Performance of FSPAOR iteration for solving one-dimensional space-fractional diffusion equation

Conference Paper *in* Journal of Physics Conference Series · October 2020

DOI: 10.1088/1742-6596/1803/1/012004

3 authors:

Andang Sunarto
Universitas Islam Negeri (UIN) Fatmawati Sukarno, Bengkulu, Indonesia

Jumat Sulaiman
Universiti Malaysia Sabah (UMS)

Jackel Vui Lung Chew
Universiti Malaysia Sabah Labuan International Campus

PAPER • OPEN ACCESS

Performance of FSPAOR iteration for solving one-dimensional space-fractional diffusion equation

To cite this article: A Sunarto *et al* 2021 *J. Phys.: Conf. Ser.* **1803** 012004

 IOP Publishing
1803 (2021) 012004 doi:10.1088/1742-6596/1803/1/012004

Performance of FSPAOR iteration for solving one-dimensional space-fractional diffusion equation

A Sunarto[1], J Sulaiman[2], and J V L Chew[3]

[1] Mathematics Department, IAIN Bengkulu, Kota Bengkulu, Indonesia
[2] Faculty of Science and Natural Resources, UMS, Kota Kinabalu, Malaysia
[3] Faculty of Computing and Informatics, UMS Labuan International Campus, Labuan F.T., Malaysia

Email: andang99@gmail.com

Abstract. This paper considers the numerical solution of a one-dimensional space-fractional diffusion equation. To obtain the solution, we use an unconditionally stable implicit finite difference approximation with the Caputo's space-fractional operator. We study on improving the convergence rate of the solution while solving the generated linear system through the approximation equation iteratively. In our study, we apply the preconditioning technique to construct a preconditioned linear system which eventually derives into a Full-Sweep Preconditioned AOR. From the presented results, we show that the proposed Full-Sweep Preconditioned AOR iterative method has superiority in efficiency compared to the basic Full-Sweep Preconditioned SOR and Full-Sweep Preconditioned Gauss-Seidel iterative methods.

1. Introduction

Fractional partial differential equations (FPDEs), have been extensively studied not only to describe the natural occurrences but also to understand several models in both physical science and social science fields. Based on the brief literature findings, [1] studied on the merging of the local volatility approach and the fractional calculus to extend the Constant Elasticity of Variance model, which is a stochastic volatility model, to the fractional and mixed-fractional cases. This author showed that the fractional and mixed-fractional Constant Elasticity of Variance model could address the smile-skew issue. Then, [2] investigated the use of Caputo's and Atangana-Baleanu's fractional operators to obtain a generalized tuberculosis model with two age groups of human. They also presented a novel numerical approach to solve the formulated fractional model sand showed that the Atangana-Baleanu's operator is more flexible than the Caputo's. Other than that, [3] applied Caputo's fractional-order and integer-order to simulate and study the gross domestic product growth in China. They showed that Caputo's fractional-order model not only able to fit the gross domestic product growth well but also had a better prediction.

Since the application of the fractional calculus provides a better advantage to understand several mathematical models, many researchers proposed numerical methods to solve theFPDEs. From these many proposed methods, [4] consider several finite-difference and element methods in solving the Distributed-Order Time Fractional Diffusion Equations. They have developed three numerical schemes to obtain the solution to the mathematical model accurately. Then, [5] introduced a spectral collocation method based on the Lagrange's polynomials to solve the one-dimensional space fractional diffusion equations approximately. Many researchers initiated the investigation on the application of finite difference method (FDM) for solving FPDEs which can be seen in [6,7,8,9].

Motivated by the simplicity yet unconditional stable implicit scheme of FDM, which is observed from our brief literature review, we aim to discretize the space-fractional diffusion equation (SFDE) via the combination of theimplicit method and Caputo's fractional partial derivative of order β .It is worth to mention that many different iterative methods can be used to solve the linear system efficiently and accurately. Out of the existing linear system solvers, we notice that the preconditioned iterative methods [10,11,12,13] have the potential to be one of the numerical methods that are efficient in solving linear systems. Many researchers have applied preconditioning techniques in their numerical study. For instance, [14] used the preconditioning technique for eigenvalue-counts which eventually demonstrate the efficiency of the computation cost. Then, [15] proposed a circulant preconditioning technique to develop an efficient solution to the fractional diffusion equation. More details about the preconditioning technique for efficiently solving the linear system can be referred in [16]. This author mentioned that an excellent preconditioner to be used to improve the structure of a linear system should be the one that easy to solve and be less complex to be constructed and applied.

Brief literature made based on several preconditioning techniques inspires our research study and becomes the aim of this paper which is to show the formulationof the Caputo's implicit finite difference approximation equation and then investigate on how effective ourproposed Full-Sweep Preconditioned AOR(FSPAOR) iterative method in solving SFDE. In our numerical investigation on the effectiveness of the FSPAOR method, we have implemented two iterative methods, namely the Full-Sweep Preconditioned SOR (FSPSOR) and Full-Sweep Preconditioned Gauss-Seidel (FSPGS) iterative methods. Here, the FSPGS acts as our control method for the numerical experiment.

2. Finite Difference Approximation with Caputo's operator

To start the discretization for the SFDE via the combination of theimplicit scheme and Caputo's space-fractional operator, let us consider the following general form of a parabolic fractional differential equation:

$$\frac{\partial u(x,t)}{\partial t}-a(x)\frac{\partial^{\beta}u(x,t)}{\partial x^{\beta}}-b(x)\frac{\partial u(x,t)}{\partial x}-c(x)u(x,t)-p(x,t)=0, \qquad (1)$$

From the main problem (1), SFDE can be obtained when $a(x)=1$, while the other two, $b(x)$and $c(x)$ are set to be zeroes. We approximate the solution of problem (1) by subjecting Problem (1) with the initial condition $u(x,0)=f(x)$ for $0\le x\le \chi$, and then the boundary condition with the left and right ends are denoted as $u(0,t)=g_0(t)$ and $u(\ell,t)=g_1(t)$, on time interval $0<t\le T$. Also, we introduce some definitions that can be applied as follows.

Definition 1. The fractional integral operator by Riemann-Liouville, J^{β} of order- β is defined as

$$J^{\beta}f(x)=\frac{1}{\Gamma(\beta)}\int_0^x (x-t)^{\beta-1}f(t)dt, \qquad (2)$$

Definition 2. The fractional partial derivative operator by Caputo, D^{β} of order - β is defined as

$$D^{\beta}f(x)=\frac{1}{\Gamma(m-\beta)}\int_0^x \frac{f^{(m)}(t)}{(x-t)^{\beta-m+1}}dt, \qquad (3)$$

with the following properties:

$$D^{\beta}k=0, \text{(k is a constant)},$$

$$D^{\beta}x^{n}=\begin{cases}0, & \text{for } n<[\beta]\\ \dfrac{\Gamma(n+1)}{\Gamma(n+1-\beta)}x^{n-\beta}, & \text{for } n\ge[\beta]\end{cases}$$

Where with m and n are natural numbers. By defining the fixed distance $h=\frac{\ell}{k}$, where k can be any integer that is positive and using the implicit scheme together with the definitions, the fractional space term in problem (1) can be formulated into

$$\frac{\partial^{\beta} u(x_i,t_n)}{\partial x^{\beta}} = \frac{1}{\Gamma(2-\beta)}\int_0^{t_n}\frac{\partial^2 u(x_i,s)}{\partial x^2}(t_n-s)^{1-\beta}\partial s = \frac{1}{\Gamma(2-\beta)}\sum_{j=0}^{i-1}\int_{jh}^{(j+1)h}\left(\frac{U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}}{h^2}\right)(nh\text{-}s)^{\beta}\partial s$$

$$=\frac{h^{-\beta}}{\Gamma(3-\beta)}\sum_{j=0}^{i-1}\left(U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}\right)\left((j+1)^{2-\beta}-j^{2-\beta}\right). \quad (4)$$

Then, we let

$$\sigma_{\beta,h}=\frac{h^{-\beta}}{\Gamma(3-\beta)},$$

and

$$g_j^{\beta}=(j+1)^{2-\beta}-j^{2-\beta},$$

so that equation (4) can be simplified into

$$\frac{\partial^{\beta} u(x_i,t_n)}{\partial x^{\beta}}=\sigma_{\beta,h}\sum_{j=0}^{i-1} g_j^{\beta}\left(U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}\right). \quad (5)$$

Now, by substituting equation (5) back to problem (1) and discretized using the implicit scheme, we may able to approximate problem (1) by using the following equation:

$$\lambda\left(U_{i,n}-U_{i,n\text{-}1}\right)-a_i\sigma_{\beta,h}\sum_{j=0}^{i-1} g_j^{\beta}\left(U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}\right)-b_i\frac{\left(-U_{i-1,n}+U_{i+1,n}\right)}{2h}-C_iU_{i,n}-p_{i,n}=0, \quad (6)$$

for i=1, 2, …, m-1.

The terms in the equation (6) can also be rearranged as

$$\lambda U_{i,n}-a_i\sigma_{\beta,h}\sum_{j=0}^{i-1} g_j^{\beta}\left(U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}\right)+\frac{b_i}{2h}\left(U_{i-1,n}-U_{i+1,n}\right)-C_iU_{i,n}-p_{i,n}=\lambda U_{i,n\text{-}1}, \quad (7)$$

Eventually, from equation (7) we get

$$\therefore\ b_i^{*}U_{i\text{-}1,n}+c_i^{*}U_{i,n}-b_i^{*}U_{i+1,n}-\alpha_i\sum_{j=0}^{i-1} g_j^{\beta}\left(U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}\right)=p_i^{*}, \quad (8)$$

where

$$\alpha_i=a_i\sigma_{\beta,h},\quad b_i^{*}=\frac{b_i}{2h},\quad c_i^{*}=\lambda-c_i,\ \text{and}\quad p_i^{*}=\lambda\left(U_{i,n\text{-}1}\right)+p_i.$$

For $n>3$, equation (8) can be rewritten into the form of

$$\tau_i U_{i\text{-}3,n}+\upsilon_i U_{i\text{-}2}+q_i U_{i\text{-}1,n}+r_i U_{i,n}+s_i U_{i+1,n}=\ell_i, \quad (9)$$

where each term represents

$$\ell_i=p_i^{*}+\alpha_i\sum_{j=0}^{i-1} g_j^{\beta}\left(U_{i-j-1,n}-2U_{i-j,n}+U_{i-j+1,n}\right),$$

$\tau_i=\left(-\alpha_i g_2^{\beta}\right)$, $\upsilon_i=\left(-\alpha_i g_1^{\beta}+2\alpha_i g_2^{\beta}\right)$, $q_i=\left(b_i^{*}-\alpha_i g_2^{\beta}+2\alpha_i g_1^{\beta}-\alpha_i\right)$, $r_i=\left(-\alpha_i g_1^{\beta}+2\alpha_i+c_i^{*}\right)$, and $s_i=\left(-\alpha_i-b_i^{*}\right)$.

Using equation (9), we can construct a system of the linear equation that has the form of

$$A\underset{\sim}{U}=\underset{\sim}{\ell}, \tag{10}$$

where

$$A=\begin{bmatrix} r_1 & s_1 & & & & & & & \\ q_2 & r_2 & s_2 & & & & & & \\ \upsilon_3 & q_3 & r_3 & s_3 & & & & & \\ \tau_4 & \upsilon_4 & q_4 & r_4 & s_4 & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & \tau_{m-3} & \upsilon_{m-3} & q_{m-3} & r_{m-3} & s_{m-3} & & \\ & & & \tau_{m-2} & \upsilon_{m-2} & q_{m-2} & r_{m-2} & s_{m-2} \\ & & & & \tau_{m-1} & \upsilon_{m-1} & q_{m-1} & r_{m-1} \end{bmatrix},$$

$\underset{\sim}{U}=\begin{bmatrix}U_1 & U_2 & \dots & U_{m-2} & U_{m-1}\end{bmatrix}^T$, and

$$\underset{\sim}{\ell}=\begin{bmatrix}\ell_1-q_1U_0 & \ell_2-\upsilon_2U_1-\tau_2U_0 & \ell_3-\tau_3U_2\dots & \ell_{m-1}-s_{m-1}U_m\end{bmatrix}^T.$$

3. FSPAOR Iteration

In this section, we show the derivation of the Full-Sweep Preconditioned Accelerated Over Relaxation (FSPAOR) iterative method for the approximate solutions of the system of linear equation (10). The coefficient matrix A as in linear system (10) isa large scale and sparse. So, to derive the FSPAOR iterative method, we first find the preconditioned linear system to linear system (10) that has the form of

$$A^*\underset{\sim}{x}=\underset{\sim}{f}, \tag{11}$$

with the following matrix transformations,

$A^*=PAP^T$, $\underset{\sim}{f}=P\underset{\sim}{\ell}$, and $\underset{\sim}{U}=P^T\underset{\sim}{x}$.

In this transformation, we defined matrix Pas our preconditioned matrix and the reference therein [9,12]. The matrix Pwith a dimension (M-1)×(M-1) that we use is

$$P=I+S$$

with the components I and S are

$$S=\begin{bmatrix} 0 & -s_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -s_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -s_3 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & -s_{m-1} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

and

$$I=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Now, using the preconditioned matrix (11) to formulate our FSPAOR iterative method, we let the coefficient matrix A^* in the preconditioned matrix (11) be expressed as

$$A^* = D - L - V. \tag{12}$$

Based on the summation of three different matrices shown in (12), the matrices D, L and V that we use are the diagonal, the lower triangular and the upper triangular parts respectively. Hence, by combining equation (11) and (12), the FSPAOR iteration can be derived into the form of [12,7],

$$\tilde{U}^{(k+1)} = (D-\omega L)^{-1}[\beta V + (\beta-\omega)L + (1-\beta)D]\tilde{U}^{(k)} + \beta(D-\omega L)^{-1} f, \tag{13}$$

where $\underset{\sim}{x}^{(k+1)}$ denotes the unknown vector at *(k+1)*th iteration which is we try to compute efficiently.

By referring to equation (13), we have two parameters to be adjusted in order to find the optimum convergence rate for the solution in problem (1). For practise, to choose these two parameters, at first, we let $\beta = 1$ and then implement the iteration cycle using equation (13) with different values of ω within the range (1, 2). The value of ω is selected when the number of iterations reached the least number. Then, using the "optimum" value of ω, again, we implement the iteration cycle using equation (13) with different positive values of β. For more details about the real parameter of β and ω, see in [16].

The way we implement the FSPAOR iteration in solving SFDE can be described as in Algorithm 1.

Algorithm 1: FSPAOR iterative method

i. Set the initial guess $\underset{\sim}{U} = 0$ and $\varepsilon = 10^{-10}$.
ii. For $j = 0,1,2,\dots,n-1$ implement
 a. For $i = 1,2,\dots,m-1$ calculate formula (13)
 b. Check if $\left|\underset{\sim}{U}^{(k+1)} - \underset{\sim}{U}^{(k)}\right| < \varepsilon$ is satisfied, then go to next time level.
iii Display output.

4. Numerical Test

This section shows the numerical result of the proposed FSPAOR using the examples of the SFDE. In this numerical test, we attempt to verify the effectiveness of the FSPAOR method together with the FSPGS and FSPSOR methods. For the comparison purpose, we observethe number iterations (k) and the execution time (seconds) between the three methods to see the performance in terms of efficiency. We also observe the magnitude of maximum error (error) among the three methods to make the solutions obtained are accurate. All comparison analysis are conducted at three different values $\beta = 1.2$, $\beta = 1.5$ and $\beta = 1.8$. To run the experiment C++ program that we build based on algorithm 1, we set the convergence stopping point at $\varepsilon = 10^{-10}$.Other than that, we choose the following space-fractional initial boundary value problem (SFIBVP) for the numerical test.

Example 1:

Let us consider the following general SFIBVP:

$$\frac{\partial U(x,t)}{\partial t} - a(x)\frac{\partial^\beta U(x,t)}{\partial x^\beta} - f(x,t) = 0, \tag{14}$$

with the specification as follows (Azizi& Loghmani, 2013)

$a(x) = 0.25\Gamma(\beta)x^{0.5}$ and $f(x,t) = (x^2+1)\cos(t+1) - 2x\sin(t+1)$.

Example 2 [18]**:**

Let us consider the following SFIBVP:

$$\frac{\partial U(x,t)}{\partial t} - \Gamma(1.2)x^{\beta}\frac{\partial^{\beta}U(x,t)}{\partial x^{\beta}} - 3x^{2}(2x-1)e^{-t} = 0, \tag{15}$$

All numerical results for the tested SFIBVP (14) and (15) are recorded in Tables 1and 2. The three iterations (FSAOR, FSPGS and FSPSOR) are implemented on these SFIBVPs using a different value of mesh size. We use five different mesh values to see the consistency in terms of performance by the three iterations.

5. Conclusion

In conclusion, we have successfully formulated the Caputo's approximation equation to SFDE that leads a large and sparse linear system. We apply the preconditioning technique to get the preconditioned linear system which eventually we use to derive our FSPAOR iterative method. After we test the FSAOR iterative method together with the FSPGS and FSPSOR iterative methods, we found that the FSPAOR method requires the least amount for the number of iterations and execution time among the three methods, particularly when $\beta = 1.2$ .The accuracy of all tested iterative methods is in a good agreement.

6. References

[1] Araneda, A. A. (2020). The fractional and mixed-fractional CEV model. *Journal of Computational and Applied Mathematics*, *363*, 106–123. https://doi.org/10.1016/j.cam.2019.06.006

[2] Fatmawati, Khan, M. A., Bonyah, E., Hammouch, Z., & Shaiful, E. M. (2020). A mathematical model of tuberculosis (TB) transmission with children and adults groups: A fractional model. *AIMS Mathematics*, *5*(4), 2813–2842. https://doi.org/10.3934/math.2020181

[3] Ming, H.; Wang, J.; Fečkan, M. The Application of Fractional Calculus in Chinese Economic Growth Models. *Mathematics* **2019**, *7*, 665.

[4] Bu, W., Xiao, A.,& Zeng, W. (2017).Finite Difference/Finite Element Methods for Distributed-Order Time Fractional Diffusion Equations. *J Sci Comput* **72,** 422–441. https://doi.org/10.1007/s10915-017-0360-8

[5] Nova, M. H., Molla, H. U., & Banu, S. (2017). Comparison of Numerical Approximations of One-Dimensional Space Fractional Diffusion Equation Using Different Types of Collocation Points in Spectral Method Based on Lagrange's Basis Polynomials. *American Journal of Computational Mathematics*, *7*(04), 469–480.https://doi.org/10.4236/ajcm.2017.74034

[6] Muhiddin, F. A., Sulaiman, J., & Sunarto, A. (2019). MKSOR iterative method for the Grünwald implicit finite difference solution of one-dimensional time-fractional parabolic equations. In *AIP Conference Proceedings*. https://doi.org/10.1063/1.5121063

[7] Sunarto, A., Sulaiman, J., & Saudi, A. (2016). Application of the Full-Sweep AOR Iteration Concept for Space-Fractional Diffusion Equation. In *Journal of Physics: Conference Series* (Vol. 710). https://doi.org/10.1088/1742-6596/710/1/012019

[8] Yuste, S.B., Quintana-Murillo, J.(2016) Fast, accurate and robust adaptive finite difference methods for fractional diffusion equations. *Numer Algor* **71,** 207–228. https://doi.org/10.1007/s11075-015-9998-1

[9] Zhang Y., Ding H. (2012) Finite Difference Method for Solving the Time Fractional Diffusion Equation. In: Xiao T., Zhang L., Fei M. (eds) AsiaSim 2012. AsiaSim 2012. Communications in Computer and Information Science, vol 325. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-34387-2_14

[10] Sunarto, A., & Sulaiman, J. (2019). Preconditioned SOR Method to Solve Time-Fractional Diffusion Equations. In *Journal of Physics: Conference Series*. https://doi.org/10.1088/1742-6596/1179/1/012020

[11] Hackbusch, W. (2016). Iterative solution of large sparse systems of equations. In *Applied Mathematical Sciences (Switzerland)* (Vol. 95). https://doi.org/10.2307/2153387

[12] Li, A. (2012). A new preconditioned AOR iterative method and comparison theorems for linear

systems. *IAENG International Journal of Applied Mathematics*, *42*(3), 161–163.

[13] Gunawardena, A. D., Jain, S. K., & Snyder, L. (1991). Modified iterative methods for consistent linear systems. *Linear Algebra and Its Applications*, *154–156*(C), 123–143. https://doi.org/10.1016/0024-3795(91)90376-8

[14] Vecharynski, E., & Yang, C. (2017). Preconditioned iterative methods for eigenvalue counts. In *Lecture Notes in Computational Science and Engineering* (Vol. 117, pp. 107–123). https://doi.org/10.1007/978-3-319-62426-6_8

[15] Fang, Z. W., Ng, M. K., & Sun, H. W. (2019). Circulant preconditioners for a kind of spatial fractional diffusion equations. *Numerical Algorithms*, *82*(2), 729–747. https://doi.org/10.1007/s11075-018-0623-y

[16] Benzi, M. (2002). Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics,182*(2), 418–477, https://doi.org/10.1006/jcph.2002.7176

[17] Hadjidimos, A. (1978). Accelerated overrelaxation method. Mathematics of Computation, 32(141), 149–149.https://doi.org/10.1090/s0025-5718-1978-0483340-6

[18] Azizi, H, and G.B. Loghmani. 2013. Numerical approximation for Space-Fractional Diffusion Equations via Chebyshev Finite Difference Method. Journal of Fractional and Applications.. 4(2): 303–311.

TABLE 1. Numerical result for example 1

Mesh	Method	$\beta = 1.8$			$\beta = 1.5$			$\beta = 1.2$		
		k	**seconds**	**error**	**k**	**seconds**	**error**	**k**	**seconds**	**error**
128	**FSPGS**	345	9.48	3.99e-02	104	2.83	6.20e-04	36	1.09	2.37e-02
	FSPSOR	246	5.76	3.99e-02	80	1.90	6.20e-04	34	0.84	2.37e-02
	FSPAOR	234	5.56	3.99e-02	77	1.84	6.20e-04	33	0.73	2.37e-02
256	**FSPGS**	1123	111.98	3.97e-02	272	27.00	5.69e-04	72	7.23	2.44e-02
	FSPSOR	806	67.75	3.97e-02	211	17.84	5.69e-04	67	5.33	2.44e-02
	FSPAOR	769	66.34	3.97e-02	204	17.51	5.69e-04	64	5.21	2.44e-02
512	**FSPGS**	3659	1398.43	3.96e-02	723	276.20	5.36e-04	151	58.11	2.47e-02
	FSPSOR	2635	843.91	3.96e-02	566	182.83	5.36e-04	129	41.43	2.47e-02
	FSPAOR	2528	828.27	3.96e-02	548	177.13	5.36e-04	127	35.22	2.47e-02
1024	**FSPGS**	11836	2138.11	3.95e-02	1935	945.20	5.13e-04	328	492.56	2.49e-02
	FSPSOR	11829	2099.87	3.95e-02	1514	898.29	5.13e-04	278	472.35	2.49e-02
	FSPAOR	11783	2081.94	3.95e-02	1469	873.87	5.13e-04	272	342.76	2.49e-02
2048	**FSPGS**	47322	8979.18	3.93e-02	8320	4348.68	5.02e-04	1547	1227.21	2.50e-02
	FSPSOR	47289	8852.28	3.93e-02	4052	4299.73	5.02e-04	608	1219.76	2.50e-02
	FSPAOR	47253	8800.61	3.93e-02	4012	4274.43	5.02e-04	597	1195.59	2.50e-02

TABLE 2. Numerical result for example 2

Mesh	Method	$\beta = 1.8$			$\beta = 1.5$			$\beta = 1.2$		
		k	**seconds**	**error**	**k**	**seconds**	**error**	**k**	**seconds**	**error**
128	**FSPGS**	213	5.27	8.88e-04	75	1.83	5.44e-02	27	0.72	1.80e-01
	FSPSOR	166	4.64	8.88e-04	62	1.66	5.44e-02	25	0.50	1.80e-01
	FSPAOR	147	4.18	8.88e-04	56	1.43	5.44e-02	24	0.36	1.80e-01
256	**FSPGS**	686	59.48	4.09e-04	197	17.11	5.58e-02	55	4.72	1.84e-01
	FSPSOR	542	51.40	4.09e-04	164	14.66	5.58e-02	48	2.88	1.84e-01
	FSPAOR	483	50.23	4.09e-04	150	12.41	5.58e-02	45	1.84	1.84e-01
512	**FSPGS**	2213	737.50	1.54e-04	522	170.92	5.65e-02	116	37.86	1.86e-01
	FSPSOR	1756	694.62	1.54e-04	438	163.79	5.65e-02	102	30.90	1.86e-01
	FSPAOR	1569	645.68	1.54e-04	403	152.34	5.65e-02	97	27.94	1.86e-01
1024	**FSPGS**	3452	820.62	1.49e-04	1435	443.81	5.69e-02	250	322.55	1.89e-01
	FSPSOR	2431	809.74	1.49e-04	1391	432.99	5.69e-02	222	310.79	1.89e-01

	FSPAOR	2353	782.32	1.49e-04	1117	421.89	5.69e-02	210	284.91	1.89e-01
2048	**FSPGS**	5127	3173.73	1.20e-04	4125	713.64	5.85e-02	518	413.21	1.88e-01
	FSPSOR	4914	3167.38	1.20e-04	4111	688.32	5.85e-02	498	395.90	1.88e-01
	FSPAOR	4854	3130.75	1.20e-04	4030	672.63	5.85e-02	470	383.87	1.88e-01