Performance Numerical Method Half-Sweep Preconditioned Gauss-Seidel for Solving Fractional Diffusion Equation

Andang Sunarto1*, Jumat Sulaiman2

1 IAIN Bengkulu, Jln Raden Fatah, Pagar Dewa Kota Bengkulu, Indonesia
2 UMS Malaysia, Jl UMS Kota Kinabalu, Sabah, Malaysia

Corresponding Author Email: andang99@gmail.com

https://doi.org/10.18280/mmep.070205

Received: 17 January 2020
Accepted: 28 March 2020

Keywords:
HSPGS, space-fractional, Caputo’s, implicit finite difference

ABSTRACT

The main purpose, we derive a finite difference approximation equation from the discretization of the one-dimensional linear space-fractional diffusion equations by using the space fractional derivative of Caputo’s. The linear system will be generated by the Caputo’s finite difference approximation equation. The resulting linear system was then resolved using Half-Sweep Preconditioned Gauss-Seidel (HSPGS) iterative method, which compares its effectiveness with the existing Preconditioned Gauss-Seidel (PGS) or call named (Full-Sweep Preconditioned Gauss-Seidel (FSPGS)) and Gauss-Seidel (HSPGS) methods. Two examples of the issue are provided in order to check the performance efficacy of the proposed approach. The findings of this study show that the proposed iterative method is superior to FSPGS and GS.

1. INTRODUCTION

From The previous studies in [1-5] many successful mathematical models, which are based on fractional partial derivative equations (FPDEs), have been developed. Following to that, there are several methods used to solve these models. For instance, we have transform method [6], which is used to obtain analytical and/or numerical solutions of the fractional diffusion equations (FDE). Other than this method, other researchers have proposed finite difference methods such as explicit, implicit and fast method [7-9] Also it is pointed out that the explicit methods are conditionally stable. Therefore, we discretize the space-fractional diffusion equation via the implicit finite difference discretization scheme and Caputo’s fractional partial derivative of order $\beta$ in order to derive a Caputo’s implicit finite difference approximation equation.

This approximation equation leads a tridiagonal linear system. Due to the properties of the coefficient matrix of the linear system which is sparse and large scale, iterative methods are the alternative option for efficient solutions. Among the existing iterative methods, the preconditioned iterative methods [10-12] have been widely accepted to be one of the efficient methods for solving linear systems.

Because of the advantages of these iterative methods, the aim of this paper is to construct and investigate the performance effectiveness of the Half-Sweep Preconditioned Gauss-Seidel (HSPGS) iterative method for solving space-fractional parabolic partial differential equations (SPPDE’s) based on the Caputo’s implicit finite difference approximation equation. To investigate the effectiveness of the HSPGS method, we also implement the Gauss Seidel (GS) and FSPGS iterative methods being used a control method.

To performance the effectiveness of HSPGS method, let space-fractional parabolic partial differential equation (SPPDE’s) be defined as:

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

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

Then to develop the linear systems, some definitions that can be applied for fractional derivative theory need to developing the approximation equation of Eq. (1) in:

**Definition 1.** [13] The Riemann-Liouville fractional integral operator, $J_0^{\beta}$ of order - $\beta$ is defined as:

$$J_0^{\beta}f(x) = \frac{1}{\Gamma(\beta)}\int_0^x (x-t)^{\beta-1}f(t)dt, \beta > 0, x > 0 \quad (2)$$

**Definition 2.** [13] The Caputo’s fractional partial derivative operator, $D_0^\beta$ of order - $\beta$ is defined as:

$$D_0^\beta f(x) = \frac{1}{\Gamma(m-\beta)}\int_0^x \frac{f^{(m)}(t)}{(x-t)^{\beta+m-1}}dt, \beta > 0 \quad (3)$$

with $m-1 < \beta \leq m$, $m \in N$, $x > 0$.

We have the following properties when $m-1 < \beta \leq m$, $x > 0$: $D_0^\beta x^n = 0$, ($k$ is a constant).

$$D_0^\beta x^n = \begin{cases} 0, & \text{for } n \in N_0 \text{ and } n < [\beta] \\ \frac{1}{\Gamma(n+1+1-\beta)}x^{n-\beta}, & \text{for } n \in N_0 \text{ and } n \geq [\beta] \end{cases}$$
where, function $[\beta]$ denotes the smallest integer greater than or equal to $\beta$, $N_0=\{0,1,2,\ldots\}$ and $\Gamma(\cdot)$ is the gamma function.

2. CAPUTO APPROXIMATION DERIVATIVE

Assume that $h=\frac{\ell}{k}$, $k$ is positive integer and using second order approximation, we get

$$\frac{\partial^\beta U(x,t)}{\partial x^\beta} = \frac{1}{\Gamma(2 - \beta)} \int_0^{t - \beta} (t - s)^{2 - \beta} \frac{\partial^2 U(x,s)}{\partial x^2} ds$$

Let us define

$$\sigma_{\beta,h} = \frac{2h^{1-\beta}}{\Gamma(3-\beta)} \text{ and } g_{\beta} = \left(\frac{j}{2} + 1\right)^{2-\beta} - \frac{j}{2}$$

then the discrete approximation of Eq. (4).

$$\frac{\partial^\beta U(x,t)}{\partial x^\beta} = \sigma_{\beta,h} \sum_{j=0,2,k}^{j+1} g_{\beta}(U_{i,j-2,n} - 2U_{i,j,n} + U_{i,j+2,n})$$

Now we approximate Eq. (1) by using Caputo’s implicit finite difference approximation:

$$\alpha(U_{i,n} - U_{i,n-2}) = a_i \sigma_{\beta,h} \sum_{j=0,2,k}^{j+1} g_{\beta}(U_{i,j-2,n} - 2U_{i,j,n} + U_{i,j+2,n})$$

$$+ b_i \left(\frac{U_{i-2,n} - 2U_{i-1,n} + U_{i,1,n}}{4h}\right) + C_i U_{i+1,n} + f_{i,n}$$

for $i=2,4,\ldots,m-2$. Then we can simplify the scheme approximation equation as:

$$\alpha(U_{i,n} - U_{i,n-2}) = a_i \sigma_{\beta,h} \sum_{j=0,2,k}^{j+1} g_{\beta}(U_{i,j-2,n} - 2U_{i,j,n} + U_{i,j+2,n})$$

$$- \frac{b_i}{4h} \left(U_{i-2,n} - 2U_{i-1,n} + U_{i,1,n}\right) - C_i U_{i+1,n} + f_{i,n}$$

So, we get:

$$\therefore \quad b_i^* U_{i-2,n} + \left(\alpha - c_i^* \right) U_{i,n} - b_i^* U_{i+2,n}$$

$$- a_i^* \sum_{j=0,2,k}^{j+1} g_{\beta}(U_{i,j-2,n} - 2U_{i,j,n} + U_{i,j+2,n}) = f_{i,n}$$

where, $a_i^* = a_i \sigma_{\beta,h}$ , $b_i^* = \frac{b_i}{4h}$ , $c_i^* = c_i$ , $F_i^* = f_{i,n}$ and $f_i = \alpha(U_{i,n} - U_{i,n-2})$.

For simplicity, let Eq. (5) for $n > 3$ be rewritten as:

$$-R_i + a_i U_{i-4,n} + s_i U_{i-2,n} + p_i U_{i-1,n} + q_i U_{i,n} + r_i U_{i+1,n} = f_i$$

Then Eq. (6) can be used to construct a linear system in matrix form as:

$$A \mathbf{U} = \mathbf{f}$$

where,

$$A = \begin{bmatrix} q_2 & r_2 & p_4 & q_4 & r_4 & \cdots & \cdots & \cdots & \cdots \\ a_6 & s_6 & p_6 & q_6 & r_6 & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \\ a_m & s_m & p_m & q_m & r_m & \cdots & \cdots & \cdots & \cdots \\ \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} U_{i,2} & \cdots & U_{i,4} & \cdots & U_{i,m-1} \end{bmatrix}^T, \quad \mathbf{f} = \begin{bmatrix} f_{i,2} & \cdots & f_{i,4} & \cdots & f_{i,m-1} \end{bmatrix}^T$$

3. HSPGS METHODS

Before applying the HSPGS iterative method, we need to transform the original linear system (7) into the preconditioned linear system.

$$A^* \mathbf{x} = \mathbf{f}^*$$

where,

$$A^* = PAP^T, \quad \mathbf{f}^* = P \mathbf{f}, \quad \mathbf{U} = \mathbf{P}^T \mathbf{x}$$

Actually, the matrix $P$ is called a preconditioned matrix and defined as [14-16] $P = I + S$.

where,

$$S = \begin{bmatrix} 0 & -r_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -r_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -r_3 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & 0 & 0 & 0 & -r_{m-1} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the matrix $I$ is an identical matrix. To formulate HSPGS method, let the coefficient matrix $A^*$ in (7) be expressed as summation of the three matrices

$$A^* = D - L - V$$

where, $D$, $L$ and $V$ are diagonal, lower triangular and upper triangular matrices respectively. By using Eq. (9) and (11), the formulation of HSPGS iterative method can be defined.
generally as [11, 17, 18]:
\[
\mathbf{x}^{(k+1)} = (\mathbf{D} - \mathbf{L})^{-1} \mathbf{V} \mathbf{x}^{(k)} + (\mathbf{D} - \mathbf{L})^{-1} \mathbf{f}
\]
where, \( \mathbf{x}^{(k+1)} \) represents an unknown vector at \((k+1)\)th iteration.

The implementation of the HSPGS iterative method can be described in Algorithm 1.

Algorithm 1: HSPGS method

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

ii. For \( j = 1,2, \ldots, n \) implement

For \( i = 1,2, \ldots, m \) calculate

\[
\mathbf{x}^{(k+1)} = (\mathbf{D} - \mathbf{L})^{-1} \mathbf{V} \mathbf{x}^{(k)} + (\mathbf{D} - \mathbf{L})^{-1} \mathbf{f}
\]

\[
\mathbf{U}^{(k+1)} = \mathbf{P} \mathbf{T} \mathbf{x}^{(k+1)}
\]

Convergence test. If the convergence criterion i.e

\[
\left\| \mathbf{U}^{(k+1)} - \mathbf{U}^{(k)} \right\| < \varepsilon = 10^{-10}
\]

is satisfied, go to Step (iii). Otherwise go back to Step (ii).

iii Display approximate solutions.

4. FINDING NUMERICAL

We have examples of the SFPDE’s to verify the effectiveness of the HSPGS methods. In comparison, three criteria such as number iterations, the execution time (seconds) and maximum error at three different values of \( \beta = 1.2, \beta = 1.5 \) and \( \beta = 1.8 \). During the implementation of the point iterations, the convergence test considered the tolerance error, \( \varepsilon = 10^{-10} \).

Example 1 [19]:

Let us consider the following space-fractional initial boundary value problem

\[
\frac{\partial \mathbf{U} (x, t)}{\partial t} = \mathbf{d} (x) \frac{\partial^\beta \mathbf{U} (x, t)}{\partial x^\beta} + \mathbf{p} (x, t).
\]

Example 2 [19]:

Let us consider the following space-fractional initial boundary value problem

\[
\frac{\partial \mathbf{U} (x, t)}{\partial t} = \Gamma (1.2) \mathbf{x}^{\beta} \frac{\partial^\beta \mathbf{U} (x, t)}{\partial x^\beta} + 3 \mathbf{x}^2 (2x - 1) e^4.
\]

All numerical results for Eqs. (11) and (12), obtained from application of GS, FSPGS and HSPGS iterative methods are recorded in Table 1 and 2 by using the different value of mesh size, \( M = 128, 256, 512, 1024 \) and 2048.

| Table 1. Comparison between number of iterations (K), the execution time (seconds) and maximum errors for the iterative methods using example at \( \beta = 1.2, 1.5, 1.8 \) |
|-----------------|-----------------|-----------------|-----------------|-----------------|
| M   | Method | K | Time | Max | K | Time | Max | K | Time | Max |
| 128 | FSPGS  | 36 | 1.09 | 2.37e-02 | 104 | 1.36 | 2.60e-04 | 345 | 2.09 | 3.99e-02 |
|     | HSPGS  | 19 | 0.26 | 2.37e-02 | 42 | 1.24 | 6.20e-04 | 108 | 3.25 | 4.60e-02 |
| 256 | FSPGS  | 72 | 2.73 | 2.44e-02 | 272 | 2.70 | 5.69e-04 | 1123 | 111.98 | 3.97e-02 |
|     | HSPGS  | 36 | 2.50 | 2.44e-02 | 104 | 11.33 | 5.69e-04 | 345 | 4.05 | 4.59e-02 |
| 512 | FSPGS  | 151 | 58.11 | 2.47e-02 | 723 | 276.20 | 5.36e-04 | 3659 | 1398.43 | 3.96e-02 |
|     | HSPGS  | 72 | 23.35 | 2.47e-02 | 272 | 124.86 | 5.36e-04 | 1123 | 478.23 | 4.55e-02 |
| 1024| FSPGS  | 328 | 492.56 | 2.49e-02 | 1935 | 945.20 | 5.13e-04 | 11836 | 2138.11 | 3.95e-02 |
|     | HSPGS  | 151 | 193.63 | 2.49e-02 | 724 | 473.13 | 5.13e-04 | 3657 | 1054.31 | 4.53e-02 |
| 2048| FSPGS  | 1547 | 1227.21 | 2.50e-02 | 8320 | 4348.68 | 5.02e-04 | 47322 | 8979.18 | 3.93e-02 |
|     | HSPGS  | 327 | 472.53 | 2.50e-02 | 1938 | 3120.96 | 5.02e-04 | 22152 | 4335.75 | 4.51e-02 |

| Table 2. Comparison between number of iterations (K), the execution time (seconds) and maximum errors for the iterative methods using example at \( \beta = 1.2, 1.5, 1.8 \) |
|-----------------|-----------------|-----------------|-----------------|-----------------|
| M   | Method | K | Time | Max | K | Time | Max | K | Time | Max |
| 128 | FSPGS  | 27 | 0.72 | 1.80e-01 | 75 | 1.83 | 5.44e-02 | 213 | 5.27 | 8.88e-04 |
|     | HSPGS  | 15 | 0.25 | 1.80e-01 | 30 | 0.54 | 5.44e-02 | 67 | 2.94 | 8.88e-04 |
| 256 | FSPGS  | 55 | 4.72 | 1.84e-01 | 197 | 17.11 | 5.58e-02 | 686 | 59.48 | 4.09e-04 |
|     | HSPGS  | 27 | 1.38 | 1.84e-01 | 75 | 7.83 | 5.58e-02 | 213 | 20.45 | 4.09e-04 |
| 512 | FSPGS  | 116 | 37.86 | 1.86e-01 | 522 | 170.92 | 5.65e-02 | 2213 | 737.50 | 1.54e-04 |
|     | HSPGS  | 55 | 10.51 | 1.86e-01 | 197 | 77.58 | 5.65e-02 | 686 | 331.95 | 1.54e-04 |
| 1024| FSPGS  | 250 | 322.55 | 1.89e-01 | 1435 | 443.81 | 5.69e-02 | 3452 | 820.62 | 1.49e-04 |
|     | HSPGS  | 116 | 147.81 | 1.89e-01 | 522 | 299.59 | 5.69e-02 | 1224 | 411.91 | 1.49e-04 |
| 2048| FSPGS  | 518 | 413.21 | 1.88e-01 | 4125 | 713.64 | 5.85e-02 | 5127 | 3173.73 | 1.20e-04 |
|     | HSPGS  | 251 | 207.81 | 1.88e-01 | 1437 | 311.27 | 5.85e-02 | 2253 | 1062.72 | 1.20e-04 |
5. DISCUSSION AND CONCLUSION

In order to get the numerical solution of the space-fractional diffusion problems, the paper presents the derivation of the Caputo’s implicit finite difference approximation equations in which this approximation equation leads to a linear system. From observation of all experimental results by imposing the GS, FSPGS and HSPGS iterative methods, it is obvious at $\beta = 1.2$ that number of iterations have declined approximately by 41.30–82.45% corresponds to the HSPGS iterative method compared with the GS and FSPGS method. Again, in terms of execution time, implementations of HSPGS method are much faster about 51.18–92.43% than the GS and FSPGS method. It means that the HSPGS method requires the least amount for number of iterations and computational time at $\beta = 1.2$ as compared with GS and FSPGS iterative methods. Based on the accuracy of both iterative methods, it can be concluded that their performance numerical solutions are in good agreement.

REFERENCES


