GMRES Based Numerical Simulation of Multicomponent Multiphase Flow in Porous Media on LuNA Fragmented Programming System

N. Kassymbek[1*], B. Matkerim[1], D. Lebedev[2], T. Imankulov[1], D. Akhmed-Zaki[2]

[1] Al-Farabi Kazakh National University; [2] University of International Business

Summary

Using supercomputers can significantly accelerate the solution of problems when using numerical methods. One of these tasks is forecasting oil and gas production at specific oil and gas fields. Modeling of multicomponent multiphase fluid (oil and gas) flow in porous media (in oil reservoirs) is relevant and at the same time complex problem of hydrodynamic simulation. To solve such problems, various methods and schemes are used, some of whom are iterative methods for solving linear systems.

In this paper, we solve the equation system with the Newton-Raphson method, within each iteration of which the algebraic equation system is solved by the generalized minimal residual method (GMRES) with the ILU(0) preconditioner. The application of this problem which is named Newton-ILU(0)-GMRES method is implemented in parallel with MPI and Fragmented Programming (FP) Technology, which is aimed at automation of implementation of numerical applications for multicomputers. As in other numerical methods, in our case is a non-degenerate matrix. Storage of a full matrix with all zero elements leads to huge memory costs. In order to optimize the memory, the format of storage of sparse matrices was chosen – compressed row storage (CSR).

Tested the developed MPI parallel application and fragmented program on the MVS-10P supercomputer of the Interdepartmental Supercomputer Center of the Russian Academy of Sciences. The runtime between the MPI parallel program and the fragmented program in the LuNA system is compared and the results were analyzed.

The work was supported by Funding Science committee of Ministry of Education and Science of Kazakhstan, grant no. AP05134651.

Introduction

The development of modern high-performance computing allows humanity to solve important scientific and applied problems in various fields. The accuracy of solving such problems requires the use of a large mesh, which in turn requires large computational resources. Currently, supercomputers are used to solve problems such as weather modeling and climate prediction, acoustic problems, hydrodynamics, economic processes, biological research, and the production of medicines. Technological advances in such areas are often related to how correctly and successfully computer computing is used.

When using numerical methods to solve specific problems, one can achieve significant acceleration using supercomputers. One and relevant of these tasks is the forecast of oil and gas production at specific oil and gas fields. Actual and at the same time quite a difficult task of hydrodynamic modeling is the simulation of multicomponent multiphase flow of liquid (oil and gas) in porous media (in oil reservoirs). In solving such problems, various methods and schemes (Borisov 2013; Chen 2006; Chen 2007; Ahmed 2006; Akhmed-Zaki *et al.* 2016; Imankulov *et al.* 2016) can be used, some of them are iterative methods for solving linear systems (Mittal and Al-Kurdi 2002; Lacroix *et al.* 2003; Wang *et al.* 2013; Vabishchevich and Vasilyeva 2011).

Various technologies are used to speed up the execution time of calculations for solving scientific problems, such as MPI (Pannala *et al.* 2003; Akhmed-Zaki *et al.* 2016), OpenMP (Iryanto and Gunawan 2017; Werneck *et al.* 2016), CUDA (Zaza *et al.* 2016; McClure *et al.* 2014; Akhmed-Zaki *et al.* 2017), OpenCL (Khramchenkov and Khramchenkov 2018), and fragmented programming (Akhmed-Zaki *et al.* 2016; Malyshkin and Perepelkin 2011). The LuNA system, which implements fragmented programming technology, uses a computational model called the fragmented program (FP). In this model, these tasks are presented as a set of separate units called data fragments (DF).

In this paper, we solve a system of equations using the Newton-Raphson method, at each iteration of which the system of algebraic equations is solved using the generalized minimum residual method (GMRES) with ILU(0) preconditioner. A program for solving this problem using a method called Newton-ILU(0)-GMRES is implemented using fragmented programming (FP) technology, which is aimed at automating the construction of parallel programs for supercomputers.

Method

One of the most important problems of sparse matrix algebra is the solution of a system of linear algebraic equations (SLAE) of the form

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

where A is a sparse n × n matrix, b is a vector consisting of n elements, and x is a vector of unknowns of n elements. Such problems arise when solving many mathematical physics problems (Tihonov and Samarskij 1977). The solution for a system of this type is the vector x, which when placed in the system gives the correct equation.

Methods for solving systems of algebraic equations are divided into direct (Gaussian elimination) and iterative (CG, BiCG, BiCGStab). Both methods have their drawbacks. So, for example, in the case of direct methods, decomposition can cause a problem with the saturation of the system matrix, which, in turn, leads to undesirable memory filling. Iterative methods do not fill the memory, but their convergence time can be slow. In practice, the choice of method often depends on the problem being solved. (Balandin and Shurina 2000). To solve our system, an iterative method of the Krylov type GMRES was chosen.

The generalized minimum residual method is an iterative method for finding an approximate solution of a system of linear algebraic equations of the form (1). The method was developed by Youcef Saad and Martin H. Schultz in 1986 (Saad and Schults 1986).

Consider the Krylov subspace K_m:

$$K_m = K_m(A,b) = span\{b, Ab, A^2b, \dots, A^{m-1}b\}$$

We can represent any vector x from the subspace $x_0 + K_m$ in the form

$$x_m = x_0 + V_m y,$$

where V_m is the matrix composed of vectors of the orthonormal basis of the Krylov subspace K_m, and y is a vector of dimension m. In this case, the residual of the equation can be defined as a function of the vector y

$$J(y) = \|b - Ax\| = \|b - A(x_0 + V_m y)\| \qquad (2)$$

Due to the fact that the columns of the matrix V_{m+1} constitute an orthonormalized system of vectors, the following relation holds:

$$J(y) = \|b - A(x_0 + V_m y)\| = \|\beta e_1 - \bar{H}_m y\|$$

The method of generalized minimal residuals approximates the exact solution of system (1) by the vector x_m taken from the subspace $x_0 + K_m$, which serves to minimize the residual (2). Vector x_m can be obtained as

$$x_m = x_0 + V_m y_m, \qquad (3)$$

The vector y_m can be found as a solution to the linear least-squares problem of size $(m+1) \times m$, where $m \ll n$

$$y_m = \underset{y}{\arg min} \|\beta e_1 - \bar{H}_m y\|.$$

To reduce the number of iterations of the method, you can use preconditioning - a modification of the system of linear equations, which allows you to accelerate its solution.

This article discusses preconditioning by multiplying the original system by some matrix M^{-1} on both sides, that is, switching to a system of the form:

$$M^{-1}Ax = M^{-1}b \qquad (2)$$

where M is our preconditioner.

There are different types of preconditioners, such as ILU(0), ILQ, ILU(p). In this work, the ILU (0) preconditioner was chosen, which is easily calculated using the LU decomposition algorithm with some modifications.

The operation of the GMRES algorithm with the ILU(0) preconditioner, which consists of several steps, such as Arnoldi's orthogonalization, Givens rotation, can be seen in the activity diagram in Figure 1.

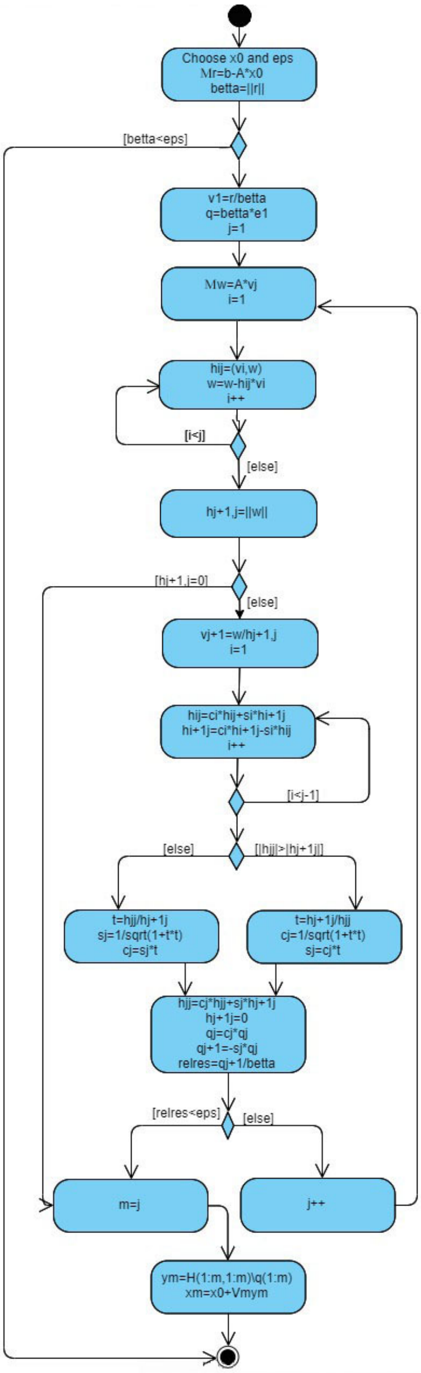

Figure 1 *Activity diagram of ILU(0)-GMRES algorithm*

Thus, to solve a system of the form (1), the ILU(0)-GMRES algorithm was used.

As in other numerical methods, in our case is a non-degenerate matrix. Storage of a full matrix with all zero elements leads to huge memory costs. In order to optimize the memory, the format of storage of sparse matrices was chosen – Compressed Sparse Rows (CSR) (Pissanetzky 1984).

In the selected format, matrix A is stored as three one-dimensional arrays:

1. All nonzero elements a_{ij} are line-wise, from the first to the last line, written to a one-dimensional array of "values".

2. The second index (column number) of each nonzero element is written to the one-dimensional array “cols”.
3. The location of a nonzero element in each row is written to the one-dimensional array "rowindex". Elements of an i-row are stored in matrix A at positions from rowindex[i] to rowindex[i+1]-1. If row i contains only null elements (row is empty), then rowindex[i]=rowindex[i+1]. For matrix A, which consists of n rows, the length of the rowindex array will be (n + 1).

Parallel and Fragmented algorithm

As can be seen in the previous chapter, ILU(0)-GMRES algorithm has different vector operations that can be implemented in parallel. The preconditioner matrix M is also calculated in parallel using the fine-grained LU decomposition algorithm. Thus, the parallel version of the ILU(0)-GMRES algorithm was implemented using the MPI standard.

To solve this problem, a fragmented version of the ILU(0)-GMRES algorithm for the LuNA system was developed. A fragmented algorithm frees the programmer from parts of parallel program execution, such as data transfer and messaging, and allows him to perform these actions automatically. In the system of fragmented programming LuNA to automate the creation of parallel programs uses the so-called fragmented algorithm (FA), which consists of light processes, computational fragments (CF). The data is represented as an immutable block called data fragments (DF). Accordingly, new data, called output DFs, are computed from input DFs using CF.

In the fragmented version of the Arnoldi orthogonalization algorithm, the vectors w and v_k are divided by n the number of DFs. The resulting matrices $H_{m,m}$ and $Q_{n,k}$ are stored as separate DFs. In each new Arnoldi iteration, the defined CFs calculate the new output DFs, using the above listed DFs as input parameters. In this case, the input DF, which will no longer be used, can be removed from memory.

You can see that calculating of the residual r and finding the approximate solution x^m of the GMRES algorithm, which was described before, are simple matrix-vector operations, the fragmented algorithms of which can be easily implemented in the LuNA system.

In LuNA system matrix A is considered as one DF. The array "values" is stored in DF A[0]. The "cols" and "rowindex" arrays are stored in DF A[0] and DF A[1], respectively.

Now consider the decomposition of the matrix A in LuNA. As seen in Figure 2, DF A[0] is divided into n parts of DF Apart[0][i], where n is equal to the number of fragments. DF A [1] and DF A[2], respectively, are also divided into DF Apart[1][i] and DF Apart[2][i].

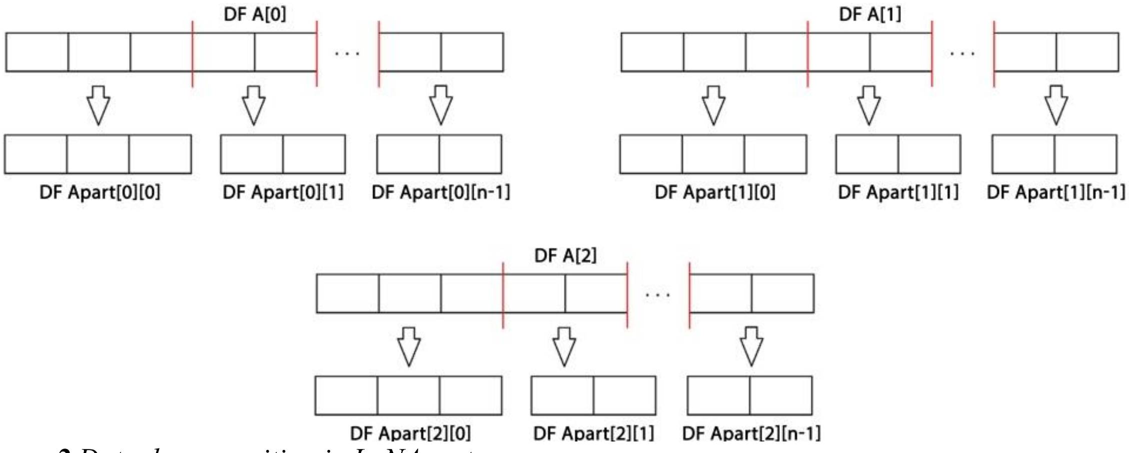

***Figure 2** Data decomposition in LuNA system*

Numerical experiments

To conduct a numerical experiment, a mathematical model of a multicomponent three-phase fluid flow in a porous medium is considered. We write the law of conservation of mass for each component (Chen 2006; Chen 2007):

$$\frac{\partial(\phi\ \xi_\omega S_\omega)}{\partial t} + \nabla \cdot (\xi_\omega u_\omega) = q_\omega,$$

$$\frac{\partial(\phi[x_{mo}\xi_o S_o + x_{mg}\xi_g S_g])}{\partial t} + \nabla \cdot \left(x_{mo}\xi_o u_o + x_{mg}\xi_g u_g\right) = q_m \tag{3}$$

$$u_\alpha = -\frac{k_{r\alpha}}{\mu_\alpha} k(\nabla p_\alpha - \rho_\alpha \wp \nabla z), \ \ \alpha = \omega, o, g. \tag{4}$$

$$\sum_{m=1}^{N_c} \quad x_{mo} = 1, \qquad \sum_{m=1}^{N_c} \quad x_{mg} = 1. \tag{5}$$

$$S_\omega + S_o + S_g = 1. \tag{6}$$

$$p_{co\omega} = p_o - p_\omega\ , p_{cgo} = p_g - p_o. \tag{7}$$

$$f_{mo}\left(p_o, x_{1o}, x_{2o,} \dots, x_{N_c o}\right) = f_{mg}\left(p_g, x_{1g}, x_{2g}, \dots, x_{N_c g}\right), \tag{8}$$

$$f_{m\alpha} = p_\alpha x_{m\alpha}\varphi_{m\alpha}, m = 1,2, \dots, N_c, \alpha = o, g.$$

m=1.2,...,N_c, where ξ_w – the molar density of the aqueous phase, q_w, q_m – the molar velocities of the water flow and *m*-th component respectively, S_w, S_o, S_g – saturation of the water, oil and gas phase, f_{mo} and f_{mg} fugacity functions.

The distribution of chemical components in the hydrocarbon phase was described by the K-value method (Pederson and Christensen 2008). The thermodynamic behavior of fluids in reservoir conditions was described by the Peng-Robinson equations of state (Peng and Robinson 1976).

The system of nonlinear equations (3) - (8) was linearized by the Newton-Raphson method.

In the (*l*+1) -th layer of the Newton iterative process, the value of the unknowns is updated by the following law (Chen 2007):

$$\begin{aligned}
x_{mo}^{n+1,l+1} &= x_{mo}^{n+1,l} + \triangle\, x_{mo}^{n+1,l+1}, m = 1,2, \dots, N_c - 1,\\
z_{mo}^{n+1,l+1} &= z_{mo}^{n+1,l} + \triangle\, z_{mo}^{n+1,l+1},\ \ m = 1,2, \dots, N_c - 1,\\
L^{n+1,l+1} &= L^{n+1,l} + \triangle\, L^{n+1,l+1},\\
F^{n+1,l+1} &= F^{n+1,l} + \triangle\, F^{n+1,l+1},\\
S^{n+1,l+1} &= S^{n+1,l} + \triangle\, S^{n+1,l+1},\\
p^{n+1,l+1} &= p^{n+1,l} + \triangle\, p^{n+1,l+1}.
\end{aligned} \tag{9}$$

The resulting system of linear equations (9) is reduced to the form (1).

As a result, we parallelized and used an algorithm called Newton-ILU(0)-GMRES, the activity diagram of which can be seen in Figure 3.

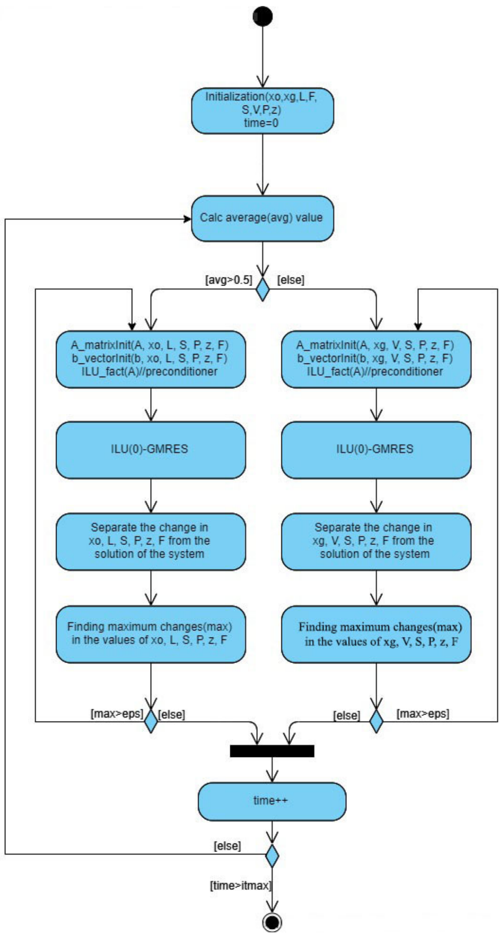
Initialization(xo,xg,L,F,
S,V,P,z)
time=0
Calc average(avg) value
[avg>0.5]
[else]
A_matrixInit(A, xo, L, S, P, z, F)
b_vectorInit(b, xo, L, S, P, z, F)
ILU_fact(A)//preconditioner
A_matrixInit(A, xg, V, S, P, z, F)
b_vectorInit(b, xg, V, S, P, z, F)
ILU_fact(A)//preconditioner
ILU(0)-GMRES
ILU(0)-GMRES
Separate the change in
xo, L, S, P, z, F from the
solution of the system
Separate the change in
xg, V, S, P, z, F from the
solution of the system
Finding maximum changes(max)
in the values of xo, L, S, P, z, F
Finding maximum changes(max)
in the values of xg, V, S, P, z, F
[max>eps]
[else]
[else]
[max>eps]
time++
[else]
[time>itmax]

Figure 3 *Activity diagram of Newton-ILU(0)-GMRES algorithm*

Experimental results

To assess the effectiveness of the parallel version of the GMRES method, a parallel program was implemented using the MPI standard. Test runs were made on the MVS-10P supercomputer of the Interdepartmental Supercomputer Center of the Russian Academy of Sciences, which includes nodes with two Xeon X5450 processors and 8 GB of RAM for each node. Testing of the parallel program was carried out on matrices corresponding to the cases when the number of nodes of the difference scheme varied in the range from 500 to 8000. The parameters of the matrix are shown in the table below.

Matrices	#1	#2	#3	#4	#5
Number of nodes	500	1000	2000	4000	8000
Size of matrix A	3000x3000	6000x6000	12000x12000	24000x24000	48000x48000

Table 1 *Matrix Description*

The test results are shown in the following table.

Matrix		#1	#2	#3	#4	#5
Sequantional Programm	**np=1**	3,74	48,38	334,24	1 688,74	20 219,34
MPI	**np=8**	0,42	7,57	53,82	394,74	2 942,02
	np=64	2,59	12,03	64,16	389,68	2 565,59
	np=128	2,95	12,66	65,25	395,58	2 569,01
	np=256	4,39	17,29	82,75	450,28	2 798,32
	np=512	11,53	42,56	202,01	866,98	4 185,07

Table 2 *Experiment results in seconds*

As we can see from the test results, the parallel program on 8 processes obtained the shortest execution time. As the size of the task increases, the parallel application running on 64 processes has the shortest calculation time, but it is not much different from the execution time of a parallel program running on 8, 128 and 256 processes. This is because with an increase in the number of processes, the counting time on each processor decreases, but the number of communications increases. This situation can be seen from the running time of a parallel program running on 512 processes, which even though it counts faster than a sequential program on a small task size (1000x1000) of approximately the same order, but on a larger scale (8000x8000) around by order of magnitude. We can conclude that the parallel algorithm of the method makes sense to run on a large number of processors, only on large sizes, such that they cannot fit into the memory of the node.

Conclusion

We have examined the parallel and fragmented implementation of the Newton-ILU0-GMRES algorithm for a multiphase, multicomponent oil filtration problem. Tests were carried out for different sizes of node and number of processes. The study showed that the MPI program has higher efficiency and scalability than the LuNA program. In the future we plan to develop a fragmented program to direct control for the considered parallel algorithm, which will increase the effectiveness of the fragmented program.

Acknowledgements

This research was funded by the Science Committee of the Ministry of Education and Science of the Republic of Kazakhstan (Grant No. AP05134651).

References

Ahmed, T. [2006]. *Reservoir Engineering Handbook. Gulf Professional Publishing*, 3rd edition, 1376.
Akhmed-Zaki, D.Zh., Daribayev, B.S., Imankulov, T.S. and Turar, O.N. [2017] High-performance computing of oil recovery problem on a mobile platform using CUDA technology. *Eurasian Journal of mathematical and computer applications*, 5(2), 4-13.

Akhmed-Zaki, D.Zh., Imankulov, T.S., Matkerim, B., Daribayev, B.S., Aidarov, K.A. and Turar, O.N. [2016]. Large-scale simulation of oil recovery by surfactant-polymer flooding. *Eurasian Journal of mathematical and computer applications*, 4(1), 12-31.

Akhmed-Zaki, D.Zh., Lebedev, D.V. and Perepelkin, V.A. [2016]. Implementation of a three dimensional three-phase fluid flow ("oil–water–gas") numerical model in LuNA fragmented programming system. *The Journal of Supercomputing*, 73(2), 624–630.

Balandin, M.YU. and Shurina, E.P. [2000]. *Metody resheniya SLAU bol'shoj razmernosti*. Novosibirsk: NGTU.

Borisov, V. E., Kritskiy, B. V., Marchenko, N. A., Mitrushkin, D. A. and Savenkov, E. B. [2013]. Non-isothermal compositional flow model with chemical reactions and active solid phase for reservoir simulation. *KIAM Preprint*, 91.

Chen, Z. [2006]. *Reservoir Simulation: mathematical techniques in oil recovery*. Society for Industrial and Applied Mathematics.

Chen, Z. [2007]. *Computational methods for multiphase flows in porous media*. Society for Industrial and Applied Mathematics.

Imankulov, T.S., Akhmed-Zaki, D.Zh., Daribayev, B.S. and Turar, O.N. [2016]. HPC Mobile Platform for Solving Oil Recovery Problem. *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016)*, 595-598.

Iryanto, I. and Gunawan, P. H. [2017]. An OpenMP parallel godunov scheme for 1D two phase oil displacement problem. *In Information and Communication Technology (ICoICT)*, 1-5.

Khramchenkov, E. and Khramchenkov, M. [2018]. Numerical Model of Two-Phase Flow in Dissolvable Porous Media and Simulation of Reservoir Acidizing. *Natural Resources Research*, 27(4), 531–537.

Lacroix, S., Vassilevski, YU., Wheeler, J. and Wheeler, M. [2003]. Iterative Solution Methods for Modeling Multiphase Flow in Porous Media Fully Implicitly. *SIAM J. SCI.COMPUT,* 25(3), 905–926.

Malyshkin, V.E. and Perepelkin, V.A. [2011]. LuNA Fragmented Programming System, Main Functions and Peculiarities of Run-Time Subsystem. *Proceedings of the 11th International Conference on Parallel Computing Technologies*, 53-61.

McClure, J. E., Prins, J. F. and Miller, C. T. [2014]. A novel heterogeneous algorithm to simulate multiphase flow in porous media on multicore CPU–GPU systems. *Computer Physics Communications*, 185(7), 1865–1874.

Mittal, R.C. and Al-Kurdi, A.H. [2002]. LU-decomposition and numerical structure for solving large sparse nonsymmetric linear systems. *Computers & Mathematics with Applications*, 185(2), 391-403.

Pannala, S., D'Azevedo, E., Syamlal, M. and O'Brien, T. [2003]. Hybrid (OpenMP and MPI) Parallelization of MFIX: A Multiphase CFD Code for Modeling Fluidized Beds.Conference. *Proceedings of the 2003 ACM Symposium on Applied Computing*, 199-206.

Pederson, K.S. and Christensen, P.L. [2008]. *Phase Behavior of Petroleum Reservoir Fluids*. CRC Press, 422.

Peng, D.Y. and Robinson, D.B. [1976]. A new two-constant equation of state. *Industrial and Engineering Chemistry Fundamentals*, 15, 59–64.

Pissanetzky, S. [1984]. *Sparse Matrix Technology*. Academic Press, London, 1984.

Saad, Y., and Schults, M.H. [1986]. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing,* 7(3), 856-869.

Tihonov, A.N. and Samarskij, A.A. [1977]. *Uravneniya matematicheskoj fiziki*. Moscow, Nauka, 1977.

Vabishchevich, P. and Vasilyeva, M. [2011]. Iterative methods for solving the pressure problem at multiphase filtration Mathematics. *Numerical Analysis*.

Wang, B., Wu, S. and Li, Q. [2013]. Applications of BILU0-GMRES in reservoir numerical simulation. *Acta Petrolei Sinica*, 34(5).

Werneck, L.F., Medeiros de Freitas, M., Guaraldi da Silva Jr, H., Grazione de Souza and Amaral Souto, H.P. [2016]. An OpenMP Parallel Implementation for Numerical Simulation of Gas Reservoirs Using Intel Xeon Phi Coprocessor An OpenMP Parallel Implementation for Numerical Simulation of Gas Reservoirs. Ed: Suzana Moreira Avila, *Proceedings of the XXXVII Iberian Latin-American Congress on Computational Methods in Engineering* , 6-9.

Zaza, A., Awotunde, A. A., Fairag, F. A. and Al-Mouhamed, M. A. [2016]. A CUDA based parallel multi-phase oil reservoir simulator. *Computer Physics Communications*, 206, 2–16.