Hybrid Hopscotch Crank-Nicholson-Du Fort and Frankel (HP-CN-DF) Method for Solving Two Dimensional System of Burgers’ Equation

S. Maritim¹, J. K. Rotich² and J. K. Bitok³

¹ University of Kabianga, Mathematics & Computer Science Department
P.O Box 2030-20200, Kericho, Kenya

² University of Kabianga, Mathematics & Computer Science Department
P.O Box 2030-20200, Kericho, Kenya
https://orcid.org/0000-0002-7275-2944

³ University of Eldoret, Mathematics & Computer Science Department
P.O Box 1125-30100, Eldoret, Kenya

Abstract
The hopscotch scheme is a general method for the solution of second order parabolic and elliptic partial differential equations. It has been shown to be a Peaceman-Rachford alternating direction implicit (ADI) process, Peaceman and Rachford [16]. The method lies somewhere between explicit and implicit. It is fast and accurate in solving partial differential equations although it has seen only limited use by researchers in science and engineering discipline. The method has been used to solve the incompressible Navier-Stokes equations. It is suitable for many fluid flow problems because it is simple to program, faster and gives consistent results in computation per iteration and storage and adaptable to vector or parallel computer architecture. In this research hybrid hopscotch method for solving two-dimensional system of Burgers’ equations is developed. A solution algorithm for two-dimensional partial differential equation with mixed boundary condition is first derived. Finally hybrid finite schemes for solving Burgers equations are developed. Hybrid Hopscotch-Crank-Nicholson-Du Fort and Frankel Scheme (HP-CN-DF) proved to be stable because the absolute errors did not change irregularly with a small change in the input data. The schemes were
also consistent because the results were not changing suddenly for small change in
time and space hence convergent in line with Lax equivalence theorem.

Keywords: Hopscotch, Crank-Nicholson, Du Fort and Frankel, Burgers equation

1.1 Background Information

The hopscotch method is a type of finite-difference method for multi-variable
partial differential equations in which implicit solutions are obtained using two
sweeps in different directions.
The Burgers’ equation is considered as the fundamental partial differential
equation in the field of applied mathematics such as fluid mechanics, nonlinear
acoustics, gas dynamics, and traffic flow among others. This equation was first
formulated by Bateman, [1] and can be regarded as a qualitative approximation of
the Navier-Stokes equations. The equation incorporates both convection and
diffusion, but preserves the hybrid characteristic of the Navier-Stokes equations,
and can be solved using similar numerical methods. It provides a good model for
the numerical solution of the complicated Navier-Stokes equations. It is named
after Johannes Martinus Burgers (1895-1981), Burgers [3].

Theorem 3.1 Lax equivalence theorem
States that for a well posed initial value problem, a finite difference
approximation to it is convergent if and only if it is consistent and stable. The
proof of this theorem is given in Strikwerda [18].

1.2 Two Dimensional System of Burgers Equation

The 2-D Burgers’ equation, which was also solved by Kweyu [14], is of the
form:
\[
\begin{align*}
  u_t &= -uu_x - vu_y + \frac{1}{Re} (u_{xx} + u_{yy}) \\
  v_t &= -uv_x - vv_y + \frac{1}{Re} (v_{xx} + v_{yy})
\end{align*}
\]  

(1.1)

In solving this equation, it is usually subjected respectively to initial and boundary
conditions:
\[
\begin{align*}
  u(x, y, 0) &= f(x, y), \quad (x, y) \in D \\
  v(x, y, 0) &= g(x, y), \quad (x, y) \in D
\end{align*}
\]  

(1.2)

And
\[
\begin{align*}
  u(x, y, t) &= f_1(x, y), \quad x, y \in \partial D, t > 0 \\
  v(x, y, t) &= g_1(x, y), \quad x, y \in \partial D, t > 0
\end{align*}
\]  

(1.3)
where \( D = \{ (x, y) | a \leq x \leq b, a \leq y \leq b \} \) and \( \partial D \) is its boundary while \( u(x, y, t) \) and \( v(x, y, t) \) are the velocity components to be determined, \( f, g, f_1 \) and \( g_1 \) are known functions.

1.3 Hopscotch Method

Gordon [7] introduced the idea of using explicit and implicit finite difference schemes and alternate mesh points to solve heat equation and further applied the technique to solve physical problems. Gourlay [10] developed Gordon’s idea in a general way for two space dimensional parabolic and elliptic problems and showed that the hopscotch method was infact a Peaceman-Rachford (Peaceman and Rachford, [16]) method with the coefficient matrix split in a rather novel way. Gourlay and McGuire [8] proposed a general class of algorithms for numerical solution of partial differential equations. The structure and properties of the algorithms make them particularly easy to implement. The research included a numerical comparisons of the algorithms for the hopscotch and Peaceman Rachford method. The research revealed that the Peaceman-Rachford method or ADI hopscotch gives high accuracy at the expense of development and running time. The line hopscotch was found to be competitive with the Peaceman-Rachford method in accuracy and is more efficient.

Gourlay and McKee [9] discussed hopscotch as a fast finite difference technique used to solve parabolic and elliptic equations in two space dimensions with a mixed derivative. The method is compared numerically with existing alternating direction implicit (A.D.I.) and locally one dimensional (L.O.D.) methods for simple problems. Douglas and Gunn's A.D.I. method is both simplified and improved by reformulating it as hopscotch method. According to the research handling of the mixed derivative caused considerable complication when using a true A.D.I. or L.O.D. approach. Its treatment by hopscotch is much less cumbersome and the resulting algorithms straightforward.

Gane and Gourlay considered a general block implicit hopscotch procedure for solving parabolic partial differential equations. Stability and convergence analyses were given and an examination of the comparative accuracy of various particular cases was given. The usefulness of these algorithms in solving problems using \((r\theta)\) geometry was indicated and their extensions to general exterior problems using a peripheral approach was included.

Evans and Danaee [4] established a new block method to solve the parabolic and elliptic partial differential equations and a comparison with the line hopscotch method was considered and proved to be useful.

Jan and Gerard [12] considered the ODE (ordinary differential equation) that arises from a semi-discretization (discretization of the spatial coordinates) of a first order system form of a fourth order parabolic PDE (partial differential equation). Analysis of the stability of the finite difference methods for this fourth order parabolic PDE that arise if one applies the hopscotch idea to this ODE. The error propagation of these methods was represented by a three terms matrix-vector recursion in which the matrices have a certain anti-hermitian structure. A uniform
expression for the stability bound (or error propagation bound) of this recursion in terms of the norms of the matrices was found. The result yielded conditions under which these methods were strongly asymptotically stable (meaning the stability is uniform both with respect to the spatial and the time stepsizes (tending to 0) and the time level (tending to infinity)), also in case the PDE has (spatial) variable coefficients. These implies convergence according to the convergence theorem.

Thije [19] explained that the odd-even hopscotch (OEH) scheme is a time-integration technique for time-dependent partial differential equations. The research applied the OEH scheme to the incompressible Navier–Stokes equations in conservative form. In order to decouple the computation of the velocity and the pressure, the OEH scheme was applied in combination with the pressure correction technique. The resulting scheme is referred to as the odd-even hopscotch pressure correction (OEH-PC) scheme. This scheme requires per time step the solution of a Poisson equation for the computation of the pressure. For space discretization standard central differences was used. The OEH-PC scheme was applied to the Navier–Stokes equations for the computation of an exact solution, with the purpose of testing the (order of) accuracy of the scheme in time as well as in space. The OEH-PC scheme was applied for the computation of a model problem. Finally, a comparison between two Poisson solvers for the computation of the pressure is presented.

Goede and Thije [6] proposed vectorized version of the odd–even hopscotch (OEH) scheme and the alternation direction implicit (ADI) scheme and implemented on vector computers for solving the two-dimensional Burgers equations on a rectangular domain. The research examines the efficiency of both schemes on vector computers. Data structures and techniques employed in vectorizing both schemes are discussed, accompanied by performance details.

Verwer and Sommeijer [20] proposed a linear stability analysis for an odd-even-line hopscotch (OELH) method, which was developed for integrating three-space dimensional, shallow water transport problems. Sufficient and necessary conditions were derived for strict von Neumann stability for the case of the general, constant coefficient, linear advection-diffusion model problem. The analysis was based on an equivalence with an associated scheme which is composed of the leapfrog, the Du Fort-Frankel, and the Crank-Nicolson schemes. The results appeared to be rather intricate. For example, the resulting expressions for critical stepsizes reveal that the presence of horizontal diffusion generally leads to a smaller value, in spite of the fact that there is unconditional stability for pure diffusion problems. It is pointed out that this is due to the Du Fort--Frankel deficiency. On the other hand, it is also shown, by a numerical experiment, that in practice it is sufficient to obey the weaker Courant-Friedrichs--Lewy (CFL) condition associated with the case of pure horizontal advection, unless a huge number of integration steps are to be taken.

Harley [11] obtained numerical solutions to the Frank-Kamenetskii partial differential equation modelling a thermal explosion in a cylindrical vessel using the hopscotch scheme. The reseracher observed that a nonlinear source term in the equation leads to numerical difficulty and hence adjust the scheme to accommo-
date such a term. Numerical solutions obtained via MATLAB, MATHEMATICA and the Crank–Nicolson implicit scheme were employed as a means of comparison. To gain insight into the accuracy of the hopscotch scheme the solution was compared to a power series solution obtained via the Lie group method. The numerical solution was also observed to converge to a well-known steady state solution. A linear stability analysis was performed to validate the stability of the results obtained.

Jan [13] used the hopscotch and the Crank-Nicolson methods to solve European option prices and analyzed the pricing results from these two methods by comparing to the pricing result generates from the Black-Scholes model. The accomplishment of the work is based on MATLAB applications. In the report, the author started with an introduction of the numerical approximation of derivatives and applied them to solve the Black-Scholes Partial differential equation. The basic way of solving PDE are use of explicit scheme and implicit scheme, there are also two methods called hopscotch method and Crank-Nicolson method which mixed the explicit and implicit scheme to enhance the accuracy of the result they approximate. By comparing these applications, it was found that it is easy to apply the explicit scheme to solve the Black-Scholes PDE. However, the numerical instability of explicit scheme was a concern when using it. The hopscotch method and Crank-Nicolson method integrated the advantage of fully implicit and explicit schemes. The two methods ensure a fairly accurate result for the users, but by comparing the CPU time it was found that the Crank-Nicolson method can save more computational time than hopscotch method even though one has to solve a set of equation simultaneously.

Rotich et al. [17] developed the pure Crank-Nicholson (CN) Scheme and Crank-Nicholson-Lax-Fredrichs’ (CN-LF) method by Operator Splitting. Crank-Nicholson-Du-Fort and Frankel being a hybrid scheme made by combining the Crank-Nicholson and Lax-Fredrich schemes. Lax-Friedrichs’ scheme is conditionally stable and an explicit scheme. The developed schemes were then solved numerically using initially solved solution via Hopf-Cole transformation and separation of variables to generate the initial and boundary conditions. Analysis of the resulting schemes was found to be unconditionally stable. And that the results of the hybrid scheme were found to compare well with those of the pure Crank-Nicholson.

Brajesh and Pramod [2] proposed a new algorithm called modified trigonometric cubic B-spline differential quadrature method for numerical computation of the time dependent partial differential equations. The numerical computation of the Burgers equation was obtained by using modified trigonometric cubic B-spline differential quadrature method with time integration algorithm. The modified trigonometric cubic B-spline differential quadrature method proved to produce better results than the results due to almost all the existing schemes. The modified trigonometric cubic B-spline differential quadrature method was shown to be conditionally stable using the matrix stability analysis method for various grid points.
1.4 Approximation at the Boundaries

The solution of Burgers system of equations (1.1) at any point \((x, y, t)\) is given by the following equations, according to Rotich \textit{et. al.} [17]:

\[
u(x, y, t) = \frac{-2y - 2\pi e^{-\frac{2\pi t}{Re}} ((\cos\pi x - \sin \pi x) \sin \pi y)}{Re(100 + xy + e^{-\frac{2\pi t}{Re}} ((\cos\pi x - \sin \pi x) \sin \pi y)}
\]

(1.4)

\[
u(x, y, t) = \frac{-2x - 2\pi e^{-\frac{2\pi t}{Re}} ((\cos\pi x - \sin \pi x) \sin \pi y)}{Re(100 + xy + e^{-\frac{2\pi t}{Re}} ((\cos\pi x - \sin \pi x) \sin \pi y)}
\]

(1.5)

Using \(\Delta x = \Delta y = h\) and \(\frac{\Delta t}{h^2} = \alpha\) and \(\frac{\Delta t}{h^2Re} = \beta\) to obtain the hopscotch Crank-Nicholson scheme as shown below

\[
(U_{i,j}^{n+1} - U_{i,j}^n) = -\alpha U_{i-1,j-1}^n (U_{i+1,j}^{n+1} - 2U_{i,j}^{n+1} + U_{i-1,j}^{n+1}) - \alpha V_{i-1,j-1}^n (U_{i,j+1}^{n+2} - 2U_{i,j}^{n+2} + U_{i,j-1}^{n+2})
+ \beta (-3U_{i-2,j}^{n+1} + 6U_{i-1,j}^{n+1} - 4U_{i,j}^{n+1} + U_{i+1,j}^{n+1})
+ \beta (-3U_{i,j-2}^{n+2} + 6U_{i,j-1}^{n+2} - 4U_{i,j}^{n+2} + U_{i,j+1}^{n+2})
\]

(1.6)

\[
(V_{i,j}^{n+1} - V_{i,j}^n) =
-\alpha U_{i-1,j-1}^n (V_{i+1,j}^{n+1} - 2V_{i,j}^{n+1} + V_{i-1,j}^{n+1}) - \alpha V_{i-1,j-1}^n (V_{i,j+1}^{n+2} - 2V_{i,j}^{n+2} + V_{i,j-1}^{n+2})
+ \beta (-3V_{i-2,j}^{n+1} + 6V_{i-1,j}^{n+1} - 4V_{i,j}^{n+1} + V_{i+1,j}^{n+1})
+ \beta (-3V_{i,j-2}^{n+2} + 6V_{i,j-1}^{n+2} - 4V_{i,j}^{n+2} + V_{i,j+1}^{n+2})
\]

(1.7)

\[
3\beta U_{i-2,j}^{n+1} + U_{i-1,j}^{n+1} (\alpha U_{i-1,j-1}^n - 6\beta) + U_{i,j}^{n+1} (1 - 2\alpha U_{i-1,j-1}^n + 4\beta)
+ U_{i+1,j}^{n+1} (\alpha U_{i-1,j-1}^n - \beta) + 3\beta U_{i,j+2}^{n+2} + U_{i,j-2}^{n+2} (\alpha V_{i-1,j-1}^n - 6\beta)
- U_{i,j+2}^{n+2} (2\alpha V_{i-1,j-1}^n - 4\beta) + U_{i,j-2}^{n+2} (\alpha V_{i-1,j-1}^n - \beta) = U_{i,j}^{n+1}
\]

(1.8)

\[
3\beta V_{i-2,j}^{n+1} + V_{i-1,j}^{n+1} (\alpha U_{i-1,j-1}^n - 6\beta) + V_{i,j}^{n+1} (1 - 2\alpha U_{i-1,j-1}^n + 4\beta)
+ V_{i+1,j}^{n+1} (\alpha U_{i-1,j-1}^n - \beta) + 3\beta V_{i,j+2}^{n+2} + V_{i,j-2}^{n+2} (\alpha V_{i-1,j-1}^n - 6\beta)
- V_{i,j+2}^{n+2} (2\alpha V_{i-1,j-1}^n - 4\beta) + V_{i,j-2}^{n+2} (\alpha V_{i-1,j-1}^n - \beta) = V_{i,j}^{n+1}
\]

(1.9)
1.5 Development of Hybrid Schemes

1.5.1 Hybrid hopscotch Crank-Nicholson-Du Fort and Frankel (HP-CN-DF) Scheme

The Hybrid hopscotch Crank-Nicholson-Du Fort and Frankel Scheme is obtained by replacing \( U_{i,j}^n \) and \( V_{i,j}^n \) by \( \frac{1}{2} (U_{i,j}^{n+1} + U_{i,j}^{n-1}) \) and \( \frac{1}{2} (V_{i,j}^{n+1} + V_{i,j}^{n-1}) \) in equations (1.8) and (1.9) respectively to give the HP-CN-DF scheme below:

\[
3\beta U_{i-2,j}^{n+1} + U_{i-1,j}^{n+1} (\alpha U_{i-1,j-1}^n - 6\beta) + U_{i,j}^{n+1} (1 - 2\alpha U_{i-1,j-1}^n + 4\beta - \frac{1}{2}) + U_{i+1,j-1}^{n+1} (\alpha U_{i-1,j-1}^n - 6\beta) - U_{i,j-1}^{n+1} (2\alpha V_{i-1,j-1}^n - 4\beta) + U_{i,j+1}^{n+2} (\alpha V_{i-1,j-1}^n - 6\beta) = \frac{1}{2} U_{i,j}^{n-1}
\]

\[
3\beta V_{i-2,j}^{n+1} + V_{i-1,j}^{n+1} (\alpha U_{i-1,j}^n - 6\beta) + V_{i,j}^{n+1} (1 - 2\alpha U_{i-1,j-1}^n + 4\beta - \frac{1}{2}) + V_{i+1,j-1}^{n+1} (\alpha U_{i-1,j-1}^n - 6\beta) - V_{i,j-1}^{n+1} (2\alpha V_{i-1,j-1}^n - 4\beta) + V_{i,j+1}^{n+2} (\alpha V_{i-1,j-1}^n - 6\beta) = \frac{1}{2} V_{i,j}^{n-1}
\]

1.6 Presentation of Numerical Results

The following data is used: \( k = 0.001, h = 0.1, l = 0.1 \) and \( \text{Re} = 4000 \) to obtain the results.

Here the computational domain is taken as a square domain \( R = \{ (x, y): 0 \leq x \leq 1, 0 \leq y \leq 1 \} \). The initial and boundary conditions for \( u(x, y, t) \) and \( v(x, y, t) \) are taken from the numerical solutions by Kweyu et al. [15]. The numerical computations are performed using uniform grid, with a mesh width \( \Delta x = \Delta y = 0.1 \). For the figures in 2-D presented below the values of the absolute errors at the corresponding points along the \( x \)-axis and \( y \)-axis are the same. Graphs of \( u(x, y, t) \) and \( v(x, y, t) \) against \( x \) is presented and is exactly the same as that of \( u(x, y, t) \) and \( v(x, y, t) \) against \( y \).

The schemes developed formed a series of equations that could be expressed in matrix form and therefore MATLAB software was required to generate the numerical results. The results are presented in graphical forms, tables as well as three-dimensional figures.

1.6.1 Two Dimensional plots of Absolute Errors in Solutions of \( u(x, y, t) \) and \( v(x, y, t) \)

Figures 1 and 2 show the absolute error in solutions of \( u(x, y, t) \) and \( v(x, y, t) \) respectively plotted as a function of position along the \( x \)-axis for the four hybrid schemes used, fixing \( t = 1.0 \).
Figure 1: Absolute error in Solution of $u$ for the 2-D Coupled Burgers’ equation

Figure 2: Absolute error in Solution of $v$ for the 2-D Coupled Burgers’ equation

The errors are the variation of the HP-CN, and HP-CN-DF when compared with values generated by the solution proposed by Kweyu et. al. [15]. The figure clearly shows that the hybrid HP-CN-DF has the least error than HP-CN. This shows that HP-CN-DF is the most accurate. This figure shows that the schemes developed are stable.
1.6.2 Three Dimensional plots of Solutions of $u(x, y, t)$ and $v(x, y, t)$

1.6.2.1 HP-CN
Figure 3 and 4 shows a 3-D plot of solutions of $u(x, y, t)$ and $v(x, y, t)$ when $t = 1.000$ for HP-CN plotted against changing values of x and y variables.

Figure 3: HP-CN Numerical Solution of $u$ at $t=1.000$

Figure 4: HP-CN Numerical Solution of $v$ at $t=1.000$

1.6.2.2 HP-CN-DF
Figure 5 and 6 shows a 3-D plot of solutions of $u(x, y, t)$ and $v(x, y, t)$ when $t = 1.000$ for HP-CN-DF plotted against changing values of x and y variables.
Figure 5: HP-CN-DF Numerical Solution of u at t=1.000

Figure 6: HP-CN-DF Numerical Solution of v at t=1.000

The figures 3, 4, 5 and 6 are 3-D images of solutions $u(x,y,t)$ or $v(x,y,t)$ plotted against $x$ and against $y$ simultaneously using MATLAB for the various hybrid schemes developed. All the figures clearly show that the solutions are not changing suddenly and in an irregular way hence the results for all the developed schemes are consistent.
Table 1: Table of Solution of \( u \) for the 2-D Burgers equation at \( t = 1.0 \) for the different schemes \( \text{Re}=4000 \)

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
<th>Kweyu et al. [15] proposed solution (*e)-006</th>
<th>HP-CN (*e)-006</th>
<th>HP-CN-DF (*e)-006</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>-0.45206816295610</td>
<td>-0.45228096838606</td>
<td>-0.45207290756473</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>-0.90666175924746</td>
<td>-0.90707726837174</td>
<td>-0.90667102275640</td>
</tr>
<tr>
<td>0.3</td>
<td>0.3</td>
<td>-1.36301183011581</td>
<td>-1.36362293690697</td>
<td>-1.36302545536009</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>-1.81993778965255</td>
<td>-1.82074210344674</td>
<td>-1.81995572275460</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>-2.27602700265147</td>
<td>-2.27702783812104</td>
<td>-2.27604931747869</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>-2.72986139439426</td>
<td>-2.73106779914542</td>
<td>-2.72988829257710</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>-3.18025559498377</td>
<td>-3.18168135576848</td>
<td>-3.18028738376313</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>-3.62646733093825</td>
<td>-3.62812910745055</td>
<td>-3.62650439036687</td>
</tr>
<tr>
<td>0.9</td>
<td>0.9</td>
<td>-4.06834558851241</td>
<td>-4.07026042802845</td>
<td>-4.06838828086325</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-4.50639373341364</td>
<td>-4.50857649857835</td>
<td>-4.50644239865479</td>
</tr>
</tbody>
</table>

Table 2: Table of Solution of \( v \) for the 2-D Burgers equation at \( t = 1.0 \) for the different schemes \( \text{Re}=4000 \)

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
<th>Kweyu et al. [15] proposed solution (*e)-006</th>
<th>HP-CN (*e)-006</th>
<th>HP-CN-DF (*e)-006</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>-4.96596485014788</td>
<td>-4.96611591898017</td>
<td>-4.96596821457264</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>-4.93021949923746</td>
<td>-4.93053010322735</td>
<td>-4.93022641672314</td>
</tr>
<tr>
<td>0.3</td>
<td>0.3</td>
<td>-4.89143745710773</td>
<td>-4.89192178335778</td>
<td>-4.89144824368021</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>-4.84860678586601</td>
<td>-4.84928319580011</td>
<td>-4.84862185048505</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>-4.80113941328106</td>
<td>-4.80202839659612</td>
<td>-4.80115921255686</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>-4.74893329120019</td>
<td>-4.75005517398268</td>
<td>-4.74895877138262</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>-4.69237703046003</td>
<td>-4.69374975282673</td>
<td>-4.69240760276388</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>-4.63229606978619</td>
<td>-4.63393334321813</td>
<td>-4.63233253363041</td>
</tr>
<tr>
<td>0.9</td>
<td>0.9</td>
<td>-4.56984942066285</td>
<td>-4.57175950062964</td>
<td>-4.56989195968453</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-4.50639373341364</td>
<td>-4.50857892118299</td>
<td>-4.50644239865479</td>
</tr>
</tbody>
</table>
Table 3: Table of Absolute Error in \( u \) for the 2-D Burgers equation at \( t = 1.0 \) for the different schemes \( \text{Re}=4000 \)

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>HP-CN (*e-006)</th>
<th>Hybrid HP-CN-DF (*e-006)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>0.000415509124277014</td>
<td>0.0000092641064756</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>0.000611106791159832</td>
<td>0.000136252442800</td>
</tr>
<tr>
<td>0.3</td>
<td>0.3</td>
<td>0.000804313794189992</td>
<td>0.000179331020500</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>0.001000835469569860</td>
<td>0.000223148272200</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>0.001206404751159960</td>
<td>0.000268981828400</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>0.001425760784710130</td>
<td>0.000317887793600</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>0.001661767512300290</td>
<td>0.000370504286200</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>0.001914839516039990</td>
<td>0.000426923508390</td>
</tr>
<tr>
<td>0.9</td>
<td>0.9</td>
<td>0.002182765164709720</td>
<td>0.000486652411500</td>
</tr>
</tbody>
</table>

Table 4: Table of Absolute Error in \( v \) for the 2-D Burgers equation at \( t = 1.0 \) for the different schemes \( \text{Re}=4000 \)

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>HP-CN (*e-006)</th>
<th>Hybrid HP-CN-DF (*e-006)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>0.00031060398989913</td>
<td>0.00006917485679</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>0.000484326250050238</td>
<td>0.00010786572480</td>
</tr>
<tr>
<td>0.3</td>
<td>0.3</td>
<td>0.000676409934100342</td>
<td>0.000150646190400</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>0.000888983315060266</td>
<td>0.00019798975800</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>0.001121882782490150</td>
<td>0.000249859380700</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>0.001372722366700610</td>
<td>0.000305723038500</td>
</tr>
<tr>
<td>0.7</td>
<td>0.7</td>
<td>0.001637273431939870</td>
<td>0.000364638442200</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>0.001910079966790160</td>
<td>0.000425390216800</td>
</tr>
<tr>
<td>0.9</td>
<td>0.9</td>
<td>0.002185187769350170</td>
<td>0.000486652411500</td>
</tr>
</tbody>
</table>

1.7 Conclusion

The hybrid schemes Hopscotch-Crank-Nicholson - Du - Fort and Frankel (HP-CN-DF) was developed and used to solve 2D Burgers’ equation. The developed scheme proved to be stable because the errors did not ‘blowup’ (absolute errors less than 0.009% as shown on the table of values of absolute errors with a small change in the input data. The schemes are also consistent because the results were not changing suddenly for small change in time and space hence convergent in line with Lax equivalence theorem.
Hybrid hopscotch Crank-Nicholson-Du Fort and Frankel method

References


Received: July 19, 2018; Published: August 9, 2018