# Results

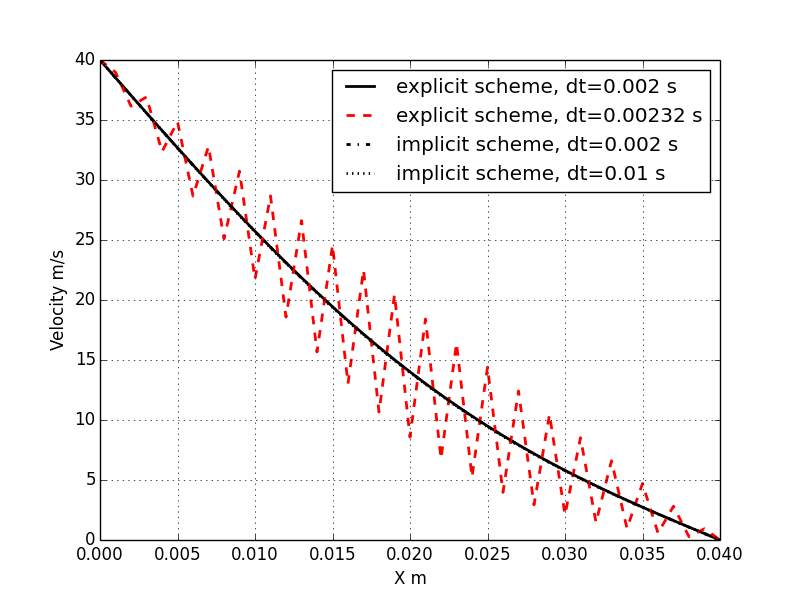


Figure simulation results at time=1.08 seconds

Figure shows that two explicit schemes and one explicit scheme with time step size of 0.002 seconds can achieve the stable and converged results. They are overlapped with each other according to the figure. The explicit scheme with time step size of 0.00232 second, which is slightly larger than the maximal time step size (0.002s), oscillates tremendously.

Results indicate that when using explicit scheme one has to strictly follow the stability law, otherwise, the calculation will be unstable. On the other hand, using implicit scheme has no limitation on the choosing of time step size theoretically. However, in practical larger time step size is undesirable due to the large truncation error followed by, although in this figure the difference of results with two time step size is not shown.

# Codes

program plate

C.. This program is used to calculate the velocity distribution within two plates,

C.. of which one is suddenly moving at u.

implicit none

C.. define the variable

double precision u(100), u0(100), u\_temp(100,100)

double precision dx, l,lx(100), t, dt, tmax, nu

double precision ub, u\_init

double precision cfl

integer imax, i, j, k, case\_no

C.. initialize the value

write (6,\*) 'please input the case no.'

write (6,\*) '1 for EXPLICIT; 2 for IMPLICIT'

read (5, \*) case\_no

write (6,\*) 'please input the time step size'

write (6,\*) 'dt=0.00200'

read (5,\*) dt

i=0

j=0

k=0

t=0.00000

l=0.0400

dx=0.00100

tmax=1.08000

ub=40.00000

u\_init=40.00000

nu=0.000217

C.. Discrete the space

imax=nint(l/dx)

do i=1,imax+1

lx(i)=(i-1)\*dx

end do

C.. give initial value

u0(1)=ub

do i=2,imax+1

u0(i)=0

end do

C.. initialize the t

t=dt

C.. set boundary conditions

u0(1)=ub

u0(imax+1)=0

u(1)=ub

u(imax+1)=0

C.. main routine for computation

select case (case\_no)

case (1)

write (6,\*) "Upwind explicit schemes"

call explict\_upwind (u,u0,imax,dx,dt,t,tmax,nu,ub,u\_temp)

case (2)

write (6,\*) "Upwind implicit schemes"

call implict\_upwind (u,u0,imax,dx,dt,t,tmax,nu,ub,u\_temp)

case default

write (6,\*) "Error with selecting the schemes"

end select

C.. output data

open (1, file='results.txt', status='unknown')

write (6, \*) 'simulation time is',tmax,'seconds'

write (6, \*),'time step size is',dt, 'seconds'

write (6, \*) 'computation length is',l, 'm'

write (1, 2001)

2001 format (10x,'NO.', 10x, 'X', 23x, 'Velocity')

do i=1,imax+1

write (1, \*) i,lx(i),u0(i)

end do

read (5, \*)

end

subroutine explict\_upwind (u,u0,imax,dx,dt,t,tmax,nu,ub,u\_temp)

implicit none

double precision u(100)

double precision u0(100), u\_temp(100,100)

integer imax, i, j

double precision dx, dt,t,tmax,nu,ub

double precision small

j=2

do while (t<=tmax)

do i=2,imax

u(i)=u0(i)+nu\*dt/dx/dx\*(u0(i+1)-2\*u0(i)+u0(i-1))

end do

do i=2,imax

u0(i)=u(i)

end do

t=t+dt

end do

return

end

subroutine implict\_upwind (u,u0,imax,dx,dt,t,tmax, nu,ub,u\_temp)

implicit none

integer imax, i,j

double precision u(100)

double precision u0(100), u\_temp(100,100)

double precision a(imax-1), b(imax-1), c(imax-1), d(imax-1)

double precision x(imax-1)

double precision dx, dt,t,tmax,nu,ub, psi

integer n

n=imax-1

C.. calculate psi

psi=nu\*dt/dx/dx

write (6,\*) "coefficient for equation is", psi

C.. Start do iteration

do while (t<=tmax)

C.. upper diagonal coefficient

do i=1,n

a(i)=psi

end do

C.. main diagonal coefficient

do i=1,n

b(i)=-1\*(2\*psi+1)

end do

C.. lower diagonal coefficient

do i=1,n

c(i)=psi

end do

C.. d constant vector

do i=1,n

if (i==n) then

d(i)=-u0(imax)-psi\*u0(imax+1)

else if (i==1) then

d(i)=-u0(i+1)-psi\*u0(i)

else

d(i)=-u0(i+1)

end if

end do

!call tdma(n,a,b,c,d,x)

call sy(n,a,b, c, d)

C.. update the u0

do i=2, imax

u0(i)=d(i-1)

end do

C.. iteration continue

t=t+dt

end do

return

end

subroutine sy(iu,bb, dd, aa, cc)

implicit none

integer il, iu, lp, i, j

double precision bb(iu), dd(iu), aa(iu), cc(iu), r

il=1

lp=il+1

do i=lp, iu

r=bb(i)/dd(i-1)

dd(i)=dd(i)-r\*aa(i-1)

cc(i)=cc(i)-r\*cc(i-1)

end do

cc(iu)=cc(iu)/dd(iu)

do i=lp,iu

j=iu-i+il

cc(j)=(cc(j)-aa(j)\*cc(j+1))/dd(j)

end do

return

end