An Explicit Predictor-Corrector Solver with Applications to Burgers' Equation

Suhrit K. Dey and Charlie Dey

September 1983
An Explicit Predictor-Corrector Solver with Applications to Burgers’ Equation

Suhrit K. Dey, Ames Research Center, Moffett Field, California
Charlie Dey, Country Lane School, San Jose, California
AN EXPLICIT PREDICTOR-CORRECTOR SOLVER WITH APPLICATION

TO BURGERS' EQUATION

Suhrit K. Dey and Charlie Dey*

Ames Research Center

SUMMARY

Forward Euler's explicit, finite-difference formula of extrapolation is used as a predictor and a convex formula as a corrector to integrate differential equations numerically. An application has been made to Burgers' equation.

INTRODUCTION

The single-step, explicit, forward Euler scheme is possibly the simplest algorithm with which to solve initial-value problems. However, it has very restricted stability properties which require small step sizes so that this formula of extrapolation may be useful. Implicit schemes, in which large step sizes may be used, have better stability properties. But iterative methods are generally used for computational solution. Most predictor-corrector algorithms require multistep operations in which correctors use iterative methods for convergence.

In this work, a filtering formula has been introduced by the second author so that extrapolated values obtained by an explicit formula may be appropriately corrected using one iteration which requires no computation of derivatives or inversion of a matrix. Mathematically, the corrector introduces a convex mapping, and as such it may be called a convex corrector. The explicit formula of extrapolation used in this work is the forward Euler difference scheme. Linearized stability analysis by Lomax (private communication) showed that the method is stable for step sizes that could be much larger than those required for stability of the forward Euler scheme. This will be discussed further on. With regard to the solution of nonlinear models, the limitations of this scheme are still being investigated.

Let us first develop the algorithm, discuss its stability properties, and look into some of its simple applications.

THE ALGORITHM

Let us consider an initial-value model:

\[ \frac{du}{dt} = f(u,t) \]

\[ u(t_0) = u_0 \]
The forward Euler scheme is:

\[ U_{n+1} = U_n + \Delta t \cdot f(U_n, t_n) \]  \hspace{1cm} (2)

where \( \Delta t = \) time-step  

\[ U_n = U(t_n) = \text{the net function corresponding to } u(t_n) \]

We may now set up a predictor-corrector formula as follows:

\[ \hat{U} = U_n + \Delta t \cdot f(U_n, t_n) \quad \text{(predictor)} \]

\[ U_{n+1} = (1 - \gamma)\hat{U} + \gamma[U_n + \Delta t \cdot f(\hat{U}, t_{n+1})] \quad \text{(corrector)} \]  \hspace{1cm} (4)

Gamma is herein called a filtering parameter. Here we have restricted \( \gamma \) to  

\[ 0 < \gamma < 1 \]  \hspace{1cm} (5)

Thus, \( \gamma \) is essentially a convex parameter. If \( \gamma = 0 \), \( U_{n+1} = \hat{U} \). The corrector is ineffective.

If the model (eq. (1)) is linear, then for a given step size it may be possible to find a \( \gamma \) such that the method could be effective. If the model is nonlinear, it may be locally linearized to obtain a \( \gamma \) (for a given step size) so that the method may remain effective. This stability analysis, done by Lomax (private communications), is our next topic of discussion.

STABILITY ANALYSIS

Let us consider a simple linear model:

\[ \frac{d}{dt} u = \lambda u \]

\[ u(t_0) = u_0 \]  \hspace{1cm} (6)

If we combine equations (3) and (4) for this model we get

\[ U_{n+1} = (1 + z + \gamma z^2)U_n \]  \hspace{1cm} (7)

where \( z = \lambda \Delta t \). Since \( \lambda \) may be complex, so is \( z \). It is well known that for stability

\[ |\sigma| < 1 \]  \hspace{1cm} (8a)

where

\[ \sigma = 1 + z + \gamma z^2. \]  \hspace{1cm} (8b)

Using \( z \) as \( z = x + iy \), a complex variable, when we plot the equality part of equations (8) in a complex plane we get a stability contour for a given value of \( \gamma \). Some of these stability contours (drawn by Lomax) have been shown here in
the figures 1-3 for $\gamma = 0.095, 0.175, \text{ and } 0.25$, respectively. Outside these stability contours, the inequality (eqs. (8)) is not valid and as such the algorithm (eq. (7)) is unstable.

If we set $\gamma = 0$, we get the stability contour for the forward Euler method. This represents a unit circle passing through the origin and $x = -2$. (This is included in fig. 3). These contours show that knowing $\lambda$, which must be real (in this case), and $\Delta t$, we can choose a $\gamma$ such that the method will remain stable. This scheme will be demonstrated in some simple examples. First we will do the truncation error analysis.

In order to make a simple analysis of truncation error we consider a simple linear model as follows:

$$\frac{du}{dt} = \lambda u + a, \quad u(0) = u_0$$

The present scheme is

$$\hat{u} = u^n + zu^n + \Delta t a + (TE)^{(n)}$$

$$u^{n+1} = (1 - \gamma) \hat{u} + \gamma [u^n + z\hat{u} + a\Delta t + (TE)^{(n)}]$$

where $z = A \Delta t$, $(TE)^{(n)} = O(\Delta t^2)$ = truncation error at $t_n$. We know that $\lim (TE)^{(n)} = 0$. This gives the consistency of the forward Euler scheme (eq. (10)).

$\Delta t \to 0$

We will now examine how errors for the combined scheme behave as $\Delta t \to 0$. Combining equations (10) and (11) we get

$$u^{n+1} = (1 + z + \gamma z^2)u^n + (1 + \gamma z)\Delta t a + (1 + \gamma z)(TE)^{(n)}$$

The term, neglected in the algorithm, is $(1 + \gamma z)(TE)^{(n)}$, which tends to zero as $\Delta t \to 0$.

We may study now the accuracy of the steady-state solution. Let us consider equation (9) again. The steady-state solution is

$$u = -a/\lambda$$

Neglecting the truncation error part in equation (12) we get

$$U^{n+1} = (1 + z + \gamma z^2)U^n + (1 + \gamma z)\Delta t a$$

where $U^n$ = the net-function corresponding to $u^n$.

At the steady-state condition, $U^{n+1} = U^n$. Thus, from equation (14), at steady-state

$$U^n = \frac{(1 + \gamma z)\Delta t}{z(1 + \gamma z)} \cdot a = - \frac{a}{\lambda} \quad (z = \lambda \Delta t)$$

Thus, the present numerical scheme leads to the same analytical steady-state solution as the one given by the differential equation itself.
APPLICATIONS

Let us consider two simple applications, one linear and the other nonlinear.

Application No. 1:

\[
\frac{du}{dt} = -100(u - \sin t) + \cos t
\]

\( u(0) = 0 \) \hspace{1cm} (15)

The analytical solution is given by

\[ u = \sin t \] \hspace{1cm} (16)

Here: \( \lambda = -100 \). If we choose \( \Delta t = 0.1 \), \( z = \lambda \Delta t = -10.0 \). From figure 1, \( z = -10 \) is within the stability contour for \( \gamma = 0.095 \). The computational results obtained here are given in table 1.

Obviously, if we set \( \gamma = 0 \), the method reduces to the forward Euler scheme which was unstable for \( \Delta t = 0.1 \). There is an interesting phenomenon here: if we reduce the step size and choose \( \Delta t = 0.06 \), then \( z = \lambda \Delta t = -6.0 \) which is exterior to the stability contour for \( \gamma = 0.095 \). Thus, the method must diverge. This has been verified computationally.

Application No. 2:

\[
\frac{du}{dt} = -25(u - 1/u)
\]

\( u(0) = \sqrt{2} \) \hspace{1cm} (17)

The analytical solution is given by

\[ u(t) = \left[ 1 + \exp(-50t) \right]^{1/2} \] \hspace{1cm} (18)

Obviously, \( \lim_{t \to \infty} u(t) = 1 \) which gives the steady-state solution.

To obtain the value of \( \lambda \) we may linearize the model near \( t = 0 \). Here,

\[ f(u, t) = -25(u - 1/u) \]

\[ \lambda = \frac{df}{du} \bigg|_{u=u_0} = -25(1 + 1/2) = -37.5 \]

Choosing \( \Delta t = 0.1 \), \( z = \lambda \Delta t = -3.75 \). Now, according to figure 1, if we choose \( \gamma = 0.095 \), since \( z = 3.75 \) is outside the contour of stability, the method must diverge (verified computationally). But \( z = -3.75 \) is well within the contour of stability for \( \gamma = 0.175 \) (fig. 2). Computational results for this case are given in table 2.
BURGERS' EQUATION

One primary objective in this work is to see how effective the method could be with regard to the solution of mathematical models represented by nonlinear partial differential equations. A very common test problem for numerical methods is Burgers' equation, which is

\[ \frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}. \]  

(19)

This is a nonlinear, parabolic, partial differential equation and as such its solution has a special significance in applied mathematics. Since, in general, closed-form solutions of this equation that are subject to a given set of initial-boundary conditions are difficult to obtain, the equation is solved numerically.

We tested the effectiveness of our algorithm with regard to solution of equation (19) subject to three distinct sets of initial-boundary conditions. Some of our findings will be discussed now.

Let us approximate \( u^n \) by the forward Euler difference formula and the space derivatives \( u_x \) and \( u_{xx} \) by central differences. The explicit finite-difference predictor is then

\[ U_j^{n+1} = U_j^n + a U_j^n (U_{j-1}^n - U_{j+1}^n) + b (U_{j-1}^n - 2U_j^n + U_{j+1}^n) \]  

(20)

where

\[ U_j^n = U(x_j, t_n) = \text{the net function corresponding to } u_j^n \]

\[ a = \frac{\Delta t}{2\Delta x} , \quad b = \nu \frac{\Delta t}{\Delta x^2} \]

\( \Delta t = \text{time-step} \), \( \Delta x = \text{net spacing} \)

\( \hat{U}_j = \text{predicted value of } U_j \text{ at } t_{n+1} \).

A corrector may now be set up as follows:

\[ U_j^{n+1} = (1 - \gamma) \hat{U}_j + \gamma [U_j^n + a \hat{U}_j^n (U_{j-1}^{n+1} - \hat{U}_{j+1}^{n+1}) + b (U_{j-1}^{n+1} - 2\hat{U}_j^{n+1} + \hat{U}_{j+1}^{n+1})] \]  

(21)

\[ (j = 1, 2, \ldots, J_{\text{MAX}} - 1) \]

where boundary conditions are given at \( j = 0 \) and \( j = J_{\text{MAX}} \). As before if we now set \( \gamma = 0 \), then \( U_j^{n+1} = \hat{U}_j \), which means the corrector is not in use. We may see now that equation (21) retains accuracy for the steady-state solution.

At the steady state, \( U_j^n = U_j^{n+1} = \hat{U}_j \) for all \( j \). Thus, if we replace in equation (20) \( \hat{U}_j \) by \( U_j^n \), we get the difference equation for the steady-state condition as

\[ a U_j^n (U_{j-1}^n - U_{j+1}^n) + b (U_{j-1}^n - 2U_j^n + U_{j+1}^n) = 0 \]  

(22)

In equation (21), if we replace \( U_j^{n+1} \) and \( \hat{U}_j \) by \( U_j^n \), we get precisely the same equation (22). Thus, accuracy of the solution is retained by the algorithm (eq. (21)).
Let us examine now how the truncation error will behave under this convex mapping analysis. Assuming that \( u(x,t) \in C^{4+2} \) (four times continuously differentiable with respect to \( x \) and twice differentiable with respect to \( t \)) we have

\[
\hat{u}_j^n = u_j^n + G_j(u^n) + (TE)_j^n
\]  

(23)

where

\[
G_j(u^n) = a \frac{u_{j-1}^n - u_{j+1}^n}{2} + b(u_{j+1}^n - 2u_j^n + u_{j-1}^n)
\]

\[
(TE)_j^n = \Delta t \left( \frac{\Delta t}{2} (u_{tt})_j^n + \frac{\Delta x^2}{6} \left[ (uu_{xxx})_j^n - \frac{\nu}{2} (u_{xxxx})_j^n \right] \right) + \ldots
\]

As \( \Delta t \to 0 \) and \( \Delta x \to 0 \), \( (TE)_j^n \to 0 \).

Let \( \hat{u} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_{J_{MAX}-1})^T, u = (u_1, u_2, \ldots, u_{J_{MAX}-1})^T \). The corrector gives

\[
u_{j}^{n+1} = \frac{1}{1 - \gamma} \hat{u}_j^n + \gamma [u_j^n + H_j(\hat{u}, u^{n+1}) + (TE)_j^n]
\]

(24)

Obviously \( G_j(u^n) \) may be replaced by the functional \( H_j(u^n, u^n) \) where

\[
H_j(\hat{u}, u^{n+1}) = a \frac{\hat{u}_{j-1}^{n+1} - \hat{u}_{j+1}^{n+1}}{2} + b(\hat{u}_{j+1}^{n+1} - 2\hat{u}_j^{n+1} + \hat{u}_{j-1}^{n+1})
\]

From equations (23) and (24)

\[
u_{j}^{n+1} = u_{j}^{n} + H(u^n, u^n) + \gamma [H(\hat{u}, u^{n+1}) - H(u^n, u^n)] + (TE)_j^n
\]

(25)

If we linearize and assume that the relation

\[
H(\hat{u}, u^{n+1}) - H(u^n, u^n) = A_n(\hat{u} - u^n) + B_n(u^{n+1} - u^n)
\]

(26)

\( A_n \) and \( B_n \) are square matrices of the order \( J_{MAX} - 1 \) is valid at each time-step, we get, from equations (25) and (26),

\[
u_{j}^{n+1} - u_{j}^{n} = H(u^n, u^n) + \gamma A_n(\hat{u} - u^n) + \gamma B_n(u^{n+1} - u^n) + (TE)_j^n
\]

(27)

Since from equation (23) \( \hat{u} - u^n = H(u^n, u^n) + (TE)_j^n \), from equation (27) we get

\[
u_{j}^{n+1} - u_{j}^{n} = (I - \gamma B_n)^{-1}(I + \gamma A_n)[H^n + (TE)_j^n]
\]

(28)

where \( H^n = H(u^n, u^n) \).

We assumed that \((I - \gamma B_n)\) is invertible for all \( n \). Since as \( \Delta t \to 0 \) and \( \Delta x \to 0 \), \( (TE)_j^n \to 0 \), equation (28) shows that the present scheme maintains its consistency property. It is also clear that as steady state is approached,

\[
(I - \gamma B_n)^{-1}(I + \gamma A_n)H^n = 0
\]

giving \( H^n_j = G_j(u^n) = 0 \), which is true, provided \((I + \gamma A_n)\) is nonsingular.

In order to make some assessment of the stability properties, let us consider the linear Burgers' equation in the next section. An attempt will be made in
the future to analyze the stability properties of the nonlinear Burgers' equation, using explicit D-Mappings, as discussed in reference 5.

LINEARIZED STABILITY ANALYSIS

Let us first consider the linearized stability properties of this method. We consider

\[
\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2} \tag{29}
\]

Then the present method may be expressed as

\[
\hat{U}_j = U^n_j + a(U^n_j - U^n_{j+1}) + b(U^n_{j-1} - 2U^n_j + U^n_{j+1}) \quad \text{(predictor)} \tag{30}
\]

\[
U^{n+1}_j = (1 - \gamma)\hat{U}_j + \gamma[U^n_j + a(U^{n+1}_{j-1} - \hat{U}_{j+1}) + b(U^{n+1}_{j-1} - 2\hat{U}_j + \hat{U}_{j+1})] \quad \text{(corrector)} \tag{31}
\]

where

\[
a = \frac{c\Delta t}{2\Delta x}, \quad b = \frac{\nu\Delta t}{\Delta x^2} \tag{32}
\]

The predictor may be expressed as

\[
\hat{U}_j = (b + a)U^n_{j-1} + (1 - 2b)U^n_j + (b - a)U^n_{j+1}
\]

In matrix notation, this becomes

\[
\hat{U} = A \cdot U^n \tag{33}
\]

where \( A \) is a tridiagonal band matrix, and may be expressed as

\[
A = B(b + a, 1 - 2b, b - a) \tag{34}
\]

The corrector may also be expressed as

\[
U^{n+1}_j = \gamma(a + b)U^{n+1}_{j-1} + (1 - \gamma - 2b\gamma)\hat{U}_j + \gamma(b - a)\hat{U}_{j+1} + \gamma U^n_j
\]

In matrix notation, we express this equation as

\[
AU^{n+1} = \Theta \hat{U} + \Gamma U^n \tag{35}
\]

where

\[
\Theta = B[-\gamma(b + a), 1, 0] \tag{36}
\]

\[
\Gamma = \gamma I \tag{37}
\]
Combining equations (33) and (35), we get
\[ u^{n+1} = \Lambda^{-1}(\Theta \cdot A + \Gamma)U^n \]  
(39)

The amplification matrix \( M \) is thus
\[ M = \Lambda^{-1}(\Theta \cdot A + \Gamma) \]  
(40)

Stability is obtained if \( M \) is a convergent matrix which is true if we can find a norm such that
\[ \|M\| < 1 \]  
(41)

Some computational experiments have been conducted to find \( \rho(M) \) numerically; \( \rho(M) \) is the spectral radius of the amplification matrix \( M \) for some given \( \Delta t, \Delta x, \nu, \) and \( \gamma \in (0,1) \). (For \( \Delta t = \Delta x = 0.05, \nu = 10^{-3}, \rho(M) < 1 \) for \( \gamma = 1.01 \).) The results are given in table 3 (\( c = 1 \) in all the cases).

If we consider the predictor alone (eq. (30)), for stability we need \( b \leq 1/2 \) and \( a \leq b \). These give, respectively,
\[ \frac{\nu \Delta t}{\Delta x^2} \leq 1/2 \quad \text{and} \quad \frac{\Delta x}{\nu} \leq 2 \quad (c = 1) \]  
(42)

If we consider \( \Delta t = \Delta x = 0.05, \nu = 0.01 \), the first criterion is satisfied; however, \( \Delta x / \nu = 5 > 2 \).

The predictor alone is unstable, whereas the present predictor-corrector is stable (first row in table 3). Also if we consider \( \Delta t = 0.05, \Delta x = 0.025, \nu = 0.01 \), both inequalities (eq. (42)) are violated.

Again, whereas the predictor itself is unstable when it is combined with the corrector for \( \gamma = 0.28 \) (table 3), the method becomes stable. Table 3 has been formed by computing \( \rho(M) \), where \( M \) is given in equation (40) for a given \( \Delta t, \Delta x, \) and \( \nu, \) and choosing \( 0 \leq \gamma \leq 1 \) (in general). Table 3 was done by Strate (private communication).

Fourier stability analysis will be done in the future.

**COMPUTATIONAL RESULTS**

At each \( t_n \), we first computed \( U_j \) for \( j = 1,2, \ldots, (J\text{MAX} - 1) \), using the predictor (eq. (20)); then in a SUBROUTINE we corrected all the predicted values by using equation (21), where the most recent corrected values were used.

The code and its description appeared in reference 8. It is written in Extended BASIC for the TRS-80 color computer made by Radio Shack by the second author. The boundary conditions were kept fixed at \( j = 0 \) and \( j = J\text{MAX} \).

Case 1:
\[ u(x,0) = \sin \pi x \]
\[ u(0,t) = u(1,t) = 0 \]  
(43)
This represents a sine wave which decays as $t \to \infty$. The analytical solution as given in references 2 and 7 is

$$u(x,t) = 2\pi \nu \frac{S_1}{S_2}$$  \hspace{1cm} (44)

where

$$S_1 = \sum_{n=1}^{\infty} n \exp(-n^2\pi^2\nu t)A_n \sin(n\pi x)$$  \hspace{1cm} (45)

$$S_2 = \sum_{n=0}^{\infty} \exp(-n^2\pi^2\nu t)A_n \cos(n\pi x)$$  \hspace{1cm} (46)

$$A_0 = \int_0^1 \exp[(\cos \pi x - 1)/2\pi \nu] dx$$  \hspace{1cm} (47)

$$A_n = 2 \int_0^1 \exp[(\cos \pi x - 1)/2\pi \nu] \cos(n\pi x) dx$$  \hspace{1cm} (48)

It may be noticed that for all $n$ and $\nu \neq 0$ the integrals are convergent and $\lim_{n \to \infty} A_n = 0$. The rate of convergence slows down as $\nu$ gets smaller. Thus, for small values of $\nu$, it is extremely difficult to compute $u(x,t)$ using the analytical expression (eq. (44)).

The computational results obtained by the present scheme have been plotted in figure 4. Here $\nu = 0.01$, $\gamma = 0.25$, $\Delta x = 0.05$, and $\Delta t = 0.1$. The time $= 2.5$ sec. Evidently, as $t \to \infty$, $u(x,t) \to 0$. (The method failed if we set $\gamma = 0$.) The sharp overshooting near $x = 1$ has been caused by the boundary conditions. Such solution patterns are quite well known; they are not time-accurate.

Case 2:

$$u(x,0) = \begin{cases} 1 & 0 \leq x \leq 0.1 \\ 0 & x > 0.1 \end{cases}$$  \hspace{1cm} (49)

This represents a moving shock. Here we chose $\gamma = 0.75$ and $\nu = 0.01$. Figure 5 describes the solution. Because of the nonconservative form of the algorithm, locations of the shock were not traced accurately. As expected, overshooting was found at $x = 1$. It was not, however, propagated upstream. With $\gamma = 0$ the oscillations were unbounded, which is true for Euler's forward scheme by eigenvalue analysis. In figure 6 an appropriate scale was used to show some of these oscillations.

Case 3:

$$\begin{align*}
 u(x,0) &= \phi(x,0) \\
 u(0,t) &= \phi(0,t) \\
 u(l,t) &= \phi(1,t)
\end{align*}$$  \hspace{1cm} (50)
where

\[ \phi(x,t) = \frac{0.1e^{-A} + 0.5e^{-B} + e^{-C}}{e^{-A} + e^{-B} + e^{-C}} \]  \hspace{1cm} (51)

\[ A = (0.5/v)(x - 0.5 + 4.95t) \]
\[ B = (2.5/v)(x - 0.5 + 0.075t) \]
\[ C = (5/v)(x - 0.375) \]  \hspace{1cm} (52)

This model is discussed in references 3 and 4. It represents the motions of two shocks which merge into one at the steady state. With \( v = 0.022 \) and \( \gamma = 0.25 \), \( \phi \)'s were computed for up to 15 time-steps and plotted in figure 7. Similar profiles \( j \) were given in references 4 and 5. With \( \gamma = 0 \), the method failed. Oscillations are shown in figure 8.

In order to remove instabilities, \( \partial u/\partial x \) was approximated by upwind differencing (MacCormack). The predictor is now

\[ \hat{U}_j = U^n_j + a(U^n_{j-1} - U^n_j) + b(U^n_{j-1} - 2U^n_j + U^n_{j+1}) \]  \hspace{1cm} (upwind-predictor) \hspace{1cm} (53)

The corrector is

\[ U^{n+1}_j = (1 - \gamma)\hat{U}_j + \gamma[U^n_j + a(\hat{U}^{n+1}_{j-1} - \hat{U}_j) + b(\hat{U}^{n+1}_{j-1} - 2\hat{U}_j + \hat{U}^{n+1}_{j+1})] \]  \hspace{1cm} (54)

where \( a = \Delta t/\Delta x \). Other parameters are the same as discussed before.

Applying this formula to case 1, with \( \Delta x = 0.05 \), \( \Delta t = 0.1 \), \( v = 10^{-6} \), and \( \gamma = 0.4 \) we obtain figure 9. Instabilities were removed; however, with \( \gamma = 0 \), the method failed (fig. 10).

The same was true for cases 2 and 3. The motion of the shock is shown in figure 11 for case 2, in which \( \Delta x = 0.05 \), \( \Delta t = 0.1 \), \( v = 0.01 \), and \( \gamma = 0.25 \). There was no overshooting. However, when \( \gamma = 0 \), the method failed after six time-steps; this is shown in figure 12.

Figure 13 shows no overshooting for case 3 (\( \Delta x = 0.05 \), \( \Delta t = 0.1 \), \( v = 0.022 \), \( \gamma = 0.15 \), number of time steps = 22). Figure 14 shows that the method collapsed with \( \gamma = 0 \).

In figure 15 it is shown that even when \( \Delta t = 4\Delta x \) (\( \Delta t = 0.2 \), \( \Delta x = 0.05 \)), the method may still be effective for representing the motion of the shock. Figure 16 shows somewhat interesting results for case 2. Here \( \Delta x = 0.05 \), \( \Delta t = 4\Delta x \), \( v = 0.01 \), and \( \gamma = 0.355 \). Numerical solutions initially showed some strong oscillations. Later they were damped out. Although the results were not time-accurate, the converged form of solution was correct.

In figures 17-19, it is shown that for the present explicit predictor-corrector scheme, \( \Delta t \), which is much larger than \( \Delta x \), may be used for small values of \( v \) if we use the upwind predictor (eq. (27)). It has also become clear that although steady-state solutions are correct here, intermediary solutions at various time-steps are not acceptable.
Larger $\Delta t$ does not necessarily mean a faster rate of convergence to steady state; the latter depends on how fast residuals converge to zero. This has been done in the past with regard to this method for solution of Burgers' equation expressed in the potential form. Details on this may be found in reference 9.

**DISCUSSION**

The primary objective of the predictor-corrector formulas discussed here is to obtain the steady-state solution. The analysis given in the Stability Analysis section shows that this objective is fulfilled.

The method is only first-order accurate unless $\gamma = 1/2$. Stability analysis for the linear Burgers' equation shows that we may use $\Delta t$ much larger than $\Delta x$ with a suitable $\gamma$, such that the numerical solution may remain stable (table 3). Computationally, this has been found to be true for the nonlinear Burgers' equation. In the future, nonlinear error analysis done in reference 5 using D-mappings will be applied to this problem.

More analyses and computational experiments are needed in order to make a more realistic assessment of the present predictor-corrector schemes for numerical solution of nonlinear models.
APPENDIX

THE CODE IN TRS-80 COLOR-EXTENDED BASIC

10 REM---------------------------------------------
11 REM
20 REM BURGERS' EQUATION
25 REM
30 REM DU/DT+U*DU/DX=NU*D2U/DX2
35 REM
40 REM A PREDICTOR-CORRECTOR METHOD
45 REM
50 REM U=VALUE OF U AT THE RECENT TIME STEP
55 REM U0=VALUE OF U AT THE PREVIOUS TIME STEP
57 REM G=A CONVEX PARAMETER
60 REM JMAX=FIELD SIZE
61 REM NTSTEPS=NO. OF TIME STEPS
63 REM K=LEVEL OF ITERATION
89 REM
90 REM---------------------------------------------
99 REM
100 DIM U(51),U0(51)
109 REM
110 REM---------------------------------------------
115 REM INPUT PARAMETERS
117 REM---------------------------------------------
120 DX=0.05:DT=0.1:JMAX=21:NTSTEPS=200
125 NU=0.01:PI=3.1416
130 A=0.5*DT/DX:B=NU*DT/(DX*DX)
132 K=0
140 REM---------------------------------------------
142 REM CHOOSE G
145 REM---------------------------------------------
150 G=0.25
160 REM---------------------------------------------
162 REM SET INITIAL & BOUNDARY CONDITIONS
165 REM---------------------------------------------
170 FOR J=1 TO JMAX:X=(J-1)*DX:U(J)=SIN(PI*X):NEXT J
179 REM
180 REM---------------------------------------------
182 REM RECORD PREVIOUS TIME STEP SOLUTIONS
185 REM---------------------------------------------
190 FOR J=1 TO JMAX:U0(J)=U(J):NEXT J
198 REM
200 REM---------------------------------------------
212 REM ITERATIONS START
215 REM---------------------------------------------
220 K=K+1
229 REM
230 REM---------------------------------------------
232 REM EULER FORWARD:PREDICTOR
234 REM---------------------------------------------
240 FOR J=2 TO JMAX-1
245 \[ U(J) = U_0(J) + A \cdot U_0(J) \cdot (U_0(J-1) - U_0(J+1)) + B \cdot (U_0(J-1) - 2 \cdot U_0(J) + U_0(J+1)) \]

250 NEXT J
260 GOSUB 1000
270 REM--------------------------------------------
275 REM OUTPUT
280 PRINT "AT TIME STEP="; K
290 FOR J=1 TO JMAX: X=(J-1)*DX
295 PRINT X, U(J)
300 NEXT J
320 IF K<NTSTEPS THEN 190
800 END
900 REM--------------------------------------------
902 REM CONVEX CORRECTOR: CHARLIE
905 REM--------------------------------------------
1000 FOR J=2 TO JMAX-1
1010 TEMP=(1.-G)\cdot U(J)
1030 V=U_0(J) + A \cdot U_0(J) \cdot (U_0(J-1) - U_0(J+1)) + B \cdot (U_0(J-1)-2 \cdot U_0(J) + U_0(J+1))
1050 U(J)=TEMP+G\cdot V
1060 NEXT J
1090 RETURN
REFERENCES


### TABLE 1.- COMPARISON BETWEEN ANALYTICAL AND COMPUTATIONAL VALUES OF $U$: $v = 0.095$, $\Delta t = 0.1$

<table>
<thead>
<tr>
<th>Time</th>
<th>$U$ (Analytical)</th>
<th>$U$ (Computed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5</td>
<td>0.598472144</td>
<td>0.597493063</td>
</tr>
<tr>
<td>5.0</td>
<td>-0.958924274</td>
<td>-0.957553061</td>
</tr>
<tr>
<td>7.5</td>
<td>0.937999977</td>
<td>0.936781981</td>
</tr>
<tr>
<td>10.0</td>
<td>-0.544021111</td>
<td>-0.543440746</td>
</tr>
<tr>
<td>12.5</td>
<td>-0.0663218955</td>
<td>-0.0660338101</td>
</tr>
<tr>
<td>15.0</td>
<td>0.650287837</td>
<td>0.649245878</td>
</tr>
</tbody>
</table>

### TABLE 2.- COMPARISON BETWEEN ANALYTICAL AND COMPUTATIONAL VALUES OF $U$: $v = 0.175$, $\Delta t = 0.1$

<table>
<thead>
<tr>
<th>Time</th>
<th>$U$ (Analytical)</th>
<th>$U$ (Computed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>1.00336332</td>
<td>1.12706991</td>
</tr>
<tr>
<td>0.5</td>
<td>1.00</td>
<td>1.04808666</td>
</tr>
<tr>
<td>0.9</td>
<td>1.00</td>
<td>1.00621156</td>
</tr>
<tr>
<td>1.3</td>
<td>1.00</td>
<td>1.0001474</td>
</tr>
<tr>
<td>1.7</td>
<td>1.00</td>
<td>1.00000293</td>
</tr>
<tr>
<td>2.1</td>
<td>1.00</td>
<td>1.00000006</td>
</tr>
</tbody>
</table>
TABLE 3.- STABILITY FOR LINEAR BURGERS' EQUATION

<table>
<thead>
<tr>
<th>Case</th>
<th>Δt</th>
<th>Δx</th>
<th>$\delta_{\text{MIN}}$</th>
<th>GAMMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.05</td>
<td>0.05</td>
<td>0.6741081217</td>
<td>0.44</td>
</tr>
<tr>
<td>2</td>
<td>0.05</td>
<td>0.025</td>
<td>0.2634833436</td>
<td>0.28</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>0.05</td>
<td>0.5591445292</td>
<td>0.39</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>0.025</td>
<td>0.4932129687</td>
<td>0.22</td>
</tr>
<tr>
<td>5</td>
<td>0.2</td>
<td>0.05</td>
<td>0.8777235062</td>
<td>0.32</td>
</tr>
<tr>
<td>6</td>
<td>0.2</td>
<td>0.025</td>
<td>0.731106526</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$v = 0.01$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case</th>
<th>Δt</th>
<th>Δx</th>
<th>$\delta_{\text{MIN}}$</th>
<th>GAMMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.05</td>
<td>0.05</td>
<td>0.8349436353</td>
<td>0.18</td>
</tr>
<tr>
<td>2</td>
<td>0.05</td>
<td>0.025</td>
<td>3.8000485810</td>
<td>0.05</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>0.05</td>
<td>1.5907523689</td>
<td>0.1</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>0.025</td>
<td>8.0442946351</td>
<td>0.03</td>
</tr>
<tr>
<td>5</td>
<td>0.2</td>
<td>0.05</td>
<td>3.7391180937</td>
<td>0.05</td>
</tr>
<tr>
<td>6</td>
<td>0.2</td>
<td>0.025</td>
<td>24.990423418</td>
<td>0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case</th>
<th>Δt</th>
<th>Δx</th>
<th>$\delta_{\text{MIN}}$</th>
<th>GAMMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.05</td>
<td>0.05</td>
<td>0.95710316836</td>
<td>1.010</td>
</tr>
<tr>
<td>2</td>
<td>0.05</td>
<td>0.025</td>
<td>0.8299183481</td>
<td>0.77</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>0.05</td>
<td>0.8320843664</td>
<td>0.89</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>0.025</td>
<td>1.7710418247</td>
<td>0.37</td>
</tr>
<tr>
<td>5</td>
<td>0.2</td>
<td>0.05</td>
<td>1.9420702213</td>
<td>0.38</td>
</tr>
<tr>
<td>6</td>
<td>0.2</td>
<td>0.025</td>
<td>4.5407595319</td>
<td>0.18</td>
</tr>
</tbody>
</table>
CONVEX CORRECTOR $\gamma = 0.095$

Figure 1. - Stability contours, convex corrector: $\gamma = 0.095$.

CONVEX CORRECTOR $\gamma = 0.175$

Figure 2. - Stability contours, convex corrector: $\gamma = 0.175$. 
Figure 3.- Stability contours for forward Euler's method ($\gamma = 0$) and that for the present scheme with $\gamma = 0.25$, $|\sigma| = 1$.

Figure 4.- Velocity profiles (case 1): $v = 0.01$, $\gamma = 0.25$, $\Delta x = 0.05$, $\Delta t = 0.1$. 
Figure 5. - Velocity profiles (case 2): $v = 0.01$, $y = 0.75$, $\Delta x = 0.05$, $\Delta t = 0.1$.

Figure 6. - Method failed: $y = 0$, other parameters are the same as those in fig. 5.
Figure 7.- Velocity profiles (case 3): $v = 0.022$, $\gamma = 0.25$, $\Delta x = 0.05$, $\Delta t = 0.1$.

Figure 8.- Method failed: $\gamma = 0$, other parameters are the same as those in fig. 7.
Figure 9. - Velocity profiles using upwind differencing (eqs. (53) and (54)) for case 1: $v = 10^{-6}$, $\gamma = 0.4$, $\Delta x = 0.05$, $\Delta t = 0.1$.

Figure 10. - Method failed (eq. (53)): $\gamma = 0$, other parameters are the same as those in fig. 9.
Figure 11.- Velocity profiles using upwind differencing for case 2: $v = 0.01, \gamma = 0.25, \Delta x = 0.05, \Delta t = 0.1$.

Figure 12.- Velocity profiles upwind differencing for case 2: $\gamma = 0$, other parameters are the same as those in fig. 11.
Figure 13. – Velocity profiles using upwind differencing for case 3: $v = 0.022$, $\gamma = 0.15$, $\Delta x = 0.05$, $\Delta t = 0.1$.

Figure 14. – Velocity profiles using upwind differencing for case 3: $\gamma = 0$, other parameters are the same as those in fig. 13.
Figure 15. - Velocity profiles using upwind differencing for case 2: $v = 0.01$, $\gamma = 0.15$, $\Delta x = 0.05$, $\Delta t = 0.2$.

Figure 16. - Velocity profiles using central differencing for case 2: $v = 0.01$, $\gamma = 0.355$, $\Delta x = 0.05$, $\Delta t = 0.2$. 
Figure 17. - Velocity profiles using upwind differencing for case 1: \( v = 0.001, \gamma = 0.25, \Delta x = 0.05, \Delta t = 0.2 \).

Figure 18. - Velocity profiles using upwind differencing for case 1: \( v = 0.00001, \gamma = 0.25, \Delta x = 0.05, \Delta t = 0.2 \), number of time steps = 30.
Figure 19. - Velocity profiles using upwind differencing for case 1: \( \nu = 0.00001, \gamma = 0.152, \Delta x = 0.05, \Delta t = 0.3 \), number of time steps = 40.
Forward Euler's explicit, finite-difference formula of extrapolation, is used as a predictor and a convex formula as a corrector to integrate differential equations numerically. An application has been made to Burgers' equation.