Four-Point EGSOR Iteration for the Grünwald Implicit Finite Difference Solution of One-Dimensional Time-Fractional Parabolic Equations

F A Muhiddin¹ J Sulaiman² and A Sunarto³

¹Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA Cawangan Sabah, Malaysia.
²Mathematics with Economics Programme, Universiti Malaysia Sabah, Malaysia.
³Faculty of Economic and Islamic Business, IAIN Bengkulu, Indonesia.

E-mail: ¹fatih170@uitm.edu.my, ²jumat@ums.edu.my, ³andang99@gmail.com

Abstract. In this paper, our main concern is on the application of the formulation of a four-point explicit group successive over-relaxation (4EGSOR) iterative method in solving one-dimensional time-fractional parabolic equations based on the second-order Grünwald implicit approximation equation. The formulation of the 4EGSOR method is constructed by using the implicit approximation equation which is derived by the Grunwald derivative operator and the implicit finite difference discretization scheme. In order to access the performance results of the 4EGSOR iterative method, another block and point iterative methods which are the four-point EGGS (4EGGS) and the Gauss-Seidel (GS) were also presented as control methods. The results of three numerical experiments show substantial improvement in terms of the number of iterations and execution time.

1. Introduction

The development of effective numerical techniques for the time-fractional partial differential equations (TFPDEs) have evolved significantly in the past few decades. The claims made on its ability to model real world problems better, have caused spark of interest among numerical analysis research community around the globe. Its application in diverse fields are mentioned in [1-3], such as but not limited to physics, chemistry, engineering, and even economics. In this paper, we consider the one-dimensional inhomogeneous time-fractional parabolic equation as given by

\[ \frac{\partial^\alpha U}{\partial t^\alpha} + p(x) \frac{\partial U}{\partial x} = q(x) \frac{\partial^2 U}{\partial x^2} + f(x,t), \quad a \leq x \leq b, \quad 0 \leq t \leq T, \quad 0 < \alpha \leq 1, \tag{1} \]

subject to the following initial and boundary conditions:

\[ U(x,0) = g_1(x), \quad a \leq x \leq b, \]

\[ U(a,t) = g_2(t); \quad U(b,t) = g_3(t) \quad 0 < t \leq T, \]

where the function \( f(x,t) \) is a source term and note that for \( \alpha = 1 \), equation (1) is the classical parabolic partial differential equation.
While dealing with complex equation as problem (1), solving it numerically seems to be more realistically efficient in both time and cost factors. Some earlier studies have suggested the use of compact difference scheme [4], Crank-Nicholson scheme [5], Chebyshev collocation method [6] and implicit scheme [7-10]. Meanwhile, in the last couple of years researchers have studies the use of reduced spline (RS) method based on a proper orthogonal decomposition (POD) technique [11], high order compact difference method [12], Crank-Nicholson scheme with finite element method [13] and alternating segment explicit-implicit/ implicit-explicit parallel difference method [14]. Mostly, the description of their solutions is done based on Caputo sense.

Solving equation (1) using numerical techniques will generate a system linear of equations (SLEs). While dealing with small size of SLEs, direct method is still acceptable. However, when it involves a large linear system with its sparse matrix, iterative method such as the Gauss-Seidel (GS) method could speed up the process of iteration. Apart from the GS iterative method, there are also Successive Over-Relaxation (SOR) method by [15-16] which also belongs to the family of point-iterative methods, but have better performance in compared to its predecessor the GS iterative method. Later than that, [17] introduced more interesting concept which allow the SLEs to be solved in block-to-block instead of point-by-point making the existence iterative method to converge quicker. Based on his work, further discussions have been done by [18-21].

Extracted from our reading, apparently there are still limited studies available in the literatures which related to iterative method for solving one-dimensional TFPE. Some earlier works have been done by [8-10], but their works are based on Caputo derivative operator and limited to point-iterative method only. This being the case and based on the claims of the block iterative method with its excellence result, this paper investigates the effectiveness of four-Point EGSOR (4EGSOR) iterative method for solving second-order Grünwald implicit approximation equation of the one-dimensional TFPDEs.

First, let us recall the two most common fractional derivative operators that ever used by researchers before. There are Riemann-Liouville and Caputo fractional derivative operators which define by [22] as

**Definition 1:** Riemann-Liouville fractional derivative

\[
D^\alpha_{RL} f(t) = \frac{1}{\Gamma(1-\alpha)} \frac{d}{dt} \int_0^t \frac{f(\tau)}{(t-\tau)^{\alpha+1}} d\tau, \quad p - 1 \leq \alpha < p
\]  

**Definition 2:** Caputo fractional derivative

\[
D^\alpha_{C} f(t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{f^{(\nu)}(\tau)}{(t-\tau)^{\alpha+1}} d\tau, \quad p - 1 \leq \alpha < p
\]

However, in this study we apply the definition of the Grünwald fractional derivative operator as defined by [23-24]

**Definition 3:** The Grünwald fractional derivative of order \( p - 1 < \alpha < p \) of a function \( f(t) \) is defined as

\[
D^\alpha_G f(t) = \frac{1}{(\Delta t)^{\alpha}} \lim_{N \to \infty} \sum_{k=0}^{N} g_{\alpha,k} f(t - k\Delta t), \quad 0 < \alpha < 1
\]

where the Grünwald weights are

\[
g_{\alpha,k} = \frac{\Gamma(k-\alpha)}{\Gamma(-\alpha) \Gamma(1+k) \Delta t^\alpha}.
\]

Meanwhile, for simplicity, the solution domain (1) is assumed to be uniformly divided into \( N = 2^r, r \geq 2 \) in directions of \( x \) and \( t \), which the subintervals are denoted as \( \Delta x \) and \( \Delta t \) respectively and defined as
\[ \Delta x = \frac{(b-a)}{N}, \quad n = N-1, \quad \Delta t = \frac{T}{M} \]  

Figure 1 illustrates the finite grid network that is used to construct the approximation equations, where the GS iterative method is applied onto each interior node point until iterative convergence is achieved.

Figure 1: the distribution of uniform node points for the solution domain at N=16.

2. Formulation of Grünwald Implicit Finite Difference Approximation

This section discussed the discretization of problem (1) by using formulation of the Grünwald fractional derivative operator and implicit difference scheme in time and space respectively. To derive numerical approximations based on Grünwald derivative operator, first let the mesh point \( x_i = \alpha + i h \), where \( i = 0,1,...,N \) and \( h = \Delta x \) as in equation (2), denotes the uniform step-size. Hence, the discrete form of equation (1) can be written generally as

\[
\frac{1}{(\Delta t)^\alpha} \sum_{k=0}^{N} g_{\alpha,k} U_{i,j-k} + \frac{p(x)}{2\Delta x} \left(U_{i+1,j} - U_{i-1,j}\right) + \frac{q(x)}{(\Delta x)^2} \left(U_{i+1,j} - 2U_{i,j} + U_{i-1,j}\right) = f_{i,j}
\]  

Then, equation (6) can be simplified and stated as

\[
\alpha U_{i-1,j} + \beta U_{i,j} + \gamma U_{i+1,j} = F_{i,j}
\]  

by letting

\[
G_k = \frac{g_{\alpha,k}}{(\Delta t)^\alpha}, \quad \rho_k = \frac{p(x)}{2\Delta x}, \quad \lambda_k = \frac{q(x)}{(\Delta x)^2},
\]

where \( \alpha_i = \lambda_i - \rho_i, \quad \beta_i = G_0 - 2\lambda_i, \quad \gamma_i = \lambda_i + \rho_i, \)

and where

\[
F_{i,j} = \begin{cases}  
  f_{i,j} - G_0 U_{i,0} & j = 1  
  \sum_{k=1}^{j} G_k U_{i,j-k} & j = 2,3,\ldots
\end{cases}
\]

Whereas, the illustration of computational molecules at time levels \( j=1 \) and \( j=2 \) for equation (7) are as pictured in Figures 2 (a) and (b) respectively. Based on the truncation error, equation (7) has the accuracy of the order \( O\left((\Delta x)^2 + (\Delta t)^2\right) \).
FIGURE 2. (a) The computational molecule for Grünwald time-fractional parabolic approximation equation at time level \( j=1 \) and (b) The computational molecule for Grünwald time-fractional parabolic approximation equation at time level \( j=2 \).

Then, equation (7) can be expressed in a matrix form as

\[
AU_j = F_j
\]  

where

\[
A = \begin{bmatrix}
\beta_1 & \gamma_1 \\
\alpha_2 & \beta_2 & \gamma_2 \\
\alpha_3 & \beta_3 & \gamma_3 & \ddots & \ddots & \ddots \\
\alpha_n & \beta_n & \gamma_n & \ddots & \ddots & \ddots & \ddots \\
\end{bmatrix}
\]

\[
U_j = \begin{bmatrix}
U_{1,j} \\
U_{2,j} \\
\vdots \\
U_{n,j}
\end{bmatrix},
\]

\[
F_j = \begin{bmatrix}
F_{1,j} - \alpha_0 U_{0,j} \\
F_{2,j} \\
\vdots \\
F_{n,j} - \gamma_{n-1} U_{n-1,j}
\end{bmatrix},
\]

3. Derivation of Four-Point EGSOR Iterative Method

To derive the formulation of the four-Point EGSOR iterative method, let consider a group of four points \((4 \times 4)\). Figure 3 illustrates the finite grid network that is used to construct the approximation equations, where the EGSOR iterative method is applied onto each four- and three-Point blocks until iterative convergence is achieved.

![Figure 3: the distribution of uniform node points for 4EGSOR at the solution domain for \(N=16\).](image)

By considering the second-order Grünwald implicit approximation equation (7), for \(i = 1, 5, 9, 13, \ldots, n - 6\), the four-point blocks can be generally expressed as

\[
\begin{bmatrix}
b_{i,1} & 1 & 0 & 0 \\
0 & a_{i+1} & b_{i+1} & 1 \\
0 & 0 & a_{i+2} & b_{i+2} \\
0 & 0 & 0 & a_{i+3} & b_{i+3}
\end{bmatrix}
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j} \\
U_{i+3,j}
\end{bmatrix}
= \begin{bmatrix}
S_1 \\
S_2 \\
S_3 \\
S_4
\end{bmatrix}
\]  

(9)

where

\[
S_1 = U_{i,j} - a_0 U_{i-1,j},
\]

\[
S_2 = U_{i+1,j},
\]

\[
S_3 = U_{i+2,j},
\]

\[
S_4 = U_{i+3,j} - U_{i+4,j},
\]

and where

\[
a_i = \frac{\alpha_i}{\gamma_i} \quad \text{and} \quad b_i = \frac{\beta_i}{\gamma_i}.
\]
Now, by determining the inverse of the coefficient matrix in equation (9) and adding one weighted parameter, \( \omega \), the formulation of the four-point EGSOR (4EGSOR) method can be stated as

\[
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j} \\
U_{i+3,j}
\end{bmatrix}^{(k+1)} = (1 - \omega)
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j} \\
U_{i+3,j}
\end{bmatrix}^{(k)} + \omega
\begin{bmatrix}
b_i & 1 & 0 & 0 \\
0 & a_{i+1} & b_{i+1} & 1 \\
0 & 0 & a_{i+2} & b_{i+2} \\
0 & 0 & a_{i+3} & b_{i+3}
\end{bmatrix}
\begin{bmatrix}
S_1 \\
S_2 \\
S_3 \\
S_4
\end{bmatrix}
\]

(10)

where the range value of \( \omega \) is given by \( 1 \leq \omega \leq 2 \). As taking \( \omega = 1 \), the four-point EGSOR method can be reduced to the four-point EGSS (4EGSS) iterative method. Referring to Figure 3, we have a group of group of three points for \( i = n-2, n-1 \) and \( n \) in which this group will be considered as an ungroup point case. For this group, let the last three-point block be stated as

\[
\begin{bmatrix}
b_{n-2} & 1 & 0 \\
a_{n-1} & b_{n-1} & 1 \\
0 & a_n & b_n
\end{bmatrix}
\begin{bmatrix}
U_{n-2,j} \\
U_{n-1,j} \\
U_{n,j}
\end{bmatrix} = \begin{bmatrix}
S_{n-2} \\
S_{n-1} \\
S_n
\end{bmatrix}
\]

(11)

where

\[
S_{n-2} = U_{n-2,j} - a_{n-2}U_{n-3,j},
S_{n-1} = U_{n-1,j},
S_n = U_{n,j} - U_{n+1,j+1}.
\]

Again, by determining the inverse of the coefficient matrix in equation (11), the formulation of the three-point EGSOR (3EGSOR) method with a weighted parameter, \( \omega \) can be given as

\[
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j}
\end{bmatrix}^{(k+1)} = (1 - \omega)
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j}
\end{bmatrix}^{(k)} + \omega
\begin{bmatrix}
b_i & 1 & 0 & 0 \\
0 & a_{i+1} & b_{i+1} & 1 \\
0 & 0 & a_{i+2} & b_{i+2}
\end{bmatrix}
\begin{bmatrix}
S_{n-2} \\
S_{n-1} \\
S_n
\end{bmatrix}
\]

(12)

As given \( \omega = 1 \), the method (12) can be assigned as the three-point EGSS (3EGSS) iterative method. Based on equations (10) and (12), the general algorithm for implementation of the 4-point EGSOR iterative method for solving problem (1) could be described as in Algorithm 1.

---

**Algorithm 1: 4EGSOR scheme**

i. Initialize \( U_i^{(0)} \leftarrow 0 \) and \( \varepsilon \leftarrow 10^{-10} \)

ii. Assign the optimal value of \( \omega = 1 \),

iii. For \( i = 1, 5, 9, \ldots, n-6 \), calculate any complete group case

\[
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j} \\
U_{i+3,j}
\end{bmatrix}^{(k+1)} = (1 - \omega)
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j} \\
U_{i+3,j}
\end{bmatrix}^{(k)} + \omega
\begin{bmatrix}
b_i & 1 & 0 & 0 \\
0 & a_{i+1} & b_{i+1} & 1 \\
0 & 0 & a_{i+2} & b_{i+2} \\
0 & 0 & a_{i+3} & b_{i+3}
\end{bmatrix}
\begin{bmatrix}
S_1 \\
S_2 \\
S_3 \\
S_4
\end{bmatrix}
\]

For \( i = n-2 \), calculate an ungroup case

\[
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j}
\end{bmatrix}^{(k+1)} = (1 - \omega)
\begin{bmatrix}
U_{i,j} \\
U_{i+1,j} \\
U_{i+2,j}
\end{bmatrix}^{(k)} + \omega
\begin{bmatrix}
b_i & 1 & 0 \\
0 & a_{i+1} & b_{i+1} \\
0 & 0 & a_{i+2} & b_{i+2}
\end{bmatrix}
\begin{bmatrix}
S_{n-2} \\
S_{n-1} \\
S_n
\end{bmatrix}
\]
iv. Perform the convergence test, \( \left| U_{i,j}^{(k+1)} - U_{i,j}^{(k)} \right| \leq \varepsilon = 10^{-10} \). If yes, go to step (v). Otherwise repeat step (iii).

v. Display approximate solutions.

4. Numerical Experiments

In this section, we examine the performance of the 4EGSOR iterative method with the Grünwald finite difference approximation equations, for solving the one-dimensional inhomogeneous time-fractional parabolic equations. Throughout the experiments, the tolerance error is set to \( \varepsilon = 10^{-10} \).

Example 1: [24]

Consider the following one-dimensional linear inhomogeneous time-fractional parabolic equation

\[
\frac{\partial^\alpha U(x,t)}{\partial x^\alpha} + \frac{\partial U(x,t)}{\partial x} - \frac{\partial^2 U(x,t)}{\partial x^2} = \frac{2t^2-\alpha}{\Gamma(3-\alpha)} \sin(2\pi x) + 2x - 2, \quad t > 0, x \in R, 0 < \alpha \leq 1,
\]

with \( p(x) = 1 \) and \( q(x) = -1 \) and subject to the initial condition, \( U(x,0) = x^2 \).

The exact solution is \( U(x,t) = x^2 + t^2 \).

Example 2: [25]

Consider the following one-dimensional linear inhomogeneous time-fractional parabolic equation

\[
\frac{\partial^\alpha U(x,t)}{\partial x^\alpha} - \frac{\partial^2 U(x,t)}{\partial x^2} = \frac{2t^2-\alpha}{\Gamma(3-\alpha)} \sin(2\pi x) + 4\pi^2 t^2 \sin(2\pi x), \quad t > 0, x \in R, \quad 0 < \alpha \leq 1,
\]

with \( p(x) = 0 \) and \( q(x) = -1 \) and subject to the initial condition, \( U(x,0) = 0 \).

The exact solution is \( U(x,t) = t^2 \sin(2\pi x) \).

Example 3: [26]

Consider the following one-dimensional linear inhomogeneous time-fractional equation

\[
\frac{\partial^\alpha U(x,t)}{\partial x^\alpha} - \frac{\partial^2 U(x,t)}{\partial x^2} = \frac{2e^\alpha t^2-\alpha}{\Gamma(3-\alpha)} \tau^2 e^\tau, \quad t > 0, x \in R, \quad 0 < \alpha \leq 1,
\]

with \( p(x) = 0 \) and \( q(x) = -1 \) and subject to the initial condition, \( U(x,0) = 0 \).

The exact solution is \( U(x,t) = t^2 e^t \).

Then, the results of numerical experiment for the three examples obtained from the implementation of 4EGSOR, 4EGGS and GS iterative methods are recorded in Tables 1 to 3. The results comprise of three parameters which are the iteration numbers (Iter), computation time (Second) and maximum absolute errors (Max Error) at \( \alpha = 0.333, 0.666, 0.999 \).

| \( \alpha = 0.333 \), 0.666, 0.999 | \( \alpha = 0.333 \), 0.666, 0.999 | \( \alpha = 0.333 \), 0.666, 0.999 |
|------------------------------|-------------------------------|-------------------------------|
| k   | Time | Max Error | k   | Time | Max Error | k   | Time | Max Error |
|-----|------|-----------|-----|------|-----------|-----|------|-----------|
| 128 | GS   | 18325     | 7.89| 2.5972e-02 | 8957 | 5.36| 1.3065e-02 | 2772 | 3.69| 1.2478e-03 |
| 4EGGS | 4972 | 2.78 | 2.5972e-02 | 2426 | 2.26| 1.3065e-02 | 752 | 1.95| 1.2480e-03 |
| 4EGSOR | 202 | 1.90 | 2.5972e-02 | 141 | 1.87| 1.3065e-02 | 82 | 1.86 | 1.2480e-03 |
| 256 GS | 67139 | 41.9 | 2.5972e-02 | 32944 | 23.62| 1.3065e-02 | 10244 | 11.42| 1.2473e-03 |
| 4EGGS | 18330 | 10.4 | 2.5972e-02 | 8961 | 7.02| 1.3065e-02 | 2775 | 4.67| 1.2478e-03 |
| 4EGSOR | 403 | 3.83 | 2.5972e-02 | 277 | 3.75| 1.3065e-02 | 163 | 3.73 | 1.2478e-03 |
| 512 GS | 243922 | 276.4 | 2.5972e-02 | 120271 | 142.56| 1.3065e-02 | 37649 | 53.24| 1.2454e-03 |
| 4EGGS | 67144 | 57.81 | 2.5972e-02 | 32949 | 32.15| 1.3065e-02 | 10247 | 15.03| 1.2478e-03 |
| 4EGSOR | 801 | 8.18 | 2.5972e-02 | 551 | 7.80| 1.3065e-02 | 320 | 7.63 | 1.2480e-03 |
| 1024 GS | 877165 | 1914.74 | 2.5981e-02 | 435083 | 965.24| 1.3074e-02 | 137338 | 323.96| 1.2376e-03 |
| 4EGGS | 243927 | 380.3 | 2.5975e-02 | 120277 | 195.6| 1.3067e-02 | 37654 | 71.65 | 1.2454e-03 |
observed that by applying four-point block iteration concept to the original GS point iterative method making the 4EGGS reduced its iteration numbers in about 71.84-80.14% for Example 1, 67.45-72.64% for Example 2 and 71.91-72.96% for Example 3. However, further application of the 4EGSOR block iteration has tremendously improved the reduction of the iteration numbers in about 97.04-99.90% for Example 1, 96.90-99.78% for Example 2 and 97.03-99.90% for Example 3. In summary, it shows that both of these block iterative methods have lessened the computation time of the original GS method especially as the mesh size increases. Meanwhile, as for the maximum error, all methods show certain agreement.

From the numerical results stated in Tables 1 to 3, it can be observed that by applying four-point block iteration concept to the original GS point iterative method making the 4EGGS reduced its iteration numbers in about 71.84-80.14% for Example 1, 67.45-72.64% for Example 2 and 71.91-72.96% for Example 3. However, further application of the 4EGSOR block iteration has tremendously improved the reduction of the iteration numbers in about 97.04-99.90% for Example 1, 96.90-99.78% for Example 2 and 97.03-99.90% for Example 3. In summary, it shows that both of these block iterative methods have lessened the computation time of the original GS method especially as the mesh size increases. Meanwhile, as for the maximum error, all methods show certain agreement. A clear decrement percentage of all examples are as depicted in Tables 4 to 5.
5. Conclusion

In this paper, the formulations of 4EGSOR, 4EGGS and GS iterative methods were derived using the second-order Grünwald implicit finite difference schemes as shown in equations (7). Numerical results obtained for the three examples clearly shows that by applying 4EGGS iterative method has certainly reduced the number of iterations and execution time of the GS point iterative method, but further application of the 4EGSOR method could superiorly reduce the number of iterations and the execution time compared to all of the other iterative methods. In addition, the accuracy of 4EGSOR is comparatively the same with the other methods, even when the mesh size is increased. Therefore, it can be concluded that the developed four-Point EGDSOR is able to show substantial improvement in the number of iterations, compared to the other iterative methods. For future work, this study will be extended to investigate on the use of four-Point EDGSOR as a smoother to the proposed approximation equation.

Acknowledgment

The authors would like to express sincere gratitude to Universiti Malaysia Sabah for funding this research under UMSSGREAT research grant for postgraduate student: GUG0221-1/2018.

References

[1] Xu Q and Xu Y 2018 Commun. Nonlinear Sci. Numer. Simulat. 64, 135-148
[2] Edeki S, Ugbebor O and Owoloko E 2017 Int. J. Appl. Math. 47(4),
[3] Paradisi P 2015 Commun. Appl. Ind. Math. e-530 1-25
[4] Gao G and Sun Z 2011 J Comput Phys 230(3) 586-595
[5] Sweilam N H, Moharram H, Moniem N K and Ahmed S 2014 J. Numer. Math. 22(4) 363-82
[6] Azizi H and Loghmani G B 2013 Iran. J. Sci. Technol. 37, 23-28

Table 5. Decrement Percentage of the Number of Iterations (k) and the Execution Time (t) for 4EGGS and 4EGSOR iterative Method Compared to GS iterative Method for Example 2 at α=0.333, 0.666, 0.999

| n   | m   | method | α=0.333 | α=0.666 | α=0.999 |
|-----|-----|--------|---------|---------|---------|
|     |     | k (%)  | t (%)   | k (%)   | t (%)   |
| 128 | 4EGGS| 72.19  | 60.78   | 72.37   | 54.45   | 72.56   | 45.94   |
|     | 4EGSOR| 98.71  | 70.63   | 98.22   | 60.59   | 96.90   | 47.34   |
| 256 | 4EGGS| 71.67  | 71.11   | 71.98   | 65.94   | 72.64   | 55.73   |
|     | 4EGSOR| 99.24  | 86.46   | 98.98   | 78.57   | 98.31   | 63.54   |
| 512 | 4EGGS| 70.85  | 76.09   | 71.33   | 74.41   | 72.61   | 69.20   |
|     | 4EGSOR| 99.53  | 94.73   | 99.40   | 91.17   | 99.14   | 82.26   |
| 1024| 4EGGS| 69.48  | 77.25   | 70.75   | 78.12   | 72.48   | 76.77   |
|     | 4EGSOR| 99.70  | 97.94   | 99.59   | 96.80   | 99.53   | 93.61   |
| 2048| 4EGGS| 67.45  | 77.49   | 71.49   | 78.77   | 72.25   | 78.76   |
|     | 4EGSOR| 99.78  | 99.18   | 99.78   | 98.94   | 99.74   | 97.94   |

Table 6. Decrement Percentage of the Number of Iterations (k) and the Execution Time (t) for 4EGGS and 4EGSOR iterative Method Compared to GS iterative Method for Example 3 at α=0.333, 0.666, 0.999

| m   | method | α=0.333 | α=0.666 | α=0.999 |
|-----|--------|---------|---------|---------|
|     | k (%)  | t (%)   | k (%)   | t (%)   |
| 128 | 4EGGS  | 72.90   | 65.49   | 72.96   | 59.39   | 72.91   | 48.41   |
|     | 4EGSOR | 98.91   | 75.69   | 98.42   | 66.25   | 97.03   | 50.53   |
| 256 | 4EGGS  | 72.74   | 74.99   | 72.84   | 70.16   | 72.95   | 58.36   |
|     | 4EGSOR | 99.41   | 90.70   | 99.15   | 83.76   | 98.40   | 67.20   |
| 512 | 4EGGS  | 72.52   | 79.13   | 72.65   | 77.64   | 72.82   | 71.50   |
|     | 4EGSOR | 99.68   | 96.99   | 99.54   | 94.35   | 99.14   | 85.16   |
| 1024| 4EGGS  | 72.25   | 80.13   | 72.41   | 79.75   | 72.63   | 77.73   |
|     | 4EGSOR | 99.82   | 99.06   | 99.75   | 98.22   | 99.54   | 94.95   |
| 2048| 4EGGS  | 71.91   | 79.25   | 72.11   | 79.38   | 72.38   | 78.83   |
|     | 4EGSOR | 99.90   | 99.68   | 99.86   | 99.42   | 99.75   | 98.40   |
[7] Mishra T N and Rai K N 2015 *Int. J. Appl. Math. Res.* 4(1), 135
[8] Baharuddin, Sunarto A and Dalle J 2017 *J. Eng. Appl. Sci.* 12, 3220-3224
[9] Sunarto A, Sulaiman J and Saudi A 2016 *Global J. Pure Appl. Math.* 12(4), 3469-3479
[10] Sunarto A, Sulaiman J and Saudi A 2014 *Borneo Science* 34, 34-42
[11] Ghaffari R and Ghoreishi F 2019 *Applied Numerical Mathematics* 137, 62–79
[12] Wang Y M and Ren L 2019 *Appl. Math. Comput.* 342, 71-93
[13] Gunzburger M and Wang J 2019 *Int. J. Numer. Anal. Mod.* 16(2), 225-239
[14] Wu L, Zhao Y and Yang X 2018 *J. Appl. Math. Phys.* 6(5), 1017-1033
[15] Young D M 1971 *Iterative Solution of Large Linear Systems* (London: Academic Press)
[16] Young D M 1972 *J. Approx. Theory* 5(2), 137-148
[17] Evans D J 1985 *Int. J. Comput. Math.* 17 81-108
[18] Evans D J and Sahimi M S 1988 *Ann. Rev. Num. Fluid Mechanic and Heat Trans.* 2 283-389
[19] Evans D J and Yousif W S 1990 *Int. J. Comput. Math.* 34 71-78
[20] Abdullah A R 1991 *Int. J. Comput. Math.* 38 61-70
[21] Sulaiman J, Othman M and Hasan M K 2008 *Half-Sweep Algebraic Multigrid (HSAMG) method applied to diffusion equations, in Modeling, Simulation and Optimization of Complex Processes* (Berlin: Springer-Verlag)
[22] Rahimy M 2010 *Appl. Math. Sci.* 4(50) 2453-2461
[23] Podlubny I 1999 *Fractional Differential Equations* (San diego: Academic Press).
[24] Zahra W K and Elkholy S M 2013 *Electr. J. Math. Anal. Appl.* 1(2), 230-241
[25] Uddin M and Haq S 2011 *Commun. Nonlin. Sci. Numer. Simul.* 16(11) pp 4208-14
[26] Jiang Y and Ma J 2011 *J Comput. Appl. Math.* 235(11) pp 3285-90
[27] Ma Y 2014 *Appl. Math. Bioinformatics* 4(2) pp 125-45