Proceedings of the Seventh International Conference on
Parallel, Distributed, GPU and Cloud Computing for Engineering
Edited by: P. Iványi, F. Magoulès and B.H.V. Topping
Civil-Comp Conferences, Volume 4, Paper 5.1
Civil-Comp Press, Edinburgh, United Kingdom, 2023
doi: 10.4203/ccc.4.5.1

Multisplitting Methods for Singular Nonlinear Systems

J. Arnal

Department of Computer Science and Artificial Intelligence
University of Alicante, Spain

Abstract

Multisplittings of a matrix are used to generate parallel algorithms to approximate the solutions of singular nonlinear algebraic systems. A class of parallel algorithms based on the Newton method is defined where the Jacobian is singular. The parallel methods are implemented on shared-memory parallel platforms using OpenMP. An application to the Chandrasekhar H-equation is presented. An illustration and comparison of these methods with their sequential versions is given. The speed-up on shared-memory parallel computers is recorded, achieving significative values of speed-up.

Keywords: parallel computing, multisplitting, nonlinear system, singular system, Newton method, Chandrasekhar H-equation

1 Introduction

Let $F : \mathbb{R}^n \to \mathbb{R}^n$ be a nonlinear mapping. Assume that $x^\star \in \mathbb{R}^n$ is a solution of the system

$$F(x^\star) = 0. \tag{1}$$

If $F'(x^\star)$ is invertible and F' is Lipschitz continuous near $x^\star$, the sequence generated by the Newton method

$$x^{k+1} = x^k - F'(x^k)^{-1}F(x^k), \quad k = 0, 1, 2, \ldots \tag{2}$$

converges quadratically to $x^\star$ if the initial approximation, x_0, is sufficiently near to $x^\star$ [1]. If $F'(x^\star)$ is not invertible then the system (1) is singular and the solution $x^\star$ is called a singular root. In this case the Newton method does not converge quadratically to $x^\star$. Singular roots cause a number of problems in implementation of iterative methods and in general deteriorate the rate of convergence. The convergence in many such situations [2, 3] has been proven to be linear if x_0 is chosen near $x^\star$ and in a special kind of region not containing any ball around $x^\star$. In [4] a modification of the Newton method for singular systems is presented. This method is based in a two-step iteration and is locally superlinearly convergent.

On the other hand, Newton iterative methods [1, 5] use an iterative method to approximate the solution of the linear system

$$F'(x^k)z = F(x^k). \tag{3}$$

In order to generate efficient algorithms to solve nonlinear systems (1) on a parallel computer, different authors [6–10] used the multisplitting technique [11–16] to approximate the linear system (3).

In this study we present a class of parallel algorithms for the solution of singular nonlinear systems based on the method introduced in [4]. The method is structured in two main stages and at each stage a multisplitting algorithm is used. The computational performance analysis is evaluated using the metrics speedup and efficiency. The results showed the good performance achieved in terms of speedup and efficiency.

The paper is organized as follows. In section 2 the formulation of the parallel multisplitting algorithms is presented. Section 3 presents the results obtained for the Chandrasekhar H-equation on two shared-memory multiprocessors. Section 4 concludes the paper with some concluding remarks and future studies.

2 Methods

In this section we briefly overview the method introduced in [4] and present the formulation of the new parallel methods.

The modified Newton algorithm suggested in [4] consists of two steps:

Given $x^0 \in \mathbb{R}^n$, for $k = 0, 1, 2, \ldots$

- Step 1: $v^k = x^k + \omega^k$ where ω^k is the solution of the linear system

$$F'(x^k)\omega^k = -F(x^k).$$

- Step 2: $x^{k+1} = v^k + (2 - C \cdot \|s^k\|^\alpha)s^k$, where s^k is the solution of the linear system

$$F'(v^k)s^k = -F(v^k).$$

In order to approximate the solution of the linear systems by using parallel multisplitting methods, let us consider two multisplittings of $F'(x)$,

$$\{M_{1,k}(x), N_{1,k}(x), E_{1,k}\}_{k=1}^p, \tag{4}$$
$$\{M_{2,k}(x), N_{2,k}(x), E_{2,k}\}_{k=1}^p. \tag{5}$$

The parallel Newton algorithm consists of two main stages that can be structured in four steps:

Given $x^0 \in \mathbb{R}^n$. For $\ell = 0, 1, 2, \ldots$

- Step 1: Solve the linear system

$$F'(x^\ell)\omega^\ell = -F(x^\ell) \tag{6}$$

in parallel using multisplitting (4), i.e., compute $\omega^{m_{1,\ell}}$ performing $m_{1,\ell}$ iterations of the parallel iterative method determined by the multisplitting (4):

$$\omega^i = H_{1,\ell}(x^\ell)\omega^{(i-1)} + B_{1,\ell}(x^\ell)F(x^\ell), \quad i = 1, 2, \ldots, m_{1,\ell},$$

with

$$\begin{aligned} H_{1,\ell}(x) &= \sum_{k=1}^p E_{1,k}\left(M_{1,k}^{-1}(x)N_{1,k}(x)\right), \\ B_{1,\ell}(x) &= \sum_{k=1}^p E_{1,k}\left(M_{1,k}^{-1}(x)N_{1,k}(x)\right)M_{1,k}^{-1}(x), \end{aligned}$$

and $\omega^0 = 0$. Thus

$$\omega^{m_{1,\ell}} = \sum_{i=0}^{m_{1,\ell}-1} H_{1,\ell}^i(x^\ell)B_{1,\ell}(x^\ell)F(x^\ell).$$

- Step 2: Compute $v^\ell = x^\ell + \omega^{m_{1,\ell}}$

- Step 3: Solve the linear system

$$F'(v^\ell)s^\ell = -F(v^\ell) \tag{7}$$

in parallel using multisplitting (5), i.e., compute $\omega^{m_{2,\ell}}$ performing $m_{2,\ell}$ iterations of the parallel iterative method determined by the multisplitting (5):

$$s^i = H_{2,\ell}(v^\ell)s^{(i-1)} + B_{1,\ell}(v^\ell)F(v^\ell), \quad i = 1, 2, \ldots, m_{2,\ell},$$

with

$$\begin{aligned} H_{2,\ell}(x) &= \sum_{k=1}^{p} E_{2,k}\left(M_{2,k}^{-1}(x)N_{2,k}(x)\right), \\ B_{2,\ell}(x) &= \sum_{k=1}^{p} E_{2,k}\left(M_{2,k}^{-1}(x)N_{2,k}(x)\right) M_{2,k}^{-1}(x), \end{aligned}$$

and $s^0 = 0$. Thus

$$s^{m_{2,\ell}} = \sum_{i=0}^{m_{2,\ell}-1} H_{2,\ell}^i(v^\ell)B_{2,\ell}(v^\ell)F(v^\ell).$$

- Step 4: Compute

$$x^{\ell+1} = v^\ell + (2 - C \cdot \|s^{m_{2,\ell}}\|^\alpha)s^{m_{2,\ell}}. \tag{8}$$

3 Experiments

In this section we describe the experiments conducted. We consider the Chandrasekhar H-equation [17] given by

$$\mathcal{F}(H)(\mu) = H(\mu) - \left(1 - \frac{\omega}{2}\int_0^1 \frac{\mu H(\nu)\, d\nu}{\mu + \nu}\right)^{-1} = 0. \tag{9}$$

This equation is used to solve exit distribution problems in radiative transfer. $\mathcal{F}$ is a map on $C[0,1]$, the Banach space of continuous functions on the interval $[0,1]$ with the $\|\cdot\|-$ norm. The unknown is a function $H \in C[0,1]$; and $\omega \in [0,1]$ is a parameter. For $\omega = 1$ this equation has a unique solution and there is a simple fold singularity [18]. We discretized the equation with the composite midpoint rule. Integrals on $[0,1]$ are approximated by

$$\int_0^1 f(\mu)\, d\mu \approx \frac{1}{N}\sum_{j=1}^{N} f(\mu_j), \tag{10}$$

where $\mu_i = (i - 1/2)/N$ for $i = 1, 2, \ldots, N$. The resulting discrete problem is

$$F(x)_i = x_i - \left(1 - \frac{\omega}{2N}\sum_{j=1}^{N} \frac{\mu_i x_j}{\mu_i + \mu_j}\right)^{-1}, \quad i = 1, 2, \ldots, N. \tag{11}$$

In the experiments we considered the case $\omega = 1$ where the Jacobian is singular at the solution. The stopping criterion used was $\|F(x^\ell)\| < 10^{-9}$ and we have used the L^2- norm to compute $\|F(x)\|$. We have considered the initial iterate x_0 with all components equal to one. In the experiments we considered the multisplittings given by $M_{1,k}(x) = M_{2,k}(x) = \mathrm{Diag}(F(x))$ which determine the Jacobi iterative method. The linear systems (6) and (7) were solved performing $m_{1,\ell} = m_{2,\ell} = \ell$ iterations. The results presented in this study were obtained with $\alpha = 0.3$ in (8).

We have coded the parallel implementation of the algorithm on two shared-memory machines using the Open Multi-Processing (OpenMP) [19]. Both the serial code and parallel code were implemented in Fortran. The GNU Fortran (GCC) 8.5.0 was used. We developed experiments on two multi-cores:

- Multi-core 1: A multi-core Intel Xeon CPU X5660 (12 cores), 2.8 GHz, with 48 GB RAM, under the operative system CentOS Linux version 7.
- Multi-core 2: A multi-core Intel Xeon CPU W-3245 (16 cores), 3.2 GHz, with 254 GB RAM, under the operative system CentOS Linux version 8.

To measure the parallel performance, the speed-up S_P is computed as:

$$S_P = \frac{T_{seq}}{T_P}, \tag{12}$$

where T_{seq} is the computation time of the sequential method and T_P is the computation time of the parallel algorithm. Table 1 shows the computational time for the systems of size $N = 1024$ and $N = 4096$ on Multi-core 1, and Table 2 presents the corresponding results on Multi-core 2. In all cases, terminating the iteration when $\|F(x^\ell)\|$ fell below 10^{-9}, all the iterations in Tables 1 and 2 stopped at the same iteration, 419 for $N = 1024$ and 499 for $N = 4096$.

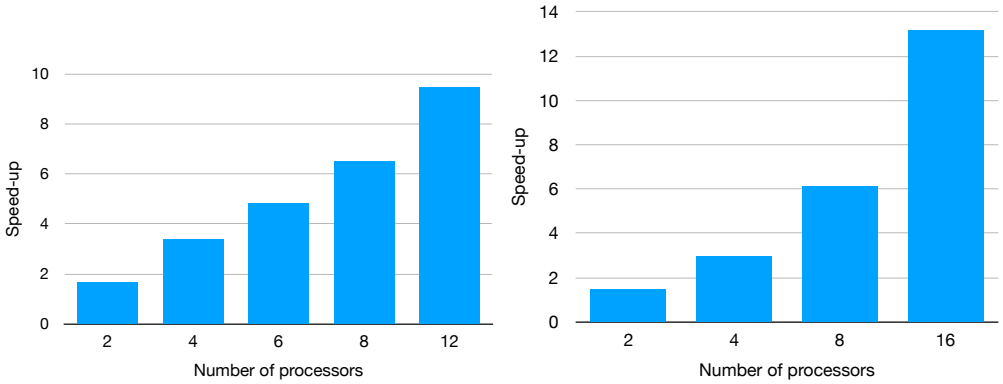
Number of processors	1	2	4	6	8	12
N = 1024	1943.9	1158.1	572.5	401.7	297.8	205.5
N = 4096	169622.7	108731.2	54565.5	36774.2	27790.1	18997.2

Table 1: Computational time in seconds on Multi-core 1.

Figures 1, 2, 3 and 4 present the speed-up and efficiency for the systems of size $N = 1024$ and $N = 4096$. Results show that the parallel method achieves speed-ups in the range 8.93 to 9.46 (efficiencies in the range 74.40% to 78.81%) when the 12 cores of the Multi-core 1 are used, and in the range 13.17 to 13.25 (efficiencies in the range 82.30% to 82.84%) when the 16 cores of Multi-core 2 are used.

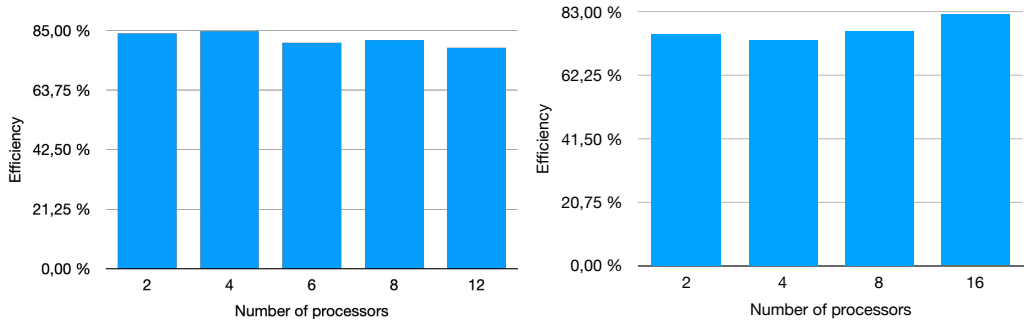
Number of processors	1	2	4	8	16
N = 1024	688.3	454.3	232.4	112.1	52.2
N = 4096	148514.7	92973.7	46435.5	22787.4	11204.9

Table 2: Computational time in seconds on Multi-core 2.



(a) Multi-core 1 (b) Multi-core 2

Figure 1: Speed-up on multi-cores; N = 1024.

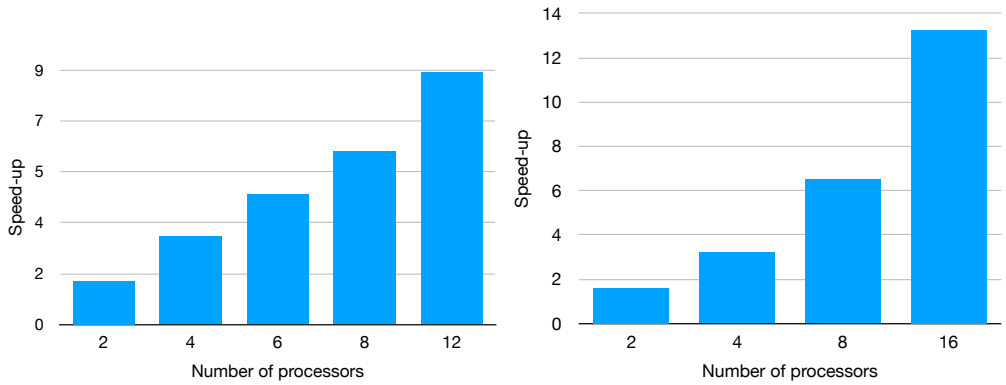


(a) Multi-core 1 (b) Multi-core 2

Figure 2: Efficiency on multi-cores; N = 1024.

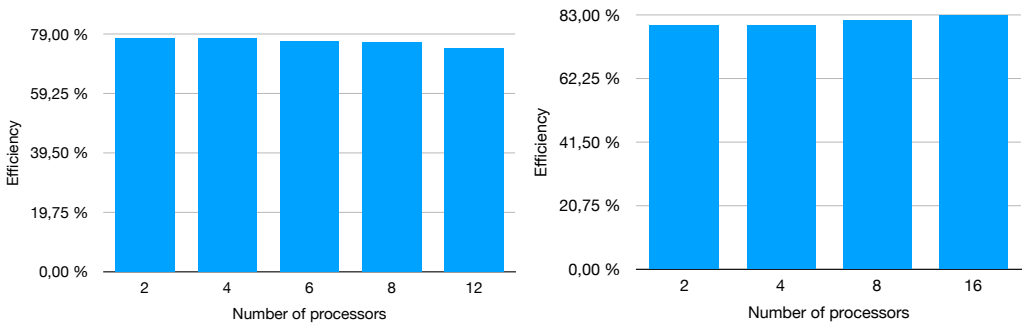
4 Conclusions

Parallel algorithms to solve singular nonlinear systems have been presented. These algorithms are a modification of Newton method structured in two main stages. At each stage a parallel multisplitting algorithm is used. The method has been implemented on shared-memory multiprocessors using OpenMP. The implementation has



(a) Multi-core 1 (b) Multi-core 2

Figure 3: Speed-up on multi-cores; N = 4096.



(a) Multi-core 1 (b) Multi-core 2

Figure 4: Efficiency on multi-cores; N = 4096.

been used to solve the Chandrasekhar H-equation fixing the parameter that determines a singular Jacobian. The experimental results show that the proposed parallel method obtains significant speed-up values on two multi-cores where the implementation has been tested. In future works we will analyze the convergence of these methods by contraction techniques. Moreover, different multisplittings will be considered and asynchronous iterations will be analyzed. We will also implement the proposed method on distributed memory machines using MPI.

Acknowledgements

This research was supported by the Spanish Ministry of Science and Innovation (Grant PID2021-123627OB-C55) co-financed by FEDER funds.

References

[1] J. Ortega, W. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 2000.

[2] A. Griewank, M.R. Osborne, "Newton's Method for Singular Problems when the Dimension of the Null Space is > 1", *SIAM Journal on Numerical Analysis*, 18(1): 145–149, 1981.

[3] G.W. Reddien, "On Newton's method for singular problems", *SIAM Journal on Numerical Analysis*, 15(5): 993–996, 1978.

[4] C.T. Kelley, R. Suresh, "A new acceleration method for Newton's method at singular points", *SIAM journal on numerical analysis*, 20(5): 1001–1009, 1983.

[5] A.H. Sherman, "On Newton-iterative methods for the solution of systems of nonlinear equations", *SIAM Journal on Numerical Analysis*, 15(4): 755–771, 1978.

[6] R.E. White, "Parallel algorithms for nonlinear problems", *SIAM Journal on Algebraic Discrete Methods*, 7(1): 137–149, 1986.

[7] A. Frommer, "Parallel nonlinear multisplitting methods", *Numerische Mathematik*, 56: 269–282, 1989.

[8] J. Arnal, V. Migallón, J. Penadés, "Parallel Newton two-stage multisplitting iterative methods for nonlinear systems", *BIT Numerical Mathematics*, 43(5): 849–861, 2003.

[9] J. Arnal, V. Migallón, J. Penadés, "Non-stationary parallel multisplitting algorithms for almost linear systems", *Numerical linear algebra with applications*, 6(2): 79–92, 1999.

[10] T. Garcia, P. Spiteri, L. Ziane-Khodja, R. Couturier, "Solution of univalued and multivalued pseudo-linear problems using parallel asynchronous multisplitting methods combined with Krylov methods", *Advances in Engineering Software*, 153: 102929, 2021.

[11] P. Spiteri, "Parallel asynchronous algorithms: A survey", *Advances in Engineering Software*, 149: 102896, 2020.

[12] D.P. O'Leary, R.E. White, "Multi-Splittings of Matrices and Parallel Solution of Linear Systems", *SIAM Journal on Algebraic Discrete Methods*, 6(4): 630–640, 1985.

[13] G. Gbikpi-Benissan, F. Magoulès, "Asynchronous multisplitting-based primal Schur method", *Journal of Computational and Applied Mathematics*, 425: 115060, 2023.

[14] V. Partimbene, T. Garcia, P. Spiteri, P. Marthon, L. Ratsifandrihana, "Asynchronous multi-splitting method for linear and pseudo-linear problems", *Advances in Engineering Software*, 133: 76–95, 2019.

[15] D.B. Szyld, M.T. Jones, "Two-stage and multisplitting methods for the parallel solution of linear systems", *SIAM Journal on Matrix Analysis and Applications*, 13(2): 671–679, 1992.

[16] M.T. Jones, D.B. Szyld, "Two-stage multisplitting methods with overlapping

blocks", *Numerical linear algebra with applications*, 3(2): 113–124, 1996.
[17] S. Chandrasekhar, *Radiative transfer*, Courier Corporation, 2013.
[18] H.B. Keller, *Lectures on Numerical Methods in Bifurcation Problems*, Springer, Berlin, 1987.
[19] L. Dagum, R. Menon, "OpenMP: An industry-standard API for shared-memory programming", *Computing in Science & Engineering*, 5(1): 46–55, 1998.