size of the block as the iteration proceeds, in order to improve the rate of convergence of the algorithm.

In this article, we use a strategy based on a adaptive law on the Ritz values to approximate the eigenvalues of matrix A at each iteration step of the BLM for solving $Ax = b$. The rest of the article is structure as follows: in §2 is described the adaptive block Lanczos algorithm, as well as, the adaptive law for s based on the Ritz eigenvalues of matrix T_j and a threshold η. In §3, numerical results for matrices problems extracted from fluid flow problem are presented.

The numerical results shows that the proposed adaptive law helps to improve the performance of the Lanczos method for algebraic symmetric linear systems.

2 THE ADAPTIVE BLOCK LANCZOS

Given an initial guess x^0 for the linear system Eq. (1), the initial residual $r^0 := b - Ax^0$ and the Lanczos vectors v_i, $i = 1, ...m$ together with the tridiagonal matrix T_m, the approximate solution obtained from an orthogonal projection method onto $\mathcal{K}_m := span\{r^0, Ar^0, ..., A^{m-1}r^0\}$ is given by

$$x^m = x^0 + V_m y^m, \qquad y^m = T_m^{-1}(\beta e_1). \tag{3}$$

Analogously to the single case (Eq. (1)), given X^0, a block Krylov subspace method determine at step j, an approximation of the form $X^j = X^0 + DX^j$ where DX^j belongs to the block Krylov subspace of the form $\mathcal{K}_j(A, R^0) = span\{R^0, AR^0, ..., A^{j-1}R^0\}$ with $R^0 = B - AX^0$. Normally, a minimization property or an orthogonalization relation is used to determine the correction DX^j. When A is symmetric, the BLM begins with an initial block of vector Q_1 of dimension $n \times s$ where s is the size of the block. The block Lanczos generates a sequence Q_j of matrices of dimension $n \times s$ by using a block three term recurrence:

$$Q_{j+1}G_{j+1} = W_j = AQ_j - Q_{j-1}G_j^T - Q_j M_j \tag{4}$$

where $M_j = Q_j^T AQ_j$, M_j, G_j are $s \times s$ matrices, and Q_{j+1}, G_{j+1} are obtained from the QR-factorization of W_j with Q_{j+1} orthogonal and G_{j+1} upper triangular for all j.

At each step j, BLM generates a block tridiagonal matrix $T_j = diag(G_j, M_j, G_{j+1})$. Hence T_m is the orthogonal projection of A onto $\mathcal{K}_j(A, R^0)$ which by construction is equal to $span\{Q_1, ..., Q_j\}$. The approximate solution X^j of X^* is obtained from

$$X^j = X^0 + Q^j Y^j, \qquad T_j Y^j = (Q^j)^T B \tag{5}$$

where $Q^j = [Q_1|Q_2|...|Q_j]$ and the block residual is given at each step j by $R^j = B - AX^j$.

The Block Lanczos breaks down if one of the matrices G or M becomes singular or indefinite before the matrix R^k. One strategy to overcome this drawback is to reduce the block size s if any linear dependence arises (Nikishin and Yeremin, 1995). We claim that a strategy based on the approximation of the eigenvalue distribution of A is more intuitive, since the BLM intrinsically approximates the eigenvalues of A at each iteration (see Saad, 2003), and because its convergence behavior is related to the eigenvalue distribution (Van der Sluis and van der Vorst, 1986; Concus et al., 1975).

We adopt the following definition for the Ritz values.

Definition. Given $\mathcal{K}_j = \mathcal{K}_j(A, R^0)$, the real number θ is a Ritz value of A with respect to $\mathcal{K}_j$ with Ritz vector U if $(AU - \theta U) \perp \mathcal{K}_j$, $U \neq 0$.

Recalling that by construction $U = Q^j u$ and using Def. 2 above, we obtain $(AU - \theta U) \perp Q^j$. Consequently, $(Q^j)^T AQ^J u - \theta u = 0$ with $T_j = (Q^j)^T AQ^J$, where θ and u are the Ritz value and Ritz eigenvector, respectively. In this way, at each step of the Block Lanczos, the distribution of eigenvalues of the matrix A can be estimated by using the Ritz values θ.

The block size is important since increasing the block size accelerates the convergence of the Ritz values to the eigenvalues of A and in particular to clustered eigenvalues. Although

increasing the block size also increases the computational complexity of the method. Therefore, it is established a trade off between choosing the adecuate block size and the computational cost, being difficult to know a priori the correct block size.

Cluster eigenvalues affects the convergence of the method and the block size is directly related to the multiplicity of multiple eigenvalues. Following, we measure the clusterization we use the following expression (Ye, 1996):

$$\frac{\theta_{i+k-1} - \theta_i}{min\{\theta_i - \theta_{i-1}, \theta_{i+k} - \theta_{i+k-1}\}} \leq \eta, \tag{6}$$

where θ_j are the Ritz values at iteration j, and k is the size of the cluster being equal to the block size s_j. This condition is to test whether the Ritz values form a cluster of size k based on the threshold η. An increment of the block size would lead to a better approximation of the Ritz values (see Golub and Van Loan, 2012, p. 568), improving the convergence rate of the BLM algorithm (Van der Sluis and van der Vorst, 1986; Concus et al., 1975) . We also allow the Algorithm 1 to freely adjust the size of the block size, and in some cases the size is increased. The latter is justified by the observation in experimental results that in early iterations, a larger block size does not contribute to the convergence.

Algorithm 1 Variable Block Lanczos Algorithm.

Let $A \in R^{n\times n}$, a random $X^0 \in R^{n\times p}$, $b \in R^n$, block size $p \in R$ and $tol \in R^+$;
$B = b\,\mathbf{1}^T$, $\mathbf{1} \in R^p$, $R^0 = B - AX^0$; set $Q_0 G_1^T = 0$
for ; **do** $\{j = 1, 2 \ldots\}$
  $M_j = Q_j^T A Q_j$;
  $W_j = AQ_j - Q_j M_j - Q_{j-1} G_j^T$;
  $Q_{j+1} G_{j+1} = W_j$;
  $Z = [Z \quad Q_{j+1}]$;
  Compute $T = Z^T AZ$ and solve $TY^j = Z^T B$
  Compute $X^j = ZY^j$ and $R^j = B - AX^j$
  if $\|R^j\| < \epsilon\|B\|$ **then**
    break;
  end if
  if $criterion\ met$ **then**
    Choose $\delta > 0$ and form an $n \times \delta$ matrix U such that
    $U^T Q_{k+1} = 0$, $U^T Q_i = 0$ and $U^T U = I_\delta$
    $p_{j+1} = p_j + \delta_j$
    $Q_{k+1} = [Q_{k+1}, U]$, $B_{j+1} = B_j + 1$
  else
    $\delta = 0$;
  end if
end for

A diffculty in the use of Alg. 1 consists in arriving at a suitable choice of threshold η in Eq. (8), and based on this, the determination of a rule for updating the block size s. Given a threshold η we assume that if the condition Eq. (8) is satisfied, then the parameter s must be increased. In such a case, the subspace is enlarged, by incorporating vectors that are orthogonal to each other and with the basis of the space before the enlargement. The resulting algorithm

is presented in Alg. 1, where the condition to be evaluated to decide the enlargement of the subspace is given by the expression Eq. (8) and in steps 13–18, the subspace is enlarged.

3 Numerical Experiments

In this section, we describe numerical results on tests performed on three matrix problems: Poisson equation using five point stencil finite difference, a matrix denoted FIDAP/ex13 which arose from the axisymmetric flow through a poppet valve simulation (Davis and Hu, 2011) and a matrix denoted K_2FF which comes from the two phase flow in porous media simulation.

The Poisson equation $\nabla^2\phi = f$ is discretized using the standard 5-point stencil operator, with mesh sizes 20×20, 35×35 and 50×50, given matrices of 400×400, 1225×1225 and 2500×2500, respectively. The FIDAP/ex13 matrix problem is described in Davis and Hu (2011) being a 2568×2568 matrix with a condition number $\kappa = 10^{12}$.

The matrix K_2FF is extracted from Varela et al. (2014) resulting from the two phase flow simulation of CO2 injection in a portion of aquifer. The porous media is considered homegenous, and the porosity as a constant; the mesh size considered is 100×100, resulting in a 10000×10.000 matrix with a condition number $\kappa = 10^4$. For all problems, the right hand sides are generated randomly.

The stopping criterion for the iterative solvers for the outer iteration is

$$\|r^k\|/\|r^0\| \leq tol, \tag{7}$$

where $\|r^k\|$ is the l_2 norm of the residual at the kth iteration of the BLM. In all cases, we consider as tolerance $tol = 10^{-9}$.

3.1 Experiment on threshold sensibility

To determine the effect of the threshold, several threshold values are tested with values $\eta_1 = 10^0$, $\eta_2 = 10^{-1}$, $\eta_3 = 10^{-2}$ and $\eta_4 = 10^{-3}$. Table 1 shows the resulting estimated cost for each problem with respect to the cost of classic CG. The iteration begin with a initial block size of $s = 4$ and is incremented by one ($s = s + 1$), once a given criterion is met until a maximum number of $s = 8$. Therefore if the following clusterization criterion is satisfied,

$$\frac{\theta_{i+k-1} - \theta_i}{min\{\theta_i - \theta_{i-1}, \theta_{i+k} - \theta_{i+k-1}\}} \leq \eta, \tag{8}$$

then the algorithm increments the value of s by one.

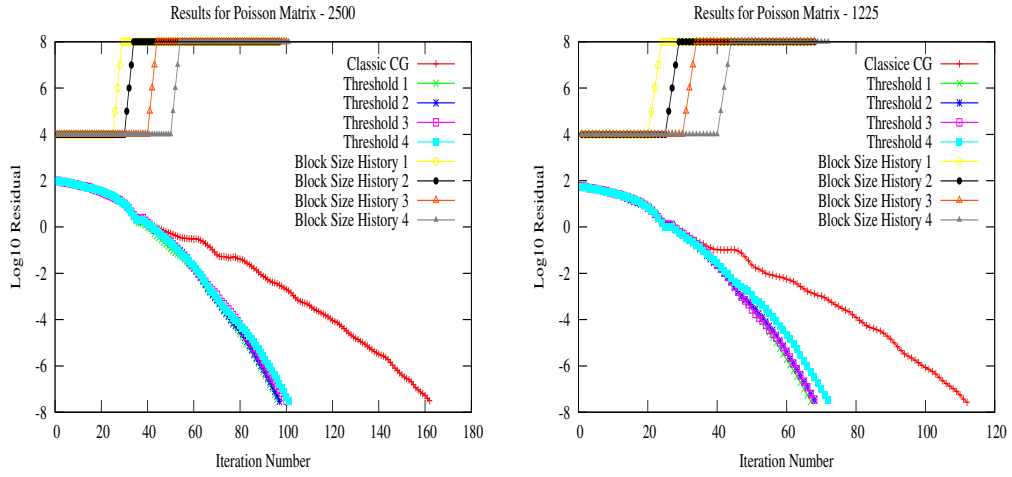
To compare the cost of the algorithm for each problem in terms of each iteration the following linear model presented in (Liu et al., 2012) is used:

$$C_T = \sum_{i=1}^{N}(1 + 0.0142 \times (s_i - 1)), \tag{9}$$

where C_T is total cost of the iteration process until the convergence, s_i is the block size at the iteration i and N is the number of iteration until convergence at the given tolerance. In the model (9), the cost of incrementing one vector is 1.42% with respect to one vector. Notice that the reduction of this percentage improves the performance, and it is machine and implementation dependent.

Table 1: Cost for several threshold η using expression (8) and relationship between the η cost and the CG cost

Matrix Size	$\eta_1 = 10^{-1}$	$\eta_2 = 10^{-2}$	$\eta_3 = 10^{-3}$	$\eta_4 = 10^{-3}$	Classic CG
Poisson 400	34.52	36.44	37.25	38.07	52
Poisson 1225	64.98	65.78	65.49	69.27	90
Poisson 2500	96.59	94.96	98.21	95.24	130
Poisson 5000	397.25	269.40	213.01	199.18	441
K_2FF	397.25	269.40	213.01	199.18	450
FIDAP/ex13	62.54	110.88	219.68	251.23	150



(a) (b)

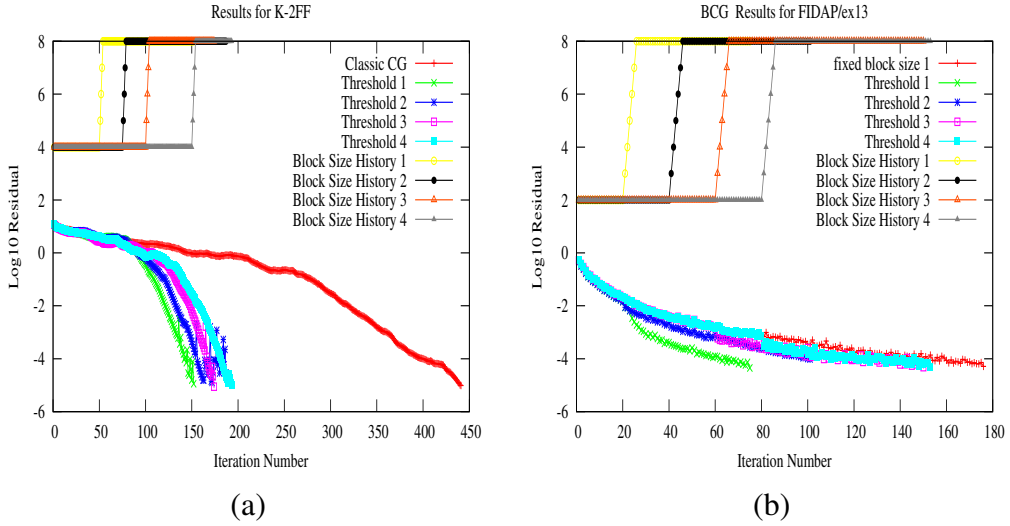
Figure 1: Results for Poisson Matrix (a) **Grid** 50 **and** (b) **Grid** 35**.**

In Table 1 the cost of the classic CG as well as the cost of the ABL using several threshold are presented. As observed for all Poisson matrix problems the cost of the ABL is smaller than the corresponding CG with ranging between 1.11 and 2.21 time faster. In addition for all cases, the total number of matrix-vector multiplication remains lower than the classic CG algorithm. In special for the *Poisson 5000* problem it is observed a significant improvement with respect to the classic CG when the threshold is reduced.

The importance of choosing the threshold η adequately is observed for the problems FIDAP/ex13 and K_2FF, where choosing adequately the value of the threshold η, in the best case scenario tested, the relative cost ranges between 2.26 and 2.4 faster the classic CG, with 0.59 and 1.13 otherwise

3.2 Experiment of convergence behavior

Figure 1 shows the residual convergence behavior for the Poisson matrix problems for each threshold tested η. In addition, the block size at each iteration is presented. Observe that in



(a) (b)

Figure 2: Results for (a) **K_2FF and** (b) **FIDAP/ex13.**

terms of the residual convergence behavior and number of iterations in all cases the ABL is better than the classic CG.

Figures 2 (a) and 2 (b) show the results for K_2FF and FIDAP/ex13 matrix problems, respectively. Notice that for the K_2FF matrix problem the residual convergence behavior for all η tested is improved. However, observe that the η that produces the better residual convergence behavior (η_1), produces a relative cost w.r.t. the classical CG moderately large ($\eta_1 = 10^{-1}$ with a relative cost of $450/397.25 = 1.13$), and choosing adequately η it is possible to improve the relative cost without significantly degrade the convergence residual behavior.

The above affirmation is corroborated in the FIDAP/ex13 matrix problem where for the η_1 it is produced the best residual convergence behavior, as well as the best relative cost $150/62.54 = 2.39$. For problem tested there is a clear advantage in using the BLM when the threshold is adequately chosen

4 Concluding Remarks

An adaptive Block Lanczos method - ABL - for solving linear systems has been presented and tested, as well as a criterion to vary the size of the block s. This criterion is based on an approximation of the eigenvalues of the matrix A using its corresponding Ritz values and a threshold η. It is important to notice that the residual convergence is improved for all cases tested independently of the choice of η in the range tested. Moreover, in order to also achieve an improvement in terms of computational cost of the ABL w.r.t. the classical CG it is important to choice adequately the threshold η. The results are quite encouraging since it is possible to obtain improvements in both the convergence behavior and the computational cost. The adequate selection of the threshold η is subject of ongoing research.

ACKNOWLEDGEMENTS

PT and CES acknowledge the financial support given by scholarship Proyecto BID 1698/OC-PR, PRONII and 14-INV-186 (CIMA) - CONACyT - Paraguay, respectively. AB acknowledges BPP grant from CNPq/Brazil that partially funds his reasearchs. PT acknowledges the J. Varela for providing the matrix K_2FF.

REFERENCES

Abu-Sufah, W. and Ahmad, K., 2014. Performance of sparse matrix-multiple vectors multiplication on multicore and gpus. In *International Conference for High Performance Computing, Networking, Storage and Analysis, Supercomputing 2014 (SC 14)*, pages 933–938.

Aktulga, H. M., Buluc, A., Williams, S., and Yang, C., 2014. Optimizing sparse matrix-multiple vectors multiplication for nuclear configuration interaction calculations. In *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*, pages 1213–1222. IEEE.

Concus, P., Golub, G. H., and O'Leary, D. P., 1975. Generalized conjugate gradient method for the numerical solution of elliptic partial differential equations. Technical report, California Univ., Berkeley (USA). Lawrence Berkeley Lab.

Davis, T. A. and Hu, Y., 2011. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, vol. 38, n. 1, pp. 1.

Feng, Y., Owen, D., and Perić, D., 1995. A block conjugate gradient method applied to linear systems with multiple right-hand sides. *Computer methods in applied mechanics and engineering*, vol. 127, n. 1, pp. 203–215.

Ghysels, P., Kłosiewicz, P., and Vanroose, W., 2012. Improving the arithmetic intensity of multigrid with the help of polynomial smoothers. *Numerical Linear Algebra with Applications*, vol. 19, n. 2, pp. 253–267.

Golub, G. H. and Van Loan, C. F., 2012. *Matrix computations*, vol. 3. JHU Press.

Gropp, W., Kaushik, D., Keyes, D., and Smith, B., 1999. Toward realistic performance bounds for implicit cfd codes. In *Proceedings of parallel CFD*, vol. 99, pages 233–240.

Gropp, W. D., Kaushik, D. K., Keyes, D. E., and Smith, B. F., 2001. High-performance parallel implicit cfd. *Parallel Computing*, vol. 27, n. 4, pp. 337–362.

Hestenes, M. R. and Stiefel, E., 1952. Methods of conjugate gradients for solving linear systems. NBS.

Hoemmen, M., 2010. *Communication-avoiding Krylov subspace methods*. PhD thesis, University of California, Berkeley.

Jianhua, Z. and Jing, Z., 2013. A novel class of block methods based on the block aa t-lanczos bi-orthogonalization process for matrix equations. *International Journal of Computer Mathematics*, vol. 90, n. 2, pp. 341–359.

Kislal, O., Ding, W., Kandemir, M., and Demirkiran, I., 2013. Optimizing sparse matrix vector multiplication on emerging multicores. In *Multi-/Many-core Computing Systems (MuCoCoS), 2013 IEEE 6th International Workshop on*, pages 1–10. IEEE.

Liu, X., Chow, E., Vaidyanathan, K., and Smelyanskiy, M., 2012. Improving the performance of dynamical simulations via multiple right-hand sides. *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium, IPDPS 2012*, vol. pp. 36–47.

Nikishin, A. and Yeremin, A., 1995. Variable block cg algorithms for solving large sparse symmetric positive definite linear systems on parallel computers, i: General iterative scheme. *SIAM Journal on Matrix Analysis and Applications*, vol. 16, n. 4, pp. 1135–1153.

O'Leary, D. P., 1980. The block conjugate gradient algorithm and related methods. *Linear algebra and its applications*, vol. 29 pp. 293–322.

Ortega, G., Vázquez, F., García, I., and Garzón, E. M., 2014. Fastspmm: An efficient library for sparse matrix matrix product on gpus. *The Computer Journal*, vol. 57, n. 7, pp. 968–979.

Saad, Y., 2003. *Iterative methods for sparse linear systems*. SIAM. Philadelphia. USA.

Torres, P., Schaerer, C., and Bhaya, A., 2015. Varying the block size on block conjugate gradient: Comparison of strategies. In *First Pan American Congress on Computational Mechanics PANACM 2015*, pages 902–911. CIMNE.

Van der Sluis, A. and van der Vorst, H. A., 1986. The rate of convergence of conjugate gradients. *Numerische Mathematik*, vol. 48, n. 5, pp. 543–560.

Varela, J., Schaerer, C., and Nogues, J. P., 2014. Two-phase flow including capillary pressure and buoyancy effects: A two-dimensional model to study the carbon sequestration process. In *Third Conference of Computational Interdisciplinary Sciences.*, pages 441–442. CCIS.

Ye, Q., 1996. An adaptive block Lanczos algorithm. *Numerical Algorithms*, vol. 12, n. 1, pp. 97–110.