IOP Conf. Series: Materials Science and Engineering **725** (2020) 012086 doi:10.1088/1757-899X/725/1/012086

Algorithm solution for space-fractional diffusion equations

A Sunarto[1*] and J Sulaiman[2]

[1] Department of Tadris Matematika , IAIN Bengkulu, Indonesia.
[2] Department of Matematika with economics, Universiti Malaysia Sabah, Malaysia.

*andang99@gmail.com

Abstract. In this study, we propose approximate algorithm solution of the space-fractional diffusion equation (SFDE's) based on a quarter-sweep (QS) implicit finite difference approximation equation. To derive this approximation equation, the Caputo's space-fractional derivative has been used to discretize the proposed problems. By using the Caputo's finite difference approximation equation, a linear system will be generated and solved iteratively. In addition to that, formulation and implementation algorithm the Quarter-Sweep AOR (QSAOR) iterative method are also presented. Based on numerical results of the proposed iterative method, it can be concluded that the proposed iterative method is superior to the FSAOR and HSAOR iterative method.

1. Introduction

In this paper we focus on numerical solution for one-dimensional SFDE's. Generally, linear SFDE's given as follows

$$\frac{\partial U(x,t)}{\partial t}=a(x)\frac{\partial^{\beta} U(x,t)}{\partial x^{\beta}}+b(x)\frac{\partial U(x,t)}{\partial x}+c(x)U(x,t)+f(x,t) \tag{1}$$

With initial condition $U(x,0)=f(x),\ 0\le x\le \ell$, and boundary conditions $U(0,t)=g_0(t)$, $U(\ell,t)=g_1(t),\ 0<t\le T$.

We describe some necessary definitions and mathematical preliminaries of the fractional derivative theory which are required for our subsequent development of the approximation equation for the problem in Eq.(1).

Definition 1.[1,2] The Riemann-Liouville fractional integral operator , J^{β} of order- β is defined as

$$J^{\beta} f(x)=\frac{1}{\Gamma(\beta)}\int_0^x (x-t)^{\beta-1} f(t)dt,\ \beta>0\ x>0 \tag{2}$$

Definition 2.[2, 3] The Caputo's fractional partial derivative operator, D^{β} of order -β is defined as

$$D^{\beta} f(x)=\frac{1}{\Gamma(m-\beta)}\int_0^x \frac{f^{(m)}(t)}{(x-t)^{\beta-m+1}}dt,\ \beta>0 \tag{3}$$

With $m-1<\beta\le m, m\in N,\ x>0$.

In this work, we discretized SFDE's equation using implicit finite difference scheme with Caputo's derivative operator in order to examine the implementation of QSAOR iteration method in solving the resultant linear system of equations. The standard AOR iterative method also known as the FSAOR iterative method and HSAOR is implemented as control method in order to investigate the performance of QSAOR iterative method.

2. Quarter-Sweep Caputo's Implicit Finite Difference Approximation Equations

In this section, the space-fractional diffusion equation (1) is solved. In order to find solution in Eq. (1), let us define $h = \frac{\ell}{m+1}$, where, $m=n+1$ is positive even integer. By implementing definition (2) we obtain

$$\frac{\partial^{\beta} Y(x_i, t_n)}{\partial x^{\beta}} = \frac{(4h)^{-\beta}}{\Gamma(3-\beta)} \sum_{j=0,4,8}^{i-4} \left(Y_{i-j+4,n} - 2Y_{i-j,n} + Y_{i-j-4,n}\right)\left(\left(\frac{j}{4}+1\right)^{2-\beta} - \frac{j}{4}^{2-\beta}\right) \quad (4)$$

Then the discrete approximation equation (4) can be written as

$$\frac{\partial^{\beta} Y(x_i, t_n)}{\partial x^{\beta}} = \sigma_{\beta,4h} \sum_{j=0,4,8}^{i-4} g_j^{\beta} \left(Y_{i-j+4,n} - 2Y_{i-j,n} + Y_{i-j-4,n}\right) \quad (5)$$

with $\sigma_{\beta,4h} = \frac{(4h)^{-\beta}}{\Gamma(3-\beta)}$, $g_j^{\beta} = \left(\frac{j}{4}+1\right)^{2-\beta} - \frac{j}{4}^{2-\beta}$.

With apply Eq. (5) and QS implicit Caputo's finite difference scheme, we approximate the problem in Eq. (1) in order to derive the QS implicit Caputo's finite difference approximation equation as follows

$$\lambda\left(Y_{i,n} - Y_{i,n-4}\right) = a_i \sigma_{\beta,4h} \sum_{j=0.4.8}^{i-4} g_j^{\beta}\left(Y_{i-j+4,n} - 2Y_{i-j,n} + Y_{i-j-4,n}\right) + b_i \frac{\left(Y_{i+4,n} - Y_{i-4,n}\right)}{8h} + C_i Y_{i,n} + f_{i,n} \quad (6)$$

For $i = 4,8,\ldots m-4$. Again based on the approximation equation (6), we have

$$\lambda Y_{i,n-4} = -a_i \sigma_{\beta,4h} \sum_{j=0,4,8}^{i-4} g_j^{\beta}\left(Y_{i-j+4,n} - 2Y_{i-j,n} + Y_{i-j-4,n}\right) - \frac{b_i}{8h}\left(Y_{i+4,n} - Y_{i-4,n}\right) - C_i Y_{i,n} + \lambda Y_{i,n} - f_{i,n} \quad (7)$$

Then by simplifying Eq.(7), it can be shown

$$b_i^* Y_{i-4,n} + \left(\lambda - c_i^*\right) Y_{i,n} - b_i^* Y_{i+4,n} - a_i^* \sum_{j=0,4,8}^{i-4} g_j^{\beta}\left(Y_{i-j+4,n} - 2Y_{i-j,n} + Y_{i-j-4,n}\right) = f_i \quad (8)$$

with $a_i^* = a_i \sigma_{\beta,4h}$, $b_i^* = \frac{b_i}{8h}$, $c_i^* = c_i$, $F_i^* = f_{i,n}$, $f_i = \lambda\left(U_{i,n-4}\right) + F_i^*$.

Let us notice the approximation equation (8) being rewritten in the following form

$$-R_i + \alpha_i Y_{i-12,n} + s_i Y_{i-8} + p_i Y_{i-4,n} + q_i Y_{i,n} + r_i Y_{i+4,n} = f_i \quad (9)$$

With $R_i = a_i^* \sum_{j=12}^{i-4} g_j^{\beta}\left(Y_{i-j+4} - 2Y_{i-j,n} + Y_{i-j-4,n}\right)$, $\alpha_i = \left(-a_i^* g_2^{\beta}\right)$,

$s_i = \left(-a_i^* g_1^{\beta} + 2a_i^* g_2^{\beta}\right)$, $p_i = \left(b_i^* - a_i^* g_2^{\beta} + 2a_i^* g_1^{\beta} - a_i^*\right)$, $q_i = \left(-a_i^* g_1^{\beta} + 2a_i^* + \left(\lambda - c_i^*\right)\right)$, $r_i = \left(-a_i^* - b_i^*\right)$.

By applying Eq.(9) into all interior points of the solution domain problem in Eq (1), the linear system to be expressed in matrix form as

$$A \underset{\sim}{Y} = \underset{\sim}{f} \quad (10)$$

with

$$A = \begin{bmatrix} q_4 & r_4 & & & & & & & \\ p_8 & q_8 & r_8 & & & & & & \\ s_{12} & p_{12} & q_{12} & r_{12} & & & & & \\ \alpha_{16} & s_{16} & p_{16} & q_{16} & r_{16} & & & & \\ & \alpha_{20} & s_{20} & p_{20} & q_{20} & r_{20} & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & \alpha_{m-8} & s_{m-8} & p_{m-8} & q_{m-8} & r_{m-8} & \\ & & & & \alpha_{m-4} & s_{m-4} & p_{m-4} & q_{m-4} & \end{bmatrix}_{(m-4)X(m-4)}$$

$$\underset{\sim}{Y} = \begin{bmatrix} Y_{4,1} & Y_{8,1} & Y_{12,1} & \cdots & Y_{m-4,1} & Y_{m-2,1} \end{bmatrix}^T,$$

$$\underset{\sim}{f} = \begin{bmatrix} f_4 + p_4 Y_{4,1} & f_8 + s_8 Y_{8,1} & f_{12} + \alpha_{12} Y_{12,1} & f_{16} + R_i & \cdots & f_{m-8,1} + R_{m-8} & f_{m-4,1} + p_{m-4} Y_{m,1} + R_{m-4} \end{bmatrix}^T$$

3. Formulation of QSAOR Iterative Method

In this paper, FSAOR, HSAOR and QSAOR iterative methods will be applied to solve linear system generated from the discretization of the problem in Eq.(1) as shown in Eq.(10). To derive the formulation of both proposed methods, let the coefficient matrix A in Eq.(10) be expressed as

$$A = D - L - V \tag{11}$$

Where D, L and V are diagonal, strictly lower triangular and strictly upper triangular matrices respectively [4, 5, and 6]. Then, based on Eq. (11) the general scheme for the QSAOR iterative method can be shown as [7, 8, 9, and 10]

$$\tilde{U}^{(k+1)} = (D - \omega L)^{-1}[\beta V + (\beta - \omega)L + (1-\beta)D]\tilde{U}^{(k)} + \beta(D - \omega L)^{-1} f \tag{12}$$

Where $\tilde{U}^{(k)}$ represents an unknown vector at k^{th} iteration. Basically, the general algorithm for QSAOR iterative method to solve linear system (10) would be generally described in Algorithm 1.

Algorithm 1: QSAOR method

i. Initialize $\tilde{U} \leftarrow 0$ and $\varepsilon \leftarrow 10^{-10}$.

ii. For $j = 0,1,2,\ldots,n-1$ implement

a. For $i = 1p, 2p, \ldots, m-p$ calculate

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

b. Convergence test. If the convergence criterion i.e. $\left\|\tilde{U}^{(k+1)} - \tilde{U}^{(k)}\right\| \le \varepsilon = 10^{-10}$ is satisfied, go to next time level. Otherwise go back to Step (ii).

Iii Display approximate solutions.

However, If p=1, Algorithm 1 will be named as FSAOR

4. Numerical Experiments

For the numerical experiments, two examples were considered to verify the effectiveness of the implementation of Algorithm the QSAOR iterative method. To comparison between FSAOR, HSAOR and QSAOR methods, three criteria will be considered such as number of iterations (K), execution time (second) and maximum error at three different values of $\beta = 1.2$, $\beta = 1.5$ and $\beta = 1.8$ with different mesh sizes as 128, 256, 512, 1024 and 2048. In implementations of two numerical experiments, the convergence test considered the tolerance error, $\varepsilon = 10^{-10}$. Results of numerical experiments, which were obtained from implementations Algorithm of the FSAOR, HSAOR and QSAOR iterative method, have been recorded in Tables 1 and 2 respectively.

Example 1: [3]

We consider the following space-fractional initial boundary value problem

$$\frac{\partial U(x,t)}{\partial t} = d(x)\frac{\partial^{\beta} U(x,t)}{\partial x^{\beta}} + p(x,t), \tag{13}$$

At finite domain $0 \le x \le 1$, with the diffusion $d(x) = \Gamma(\beta)x^{0.5}$.

Example 2: [3]

We consider the following space-fractional initial boundary value problem

$$\frac{\partial U(x,t)}{\partial t} = \Gamma(1.2)x^{\beta}\frac{\partial^{\beta} U(x,t)}{\partial x^{\beta}} + 3x^2(2x-1)e^{-t}, \tag{14}$$

With the initial condition $U(x,0) = x^2 - x^3$ and zero Dirichlet conditions.

Table 1. Comparison between number of iterations (K), the execution time (second) and maximum errors for the iterative methods using example at $\beta = 1.2, 1.5, 1.8$

M	Method	β = 1.2			β = 1.5			β = 1.8		
		K	**Time**	**Max Error**	**K**	**Time**	**Max Error**	**K**	**Time**	**Max Error**
128	**FSAOR**	65	1.32	2.37e-02	188	3.88	6.21e-04	269	5.35	3.99e-02
	HSAOR	46	0.53	2.24e-02	78	0.83	6.99e-04	225	2.13	4.03e-02
	QSAOR	22	0.11	1.99e-02	40	0.13	8.19e-04	90	0.23	4.11e-02
256	**FSAOR**	128	10.00	2.44e-02	370	28.88	5.69e-04	756	58.90	3.97e-02
	HSAOR	77	2.94	2.37e-02	204	7.70	6.21e-04	732	28.08	3.99e-02
	QSAOR	38	0.39	2.24e-02	96	0.16	6.99e-04	282	1.61	4.03e-02
512	**FSAOR**	270	84.05	2.47e-02	983	104	5.35e-04	2497	703	3.96e-02
	HSAOR	129	19.88	2.44e-02	544	83.61	5.69e-04	2388	368.65	3.97e-02
	QSAOR	73	1.69	2.37e-02	247	5.38	6.22e-04	912	19.44	3.99e-02
1024	**FSAOR**	577	125	2.49e-02	3640	689	5.13e-04	5220	1119	2.36e-02
	HSAOR	278	179.11	2.47e-02	1457	502	5.35e-04	4098	982	3.38e-02
	QSAOR	150	12.59	2.44e-02	677	58.45	5.68e-04	2971	246.77	3.97e-02
2048	**FSAOR**	1150	540	2.52e-02	5950	3102	5.09e-04	13203	3920	2.30e-02
	HSAOR	606	424	2.49e-02	3885	2035	5.24e-04	11376	3256	2.35e-02
	QSAOR	321	112.5	2.47e-02	1751	614.16	5.36e-04	9653	2977	3.96e-02

Table 2. Comparison between number of iterations (K), the execution time (second) and maximum errors for the iterative methods using example at $\beta = 1.2, 1.5, 1.8$

M	Method	β = 1.2			β = 1.5			β = 1.8		
		K	**Time**	**Max Error**	**K**	**Time**	**Max Error**	**K**	**Time**	**Max Error**
128	**FSAOR**	48	0.93	1.80e-01	133	1.41	5.44e-02	148	1.52	1.25e-04
	HSAOR	34	0.45	1.73e-01	55	0.70	5.16e-02	135	1.24	1.76e-04
	QSAOR	20	0.09	1.59e01	24	0.08	4.61e-02	46	0.16	3.29e-04
256	**FSAOR**	97	3.58	1.84e-01	197	10.93	5.58e-02	457	16.66	1.44e-04
	HSAOR	55	2.67	1.81e-01	145	6.91	5.44e-02	439	11.61	8.88e-04
	QSAOR	29	0.27	1.73e-01	59	0.42	5.16e-02	147	0.87	1.76e-04

512	**FSAOR**	106	18.71	5.39e-01	525	83.02	1.28e-02	1357	193.83	1.53e-04
	HSAOR	97	17.52	1.84e-01	386	73.38	5.58e-02	1147	101.20	4.09e-04
	QSAOR	49	1.08	1.80e-01	155	23.30	5.44e-02	475	49.98	8.8e-04
1024	**FSAOR**	213	168	5.45e-01	1298	198	1.32e-02	4329	2103	1.25e-04
	HSAOR	209	150.23	1.86e-01	1030	160	5.65e-02	3731	1984.23	1.54e-04
	QSAOR	103	28.37	1.84e-01	413	33.56	5.58e-02	1538	426.05	4.09e-04
2048	**FSAOR**	815	398	1.92e-01	2506	912	5.73e-02	6520	3834	2.30e-04
	HSAOR	456	273	1.86e-01	2326	878	5.80e-02	6290	3462	2.45e-04
	QSAOR	220	75.40	1.86e-01	1099	378.68	5.65e-02	4940	1714	1.54e-04

5. Conclusion

In this work, we discussed the implementation algorithm of the QSAOR iterative algorithm which uses two accelerated parameter. The QSAOR Algorithm has performance good speedup and efficiency for computational time and number of iterations. Again, the QSAOR algorithm has shown their superiority over the FSAOR and HSAOR algorithm. For our future works, this study can be extended to investigate on the use of the AOR to combine with the concept pre-conditioner iterative family.

References

[1] Zhang, Y. 2009. A Finite Difference Method for Fractional Partial Differential Equation. *Applied Mathematics and Computation*.215:524-529.
[2] Li, C., D. Qian, and Y.Q. Chen. 2011. On Riemann-Liouville and Caputo Derivatives. *Hindawi Publishing Corporation Discrete Dynamics in Nature and Science*. 1: 1–15.
[3] 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.
[4] Hasan, M.K., M. Othman, Z. Abbas, J. Sulaiman, and F. Ahmad. 2007. Parallel Solution of High Speed Low Order FDTD on 2D Free Space Wave Propagation. *Lecturer Notes in Computer Science LNCS 4706*.13-24.
[5] Sunarto, A., J. Sulaiman, and A. Saudi. 2014. Half-Sweep Accelerated Over-Relaxations Iterative Method for the Solution Time-Fractional Diffusion Equations. *Simposium Kebangsaan Sains Mathematiks ke 22.* Shah-Alam, Malaysia. 24-26 November 2014. 109-115.
[6] Young, D.M. 1954. Iterative Methods for Solving Partial Difference Equations of Elliptic Type. *Transaction of The AMS-American Mathematical Society.* 76:92-111.
[7] Young, D.M.1971. *Iterative Solution of Large Sparse Systems.*. London: Academic Press.
[8] Young, D.M. 1972. Second-Degree Iterative Methods for The Solution of Large Linear Systems. *Journal of Approximation Theory.* 15:37-148.
[9] Hadjidimos, A. 1978. Accelerated Over Relaxation Method. *Mathematics of Computation.* 32:149-157
[10] Tian, H. 2003. Accelerated Over-relaxation Method for Rank Deficient Linear Systems. *Applied Mathematics and computation.* 14:485-499.