**MSDM5004 Project 1 solution**

**Q1**

**(i-ii) Numerical solutions by explicit scheme**

J = 20;

dx = 0.05;

N = 51;

dt = 0.0012; %[0.0012,0.0013];

x = linspace(0,1,J+1);

gd = zeros(N+1,J+1);

u0 = @(x) (x<=0.5).\*(2\*x) + (x>0.5).\*(2-2\*x);

gd(1,:) = u0(x);

mu = dt/dx^2;

for n = 1:N

for j = 2:J

gd(n+1,j) = gd(n,j) + mu\*(gd(n,j+1) - 2\*gd(n,j) + gd(n,j-1));

end

end

time = [1, 2, 26, 51];

for i = 1:4

subplot(4,1,i);

plot(x, gd(time(i),:))

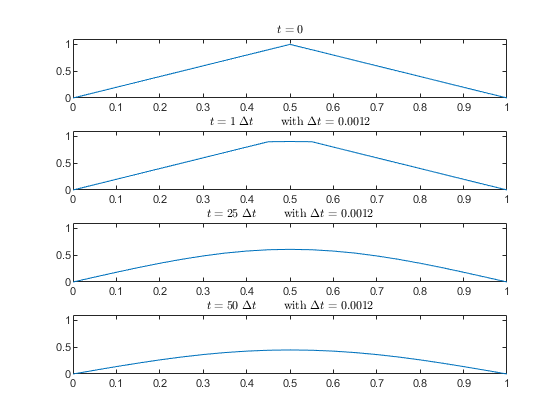
ylim([0, 1.1])

tt = {'$$t=', num2str(time(i)-1),'$$', '$$\Delta t\quad\quad$$ with $$\Delta t=$$', num2str(dt)};

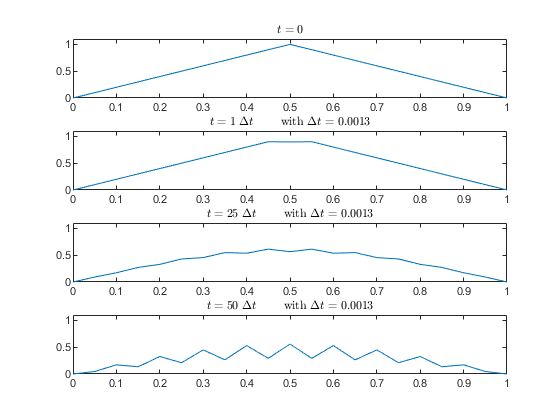
title(join(tt(1:3+(i>1)\*2)),'interpreter','latex');

end

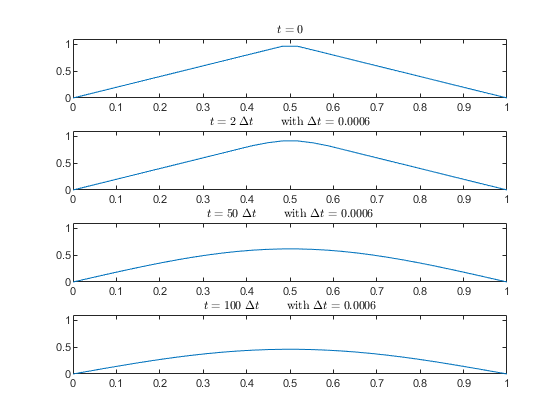
For :



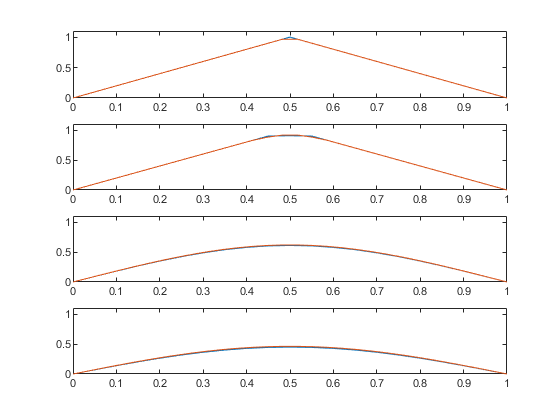
For :

****

For :



Difference from using at the same . Blue: , orange: .



**(iii) Crank-Nicolson Method**

From

Numerical scheme:

Identify the tridiagonal matrix system as

Thomas algorithm: For the system

Denoting

The solution is then by back substitution:

**Code:**

J = 20;

dx = 0.05;

N = 51;

dt = 0.0012; %[0.0012, 0.0013, 0.012];

x = linspace(0,1,J+1);

gd = zeros(N+1,J+1);

u0 = @(x) (x<=0.5).\*(2\*x) + (x>0.5).\*(2-2\*x);

gd(1,:) = u0(x);

mu = dt/dx^2;

a = -mu/2;

b = (1+mu);

c = -mu/2;

ci = zeros(J-2,1);

ci(1) = c/b;

for j=2:J-2

ci(j) = c/(b-a\*ci(j-1));

end

d = zeros(J-1,1);

di = zeros(J-1,1);

for n=1:N

for j=2:J

d(j-1) = mu/2\*gd(n,j-1) + (1-mu)\*gd(n,j) + mu/2\*gd(n,j+1);

end

di(1) = d(1)/b;

for j=2:J-1

di(j) = (d(j)-a\*di(j-1))/(b-a\*ci(j-1));

end

gd(n+1,J) = di(J-1);

for j=2:J-1

gd(n+1,J-j+1) = di(J-j) - ci(J-j)\*gd(n+1,J-j+2);

end

end

time = [1, 2, 26, 51];

for i = 1:4

subplot(4,1,i);

plot(x\*J, gd(time(i),:))

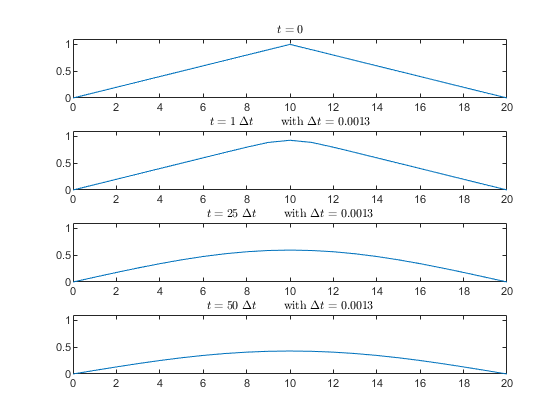
ylim([0, 1.1])

tt = {'$$t=', num2str(time(i)-1),'$$', '$$\Delta t\quad\quad$$ with $$\Delta t=$$', num2str(dt)};

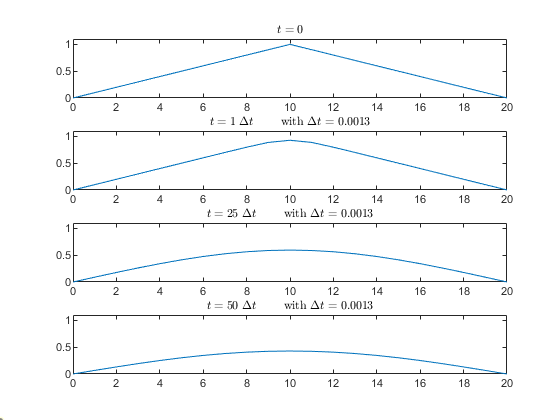
title(join(tt(1:3+(i>1)\*2)),'interpreter','latex');

end

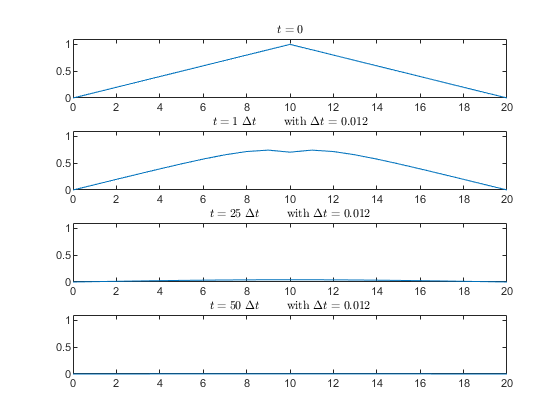
For :



For :



For :



**Q2**

The exact solution should be . Therefore at the discontinuity should appear at .

**Code for upwind method:**

dt = 0.0025; %[0.01, 0.0025]

N = int32(0.5/dt);

dx = dt\*2; % nu=dt/dx=0.5

J = int32(2/dx);

a = 1;

% a=1 > 0 so should use scheme (2) in lecture notes

x = linspace(-1,1,J+1);

u0 = @(x) (x<=0).\*1;

gd = u0(x);

for n = 1:N

for j = J:-1:2

gd(j) = gd(j) - a\*dt/dx\*(gd(j)-gd(j-1));

end

end

% exact solution

u = @(x,t) u0(x-t);

plot(x,gd(:),'b');

hold on

plot(x,u(x,0.5))

hold off

**Code for Lax Wendroff method:**

dt = 0.01; %[0.01, 0.0025]

N = int32(0.5/dt);

dx = dt\*2; % nu=dt/dx=0.5

J = int32(2/dx);

a = 1;

x = linspace(-1,1,J+1);

u0 = @(x) (x<=0).\*1;

gd = u0(x);

gd\_temp = gd;

for n = 1:N

for j = 2:J

gd\_temp(j) = gd(j) - a\*nu/2\*(gd(j+1)-gd(j-1)) + a^2\*nu^2/2\*(gd(j+1)-2\*gd(j)+gd(j-1));

end

gd = gd\_temp;

end

% exact solution

u = @(x,t) u0(x-t);

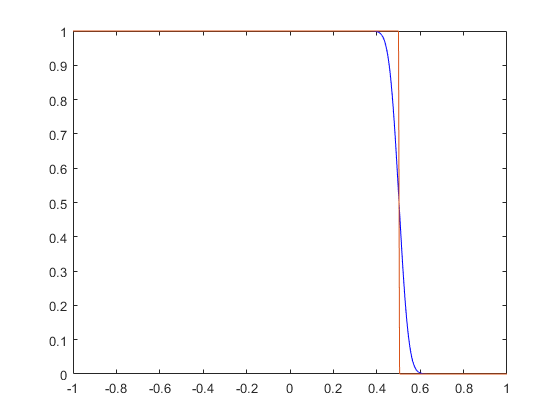
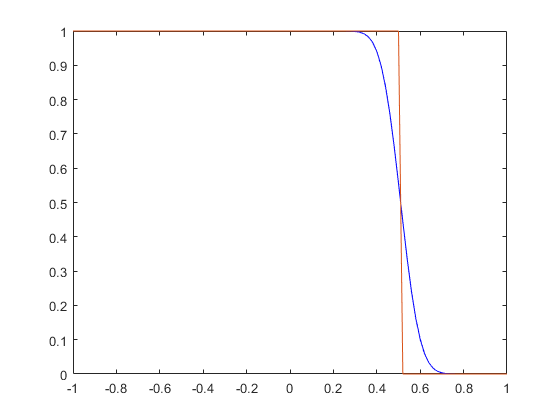
plot(x,gd(:),'b');

hold on

plot(x,u(x,0.5))

hold off

Upwind method, (left) and (right)



Lax-Wendroff method, (left) and (right)

