

Homework #6:
**Numerical Diffusion and Dispersion: Solution of the 1-D
Linear Advection Equation**

by

Joseph P. Kubitschek
27 November 1998

ABSTRACT

Various numerical schemes were used to solve the linear advection (i.e. 1-D wave) equation as a means for improved understanding of numerical diffusion and dispersion inherent to many numerical schemes. The linear advection equation, subject to an initial sinusoidal disturbance, was solved over a 2.0-m spatial domain using the first order Upwind scheme; the second order MacCormack scheme; the second order Lax-Wendroff scheme; and the second order Lax-Wendroff scheme with diffusion/anti-diffusion. Comparison of the results indicates that the Upwind scheme is too diffusive. Conversely, both the MacCormack and Lax-Wendroff schemes are too dispersive. However, in the case of the Lax-Wendroff scheme, the inherent dispersion was eliminated using the appropriate diffusion/anti-diffusion producing improved results as compared with all other cases investigated.

INTRODUCTION

Background

The linear advection equation is a hyperbolic partial differential equation with a steady propagating solution and is written as,

$$\partial u / \partial t + a(\partial u / \partial x) = 0 \quad \text{or} \quad \partial E / \partial t + \partial F / \partial x = 0 \quad (\text{flux conservative form with } E = u \text{ and } F = au).$$

Physically, this equation governs the linear transport of some parameter u where the solution represents wave propagation without changes in shape (i.e. inviscid fluid). For the purposes of this investigation u is taken as the velocity of some initial disturbance that propagates across a pre-specified domain. In this case, a is taken as 200 m/s and the initial disturbance is defined as

$$u = U \sin(2\pi x) \quad \text{for } 0 \leq x \leq 1.0\text{m}, \\ u = 0.0 \quad \text{for } 1.0\text{m} \leq x \leq 2.0\text{m}.$$

Where $U=10.0$ m/s, and x represents the 2.0-m spatial domain across which the disturbance propagates. Such an equation as this lends nicely to the use of finite difference techniques to obtain a time-marching solution in the spatial domain.

Solution Techniques

Upwind Scheme

The Upwind finite difference scheme for the linear advection equation is first order accurate and may be written as

$$u_j^{n+1} = (1 - C)u_j^n + Cu_{j-1}^n.$$

Where $C = a\Delta t/\Delta x$ and represents the Courant number. Examination of stability indicates this scheme to be conditionally stable. That is the solution is stable provided $C \leq 1$ (Courant-Friedrichs-Lewy or CFL condition). However, such a condition introduces first order truncation error in both space and time and hence inherently introduces what may be considered as numerical or artificial diffusion that influences the accuracy of the scheme.

MacCormack Scheme

The MacCormack method is a second order accurate predictor-corrector finite difference scheme that may be written as

$$u_{j-1}^{n+1} = u_j^n - \frac{1}{2}C[(1 - C)u_{j+1}^n + 2Cu_j^n - (1 + C)u_{j-1}^n].$$

Again, a stability analysis indicates the scheme to be stable provided $|C| \leq 1$. Intuitively, the accuracy of this scheme is expected to be much improved over the first order Upwind scheme. However, as will be discussed, such is not entirely the case as numerical dispersion is introduced due leading order of the truncation error.

Lax-Wendroff Scheme

The Lax-Wendroff scheme, also a second order scheme, is an example of a two-step scheme and may be written as

$$u_{j-1}^{n+1} = u_j^n - C[\frac{1}{2}(u_{j+1}^n + u_j^n) - \frac{1}{2}C(u_{j+1}^n - u_j^n) - \frac{1}{2}C(u_j^n + u_{j-1}^n) + \frac{1}{2}C(u_j^n - u_{j-1}^n)].$$

Again a stability analysis produces the CFL condition making the solution numerically stable provided $C \leq 1$. However, this scheme is also excessively dispersive due to leading order truncation error.

Lax-Wendroff Scheme with diffusion/anti-diffusion

Although the Lax-Wendroff scheme is too dispersive, improved results may be obtained by first introducing diffusion to damp the inherent dispersion and then adding anti-diffusion to recover the desired linear advection equation solution. This two-step method may be written as

$$u_j^* = u_j^n - \frac{1}{2}C(u_{j+1}^n - u_{j-1}^n) + (\epsilon_1 + \frac{1}{2}C^2)(u_{j+1}^n - 2u_j^n + u_{j-1}^n),$$

$$u_j^{n+1} = u_j^* - \epsilon_2(u_{j+1}^* - 2u_j^* - u_{j-1}^*).$$

Where, $\epsilon_1 = (1 - 2C^2)/6$, $\epsilon_2 = (1 + C^2)/6$, and u_j^* represents the damped solution as the diffusion term is added in the first step and the anti-diffusion term in the second step to recover the desired solution, u_j^{n+1} (“ASEN5317 - Lecture Notes,” Kantha⁵).

RESULTS AND DISCUSSION

The results of each numerical scheme investigated are presented as figure 1-6, appendix A. Each figure represents the solution or disturbance propagation as obtained using each numerical scheme respectively. The solutions computed at 6 successive time steps are plotted over the spatial domain and provide an indication of disturbance propagation characteristics in the x-direction until it reaches the end of the domain in question (i.e. 2.0 m in this case). Initially, there was some question as to how best to ascertain when the disturbance had propagated to the upper eastern end of the spatial domain (i.e. the termination criteria for each of the routines). At first this was taken to be the departure of the eastern boundary, at $x = 2.0\text{m}$, from zero. However, the result was premature termination of the time marching. To correct this the termination criteria was set to $u_{j+1}^{n+1} = -u_{j-1}^{n+1}$ by noting that the wave form may be considered as extending beyond the eastern end of the spatial domain by a single spatial grid point. This condition was also required as a necessary boundary condition for both the MacCormack and Lax-Wendroff schemes as they are second order and centered in space.

Upwind Scheme

Figure 1 represents the solution to the linear advection equation using the first order upwind scheme with a Courant number, $C=0.8$. As can be seen, this scheme introduces artificial viscosity or numerical diffusion, a direct result of leading order truncation error term. This numerical diffusion is evidenced as the broadening of the wavelength and the corresponding reduction in amplitude of the disturbance. Although the dispersion is subtle, marching further in time for a larger spatial domain would provide an enhanced perspective of this feature. Never the less, it is present and entirely an artifact of the numerical scheme (i.e. it is independent of the physics).

MacCormack Scheme

In contrast to the Upwind scheme, the MacCormack scheme exhibits solution characteristics that are too dispersive. Again this artificial feature is attributed to leading third order truncation error term. And, because this is a second order scheme the leading third order error term causes the dispersion evident in the small oscillations illustrated by figure 2.

Lax-Wendroff Scheme

Similar the MacCormack scheme, the Lax-Wendroff scheme is also too dispersive. The results exhibited by figure 3 show the dispersion manifest as small-scale oscillations in the successive time-step solutions.

Lax-Wendroff Scheme with anti-diffusion

The problem of dispersion may be handled using the diffusion/anti-diffusion method as previously discussed. In this case, adding artificial diffusion acts to offset the excessive dispersion inherent to the scheme. This is achieved by means of a two step algorithm in which the first step adds diffusion to obtain a damped solution and the second step applies anti-diffusion to recover the desired solution. The effect is much improved results. However, it should be noted that adding diffusion impacts both the accuracy and stability of the numerical scheme. Although artificial diffusion reduces accuracy it has the benefit of improving numerical stability. As such three different magnitudes of diffusion were added as a means of obtaining the optimal diffusion by trial and error. Figure 4 represents the initial diffusion addition investigated as previously defined. However, this magnitude is apparently excessive, as the solution is observed to behave similar to that of the Upwind scheme solution (i.e. too diffusive). To correct this, the diffusion term was decreased by a factor of 2. Figure 5 represents this case and illustrates improved results without the previously seen excessive diffusion inherent in the Upwind scheme. To test this, the original magnitude of diffusion was again decreased, this time by a factor of 10. The result was excessive dispersion similar to that for the scheme without diffusion/anti-diffusion, as expected. Figure 6 illustrates these results and on comparison appears very much similar to the original Lax-Wendroff solution as insufficient diffusion was added. Therefore, the results presented as figure 5 represent somewhat of a happy medium between excessive diffusion exhibited by the Upwind scheme and excessive dispersion exhibited by both the MacCormack and Lax-Wendroff schemes and hence diffusion/anti-diffusion algorithm is undoubtedly the desired method for solution of the linear advection equation.

CONCLUSIONS

- Based on these results, the Upwind scheme exhibits excessive diffusion, an artifact of the second or leading truncation error term. In contrast, both the MacCormack and the Lax-Wendroff schemes exhibit excessive dispersion as a result of the leading, third order truncation error term.
- In the case of the excessive dispersion exhibited by the Lax-Wendroff scheme, improved results may be obtained by successively adding diffusion to the numerical scheme and then adding anti-diffusion to recover the desired solution (i.e. the solution for the linear advection equation).
- The magnitude of diffusion applied to improve the results of the Lax-Wendroff scheme must be appropriate for the problem at hand and the value of the Courant number. Adding diffusion for a particular value of C has the benefit of improving numerical stability but this is at the price of degraded accuracy as adding excessive diffusion creates the original problem inherent to the Upwind scheme. Therefore, trial and error is likely required to obtain a sound solution that exhibits good accuracy (i.e. without excessive diffusion) without adverse dispersion.

REFERENCES

1. Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B.P., Numerical Recipes in Fortran 77, 2nd Edition, Volume 1, Cambridge University Press, 1992.
2. Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B.P., Numerical Recipes in Fortran 77 – Examples Book, Cambridge University Press, 1992.
3. Peyret, R., and Taylor, T.D., Computational Methods for Fluid Flow, Springer-Verlag, 1983.
4. Anderson, J.D., Computational Fluid Dynamics, McGraw-Hill, Inc., 1995.
5. Kantha, L., “ASEN5317 - Lecture Notes,” University of Colorado at Boulder, Fall 1998.

APPENDIX A – FIGURES

Linear Advection Equation Solution

Upwind Scheme (C = 0.8)

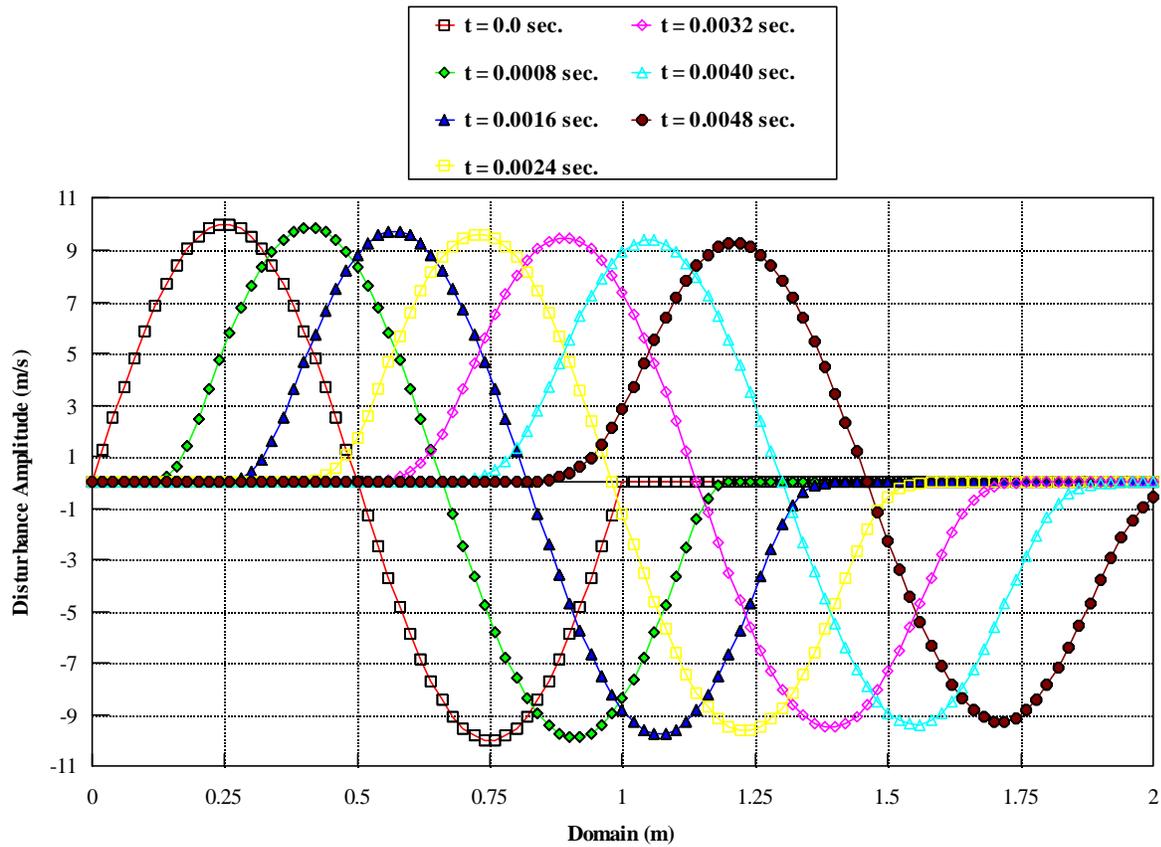


Figure 1. - Results of first order diffusive upwind scheme. Plot of disturbance propagation at various time steps across the domain, 0-2.0 m for Courant number, $C = 0.8$.

Linear Advection Equation Solution

MacCormack Scheme (C = 0.8)

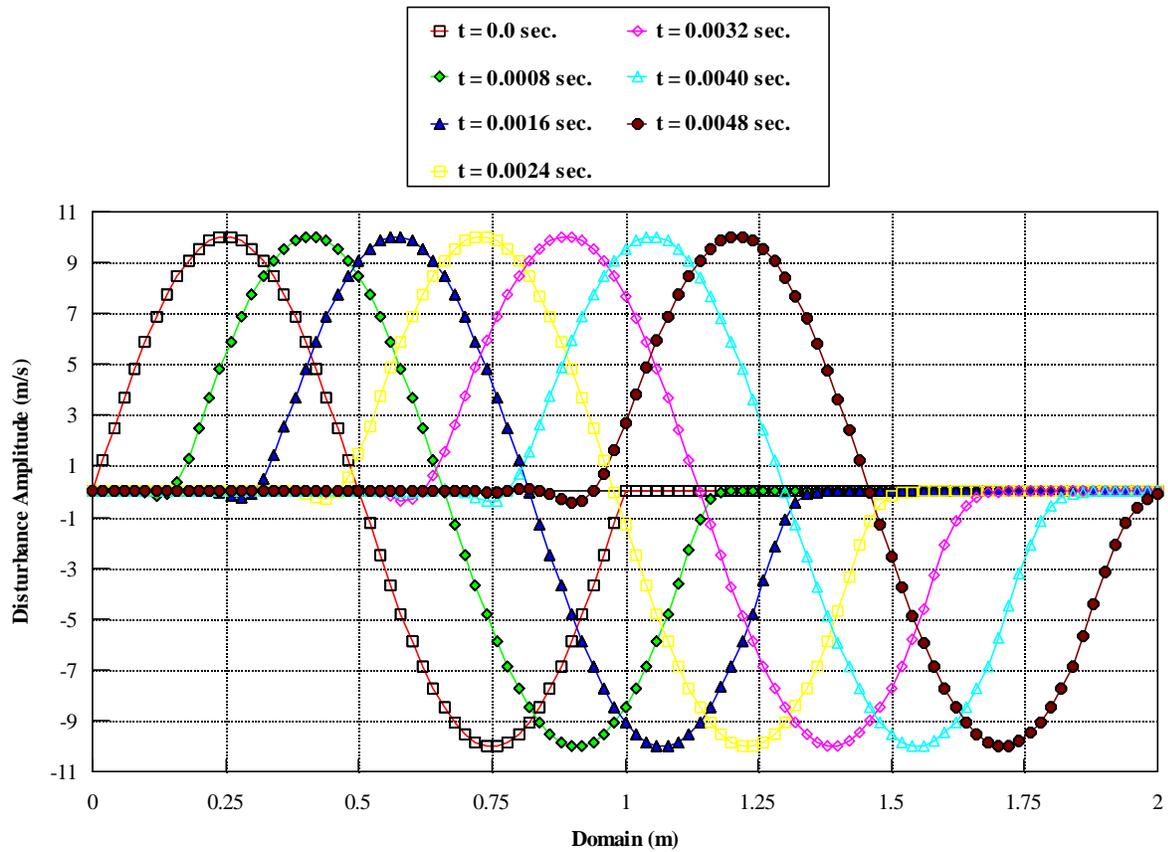


Figure 2. - Results of second order dispersive MacCormack scheme. Plot of disturbance propagation at various time steps across the domain, 0-2.0 m for Courant number, $C = 0.8$.

Linear Advection Equation Solution

Lax-Wendroff Scheme (C = 0.8)

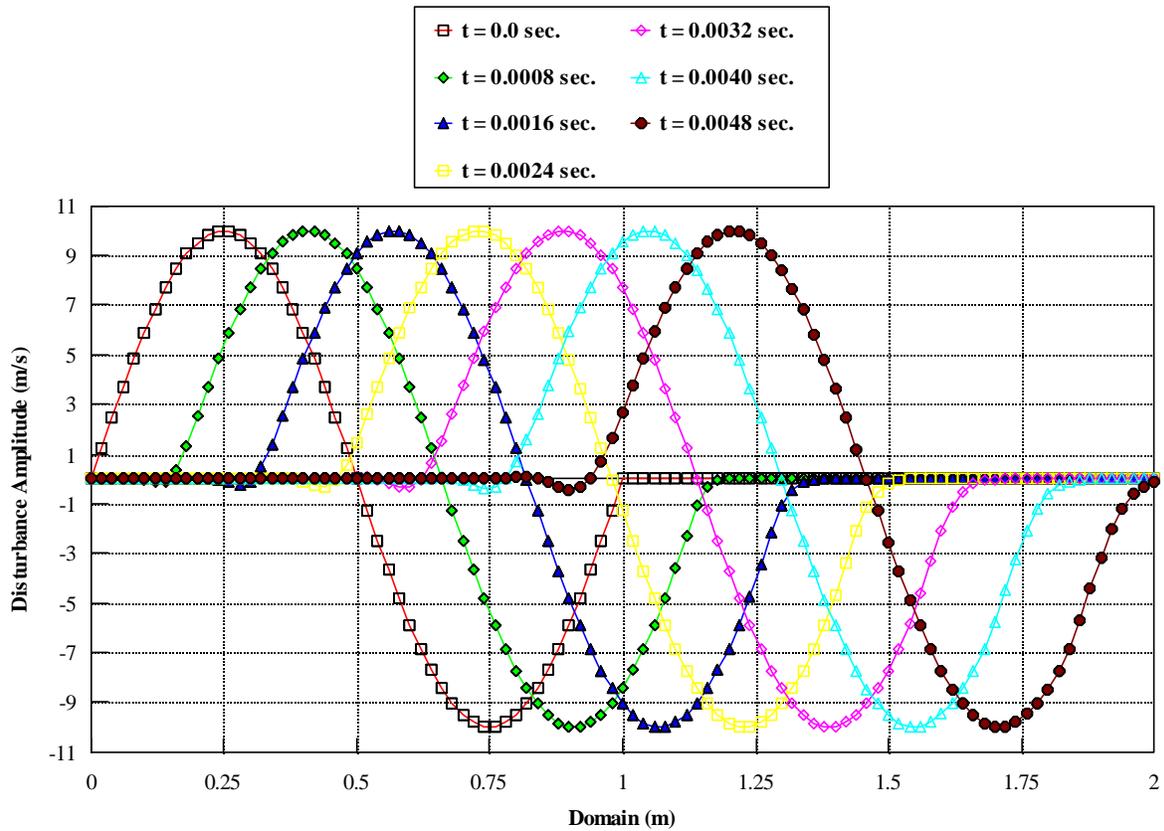


Figure 3. - Results of second order dispersive Lax-Wendroff scheme. Plot of disturbance propagation at various time steps across the domain, 0-2.0 m for Courant number, $C = 0.8$.

Linear Advection Equation Solution

Lax-Wendroff Scheme w/ anti-diffusion ($C = 0.8$)

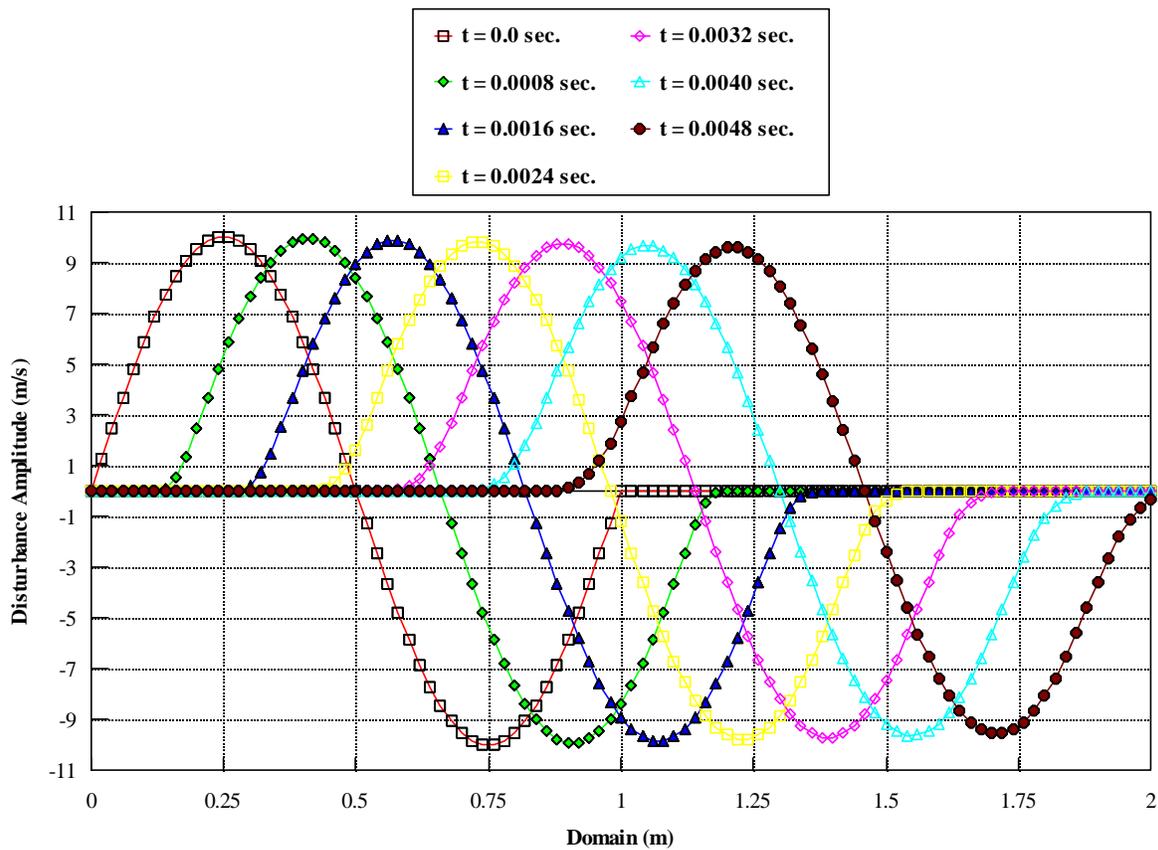


Figure 4. - Results of second order dispersive Lax-Wendroff scheme with diffusion/anti-diffusion applied. Plot of disturbance propagation at various time steps across the domain, 0-2.0 m for Courant number, $C = 0.8$.

Linear Advection Equation Solution

Lax-Wendroff Scheme w/ anti-diffusion (C = 0.8)

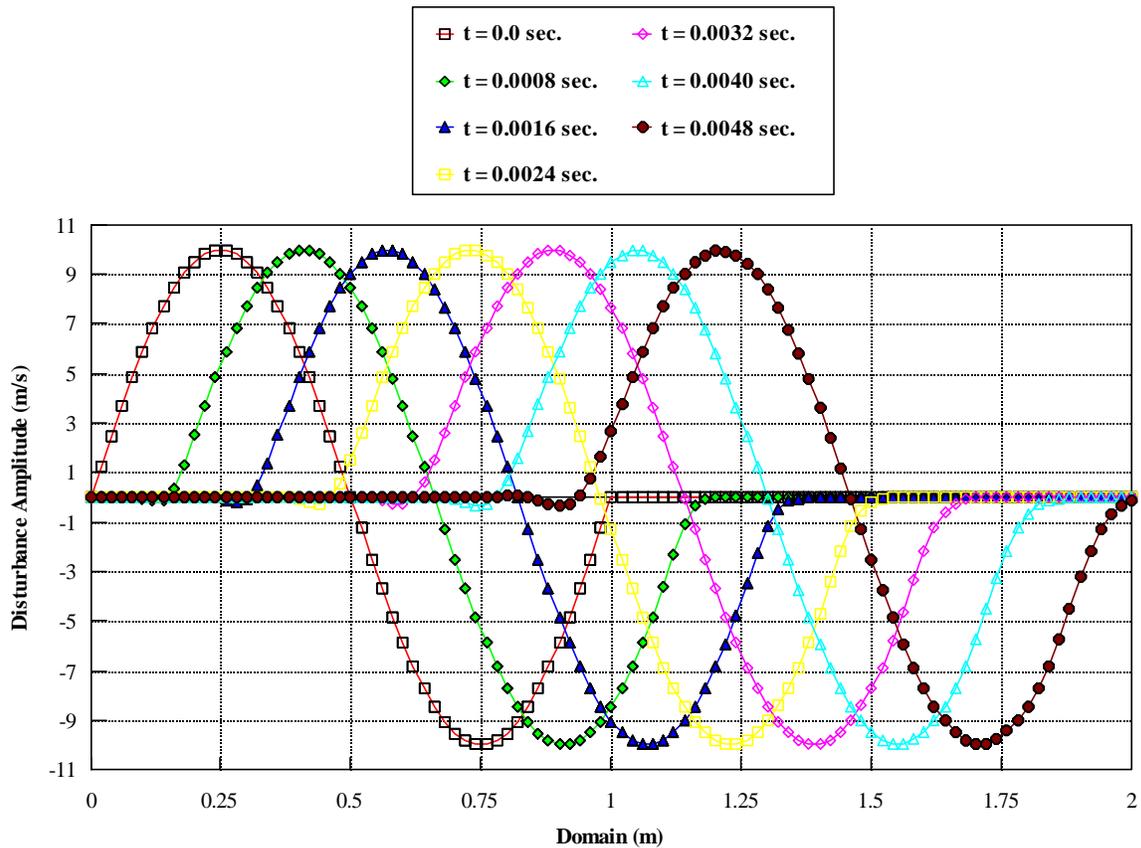


Figure 6. - Results of second order dispersive Lax-Wendroff scheme with anti-diffusion applied. Plot of disturbance propagation at various time steps across the domain, 0-2.0 m for Courant number, $C = 0.8$. Further reduction in diffusion addition nearly recovers the original dispersive nature of the scheme.

APPENDIX B – FORTRAN CODE

1.) **1st Order Upwind Scheme:** The driver code, *hw6a.f* implements the Upwind scheme with $C = 0.8$.

```
program hw6a
C
parameter(nmax=102,pi=3.141592654)
integer n,m,count
real uold(nmax),unew(nmax),x(nmax)
real t,tau,delx,c,u,a
C
data delx,n,xmax,a,u,c /0.02,101,2.0,200.0,10.0,0.8/
C
open(unit=1,file='hw6a.out',status='unknown')
tau=c*delx/a
m=(n-1)/2
C
count=0
t=0.0
x(1)=0.0
C
do 2 i=2,n
  x(i)=x(i-1)+delx
2 continue
C
do 3 i=1,m+1
  uold(i)=u*sin(2.0*pi*x(i))
3 continue
C
do 4 i=m+2,n
  uold(i)=0.0
4 continue
C
write(1,*)'Velocity Results:'
write(1,10)t,(uold(j),j=1,n)
do while (abs(uold(n-1)).lt.1.25)
  count=count+1
  t=t+tau
  do 5 i=2,n
    unew(i)=(1.0-c)*uold(i)+c*uold(i-1)
5 continue
C
do 6 i=2,n
  uold(i)=unew(i)
6 continue
C
write(1,10)t,(uold(j),j=1,n)
end do
C
10 format(2x,f6.5,5x,101f10.6)
write(1,(' Number of time steps for propagation= ',i4))count
end
```

2.) **2nd Order MacCormack Scheme:** The driver code, *hw6b.f* implements the MacCormack scheme with $C = 0.8$.

```

program hw6b
C
parameter(nmax=102,pi=3.141592654)
integer n,m,count
real uold(nmax),unew(nmax),x(nmax)
real t,tau,delx,c,u,a
C
data delx,n,xmax,a,u,c /0.02,101,2.0,200.0,10.0,0.8/
C
open(unit=1,file='hw6b.out',status='unknown')
tau=c*delx/a
m=(n-1)/2
C
count=0
t=0.0
x(1)=0.0
C
do 2 i=2,n
  x(i)=x(i-1)+delx
2 continue
C
do 3 i=1,m+1
  uold(i)=u*sin(2.0*pi*x(i))
3 continue
C
do 4 i=m+2,n
  uold(i)=0.0
4 continue
C
write(1,*)'Velocity Results:'
write(1,10)t,(uold(j),j=1,n)
do while (abs(uold(n-1)).lt.1.25)
  uold(n+1)=uold(n-1)
  count=count+1
  t=t+tau
  do 5 i=2,n
    unew(i)=uold(i)-(c/2)*((1.0-c)*uold(i+1)+2*c*uold(i)-
$      (1+c)*uold(i-1))
5 continue
C
do 6 i=2,n
  uold(i)=unew(i)
6 continue
C
write(1,10)t,(uold(j),j=1,n)
end do
C
10 format(2x,f6.5,5x,101f10.6)
write(1,(' Number of time steps for propagation= ',i4))count
end
```

3.) **2nd Order Lax-Wendroff Scheme:** The driver code, *hw6c.f* implements the Lax-Wendroff scheme with $C = 0.8$.

```

program hw6c
C
parameter(nmax=102,pi=3.141592654)
integer n,m,count
real uold(nmax),unew(nmax),x(nmax)
real t,tau,delx,c,u,a
C
data delx,n,xmax,a,u,c /0.02,101,2.0,200.0,10.0,0.8/
C
open(unit=1,file='hw6c.out',status='unknown')
tau=c*delx/a
m=(n-1)/2
C
count=0
t=0.0
x(1)=0.0
C
do 2 i=2,n
  x(i)=x(i-1)+delx
2 continue
C
do 3 i=1,m+1
  uold(i)=u*sin(2.0*pi*x(i))
3 continue
C
do 4 i=m+2,n
  uold(i)=0.0
4 continue
C
write(1,*)'Velocity Results:'
write(1,10)t,(uold(j),j=1,n)
do while (abs(uold(n-1)).lt.1.25)
  uold(n+1)=uold(n-1)
  count=count+1
  t=t+tau
  do 5 i=2,n
    unew(i)=uold(i)-c*(0.5*(uold(i+1)+uold(i))-(c/2.0)*(uold(i+1)-
$      uold(i))-0.5*(uold(i)+uold(i-1))+(c/2.0)*(uold(i)-
$      uold(i-1)))
5 continue
C
do 6 i=2,n
  uold(i)=unew(i)
6 continue
C
write(1,10)t,(uold(j),j=1,n)
end do
C
10 format(2x,f6.5,5x,101f10.6)
write(1,(' Number of time steps for propagation= ',i4))count
end

```

4.) **2nd Order Lax-Wendroff Scheme with anti-diffusion:** The driver code, *hw6d1.f* implements the Lax-Wendroff scheme with diffusion/anti-diffusion for $C = 0.8$.

```

program hw6d1
parameter(nmax=102,pi=3.141592654)
integer n,m,count
real uold(nmax),unew(nmax),udamp(nmax),x(nmax)
real t,tau,delx,c,u,a,epsa,epsb
data delx,n,xmax,a,u,c /0.02,101,2.0,200.0,10.0,0.8/
C
open(unit=1,file='hw6d1.out',status='unknown')
tau=c*delx/a
m=(n-1)/2
C Diffusion/anti-diffusion terms.
epsa=(1.0-2.0*c**2)/6.0
epsb=(1.0+c**2)/6.0
count=0
t=0.0
x(1)=0.0
C
do 2 i=2,n
  x(i)=x(i-1)+delx
2 continue
C
do 3 i=1,m+1
  uold(i)=u*sin(2.0*pi*x(i))
3 continue
C
do 4 i=m+2,n
  uold(i)=0.0
4 continue
C
write(1,*)'Velocity Results:'
write(1,10)t,(uold(j),j=1,n)
do while (abs(uold(n-1)).lt.1.25)
  count=count+1
  t=t+tau
  uold(n+1)=-uold(n-1)
  do 5 i=2,n
    udamp(i)=uold(i)-(c/2.0)*(uold(i+1)-uold(i-1))+(c**2/2.0)*
$      (uold(i+1)-2.0*uold(i)+uold(i-1))-
$      epsa*(uold(i+1)-2.0*uold(i)+uold(i-1))
5 continue
C
do 6 i=2,n
  unew(i)=udamp(i)+epsb*(udamp(i+1)-2.0*udamp(i)+udamp(i-1))
  uold(i)=udamp(i)
  write(*,*)uold(i)
6 continue
C
write(1,10)t,(uold(j),j=1,n)
end do
C
10 format(2x,f6.5,5x,101f10.4)
write(1,(' Number of time steps for propagation= ',i4))count
end

```

5.) **2nd Order Lax-Wendroff Scheme with diffusion/anti-diffusion:** The driver code, *hw6d2.f* implements the Lax-Wendroff scheme with less diffusion/anti-diffusion for $C = 0.8$.

```

program hw6d2
parameter(nmax=102,pi=3.141592654)
integer n,m,count
real uold(nmax),unew(nmax),udamp(nmax),x(nmax)
real t,tau,delx,c,u,a,epsa,epsb
data delx,n,xmax,a,u,c /0.02,101,2.0,200.0,10.0,0.8/
C
open(unit=1,file='hw6d2.out',status='unknown')
tau=c*delx/a
m=(n-1)/2
C Diffusion/anti-diffusion terms.
epsa=(1.0-2.0*c**2)/12.0
epsb=(1.0+c**2)/12.0
count=0
t=0.0
x(1)=0.0
C
do 2 i=2,n
  x(i)=x(i-1)+delx
2 continue
C
do 3 i=1,m+1
  uold(i)=u*sin(2.0*pi*x(i))
3 continue
C
do 4 i=m+2,n
  uold(i)=0.0
4 continue
C
write(1,*)'Velocity Results:'
write(1,10)t,(uold(j),j=1,n)
do while (abs(uold(n-1)).lt.1.25)
  count=count+1
  t=t+tau
  uold(n+1)=-uold(n-1)
  do 5 i=2,n
    udamp(i)=uold(i)-(c/2.0)*(uold(i+1)-uold(i-1))+(c**2/2.0)*
$      (uold(i+1)-2.0*uold(i)+uold(i-1))-
$      epsa*(uold(i+1)-2.0*uold(i)+uold(i-1))
5 continue
C
do 6 i=2,n
  unew(i)=udamp(i)+epsb*(udamp(i+1)-2.0*udamp(i)+udamp(i-1))
  uold(i)=udamp(i)
  write(*,*)uold(i)
6 continue
C
write(1,10)t,(uold(j),j=1,n)
end do
C
10 format(2x,f6.5,5x,101f10.4)
write(1,(' Number of time steps for propagation= ',i4))count
end

```

6.) **2nd Order Lax-Wendroff Scheme with diffusion/anti-diffusion:** The driver code, *hw6d3.f* implements the Lax-Wendroff scheme with even less diffusion/anti-diffusion for $C = 0.8$.

```

program hw6d3
parameter(nmax=102,pi=3.141592654)
integer n,m,count
real uold(nmax),unew(nmax),udamp(nmax),x(nmax)
real t,tau,delx,c,u,a,epsa,epsb
data delx,n,xmax,a,u,c /0.02,101,2.0,200.0,10.0,0.8/
C
open(unit=1,file='hw6d3.out',status='unknown')
tau=c*delx/a
m=(n-1)/2
C Diffusion/anti-diffusion terms.
epsa=(1.0-2.0*c**2)/60.0
epsb=(1.0+c**2)/60.0
count=0
t=0.0
x(1)=0.0
C
do 2 i=2,n
  x(i)=x(i-1)+delx
2 continue
C
do 3 i=1,m+1
  uold(i)=u*sin(2.0*pi*x(i))
3 continue
C
do 4 i=m+2,n
  uold(i)=0.0
4 continue
C
write(1,*)'Velocity Results:'
write(1,10)t,(uold(j),j=1,n)
do while (abs(uold(n-1)).lt.1.25)
  count=count+1
  t=t+tau
  uold(n+1)=-uold(n-1)
  do 5 i=2,n
    udamp(i)=uold(i)-(c/2.0)*(uold(i+1)-uold(i-1))+(c**2/2.0)*
$      (uold(i+1)-2.0*uold(i)+uold(i-1))-
$      epsa*(uold(i+1)-2.0*uold(i)+uold(i-1))
5 continue
C
do 6 i=2,n
  unew(i)=udamp(i)+epsb*(udamp(i+1)-2.0*udamp(i)+udamp(i-1))
  uold(i)=udamp(i)
  write(*,*)uold(i)
6 continue
C
write(1,10)t,(uold(j),j=1,n)
end do
C
10 format(2x,f6.5,5x,101f10.4)
write(1,(' Number of time steps for propagation= ',i4))count
end

```