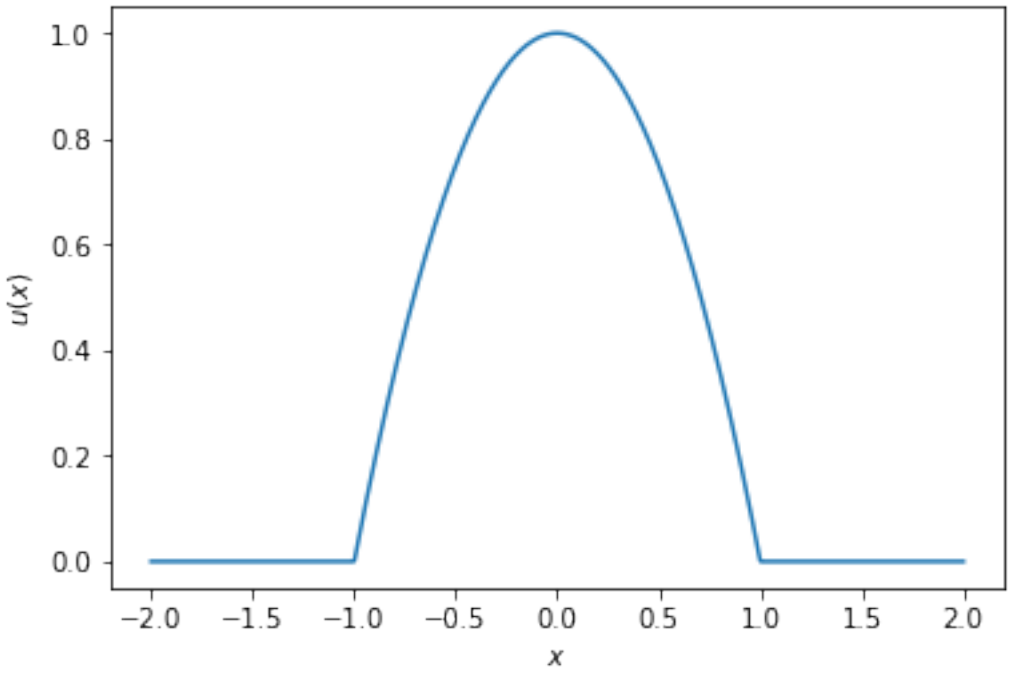
FD_schemes

February 4, 2018

1 Transport equation

$$u_t + a u_x = 0 \tag{1}$$
$$u(x,0) = u_0(x) \tag{2}$$
$$u(0,t) = 0 \tag{3}$$

Following are the implementations of various finite difference schemes for the above transport equation given the following initial condition



2 Finite difference schemes

2.1 FT-BS

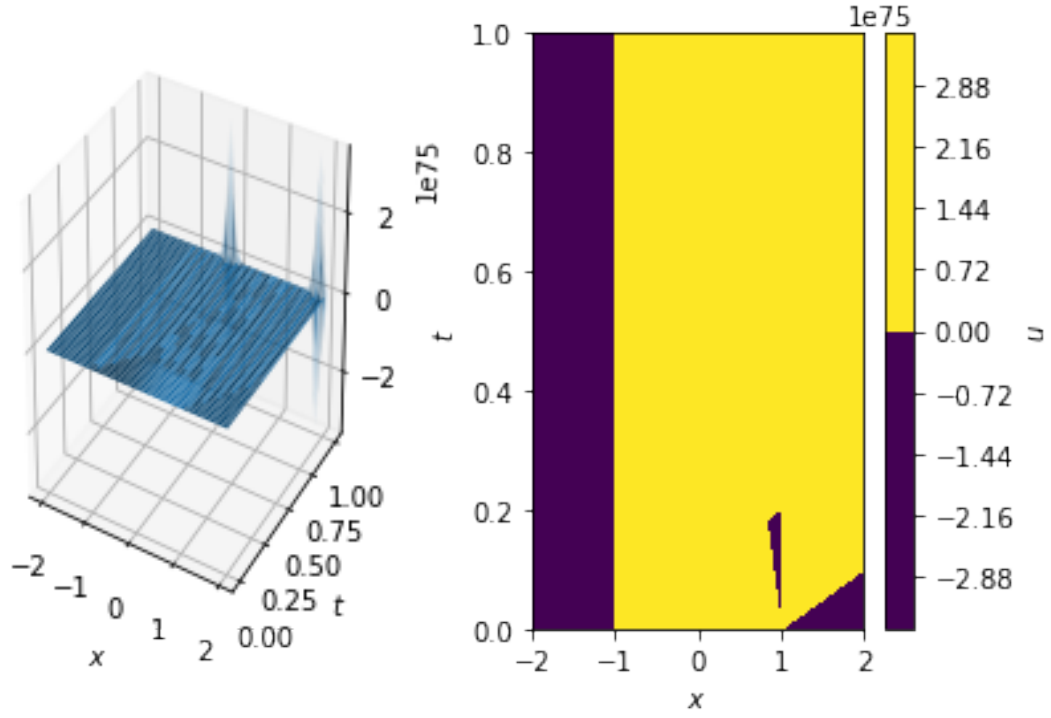
These are the results obtained by implementing the forward time backward space scheme.

$$U_j^{k+1} = U_j^k - \frac{a\Delta t}{\Delta x}\left(U_j^k - U_{j-1}^k\right) \tag{4}$$

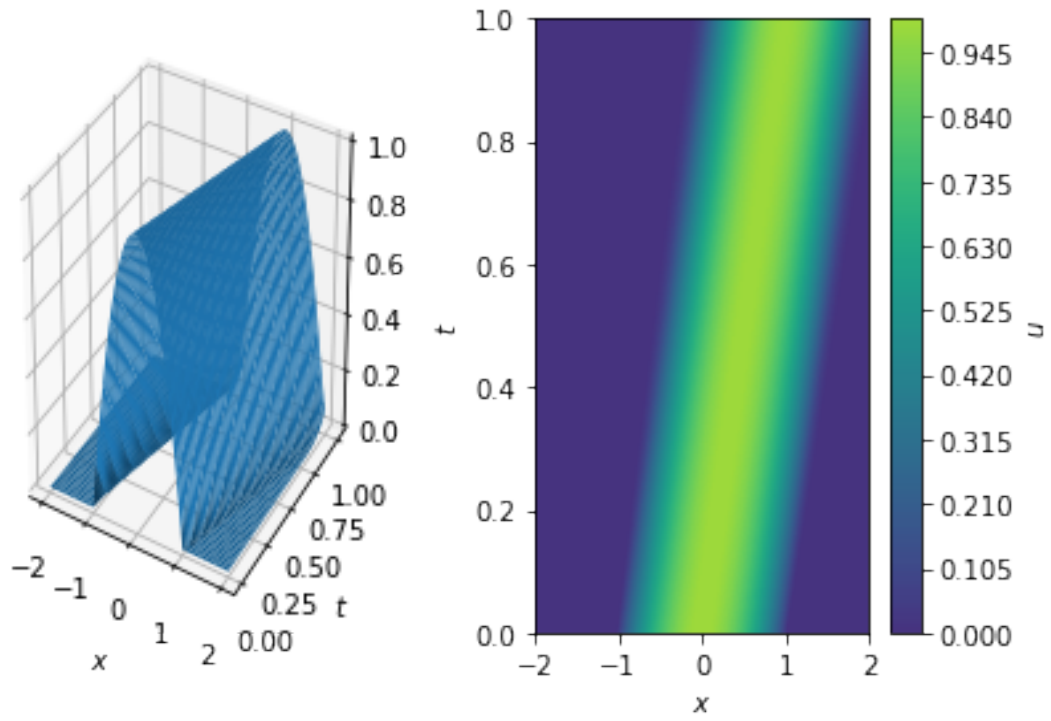
We will first run it with the following parameters:

$$dt = 1 \times 10^{-3} \tag{5}$$
$$a = -1 \tag{6}$$
$$dx = 1 \times 10^{-2} \tag{7}$$
$$\left|\frac{a\Delta t}{\Delta x}\right| = 0.1 \tag{8}$$

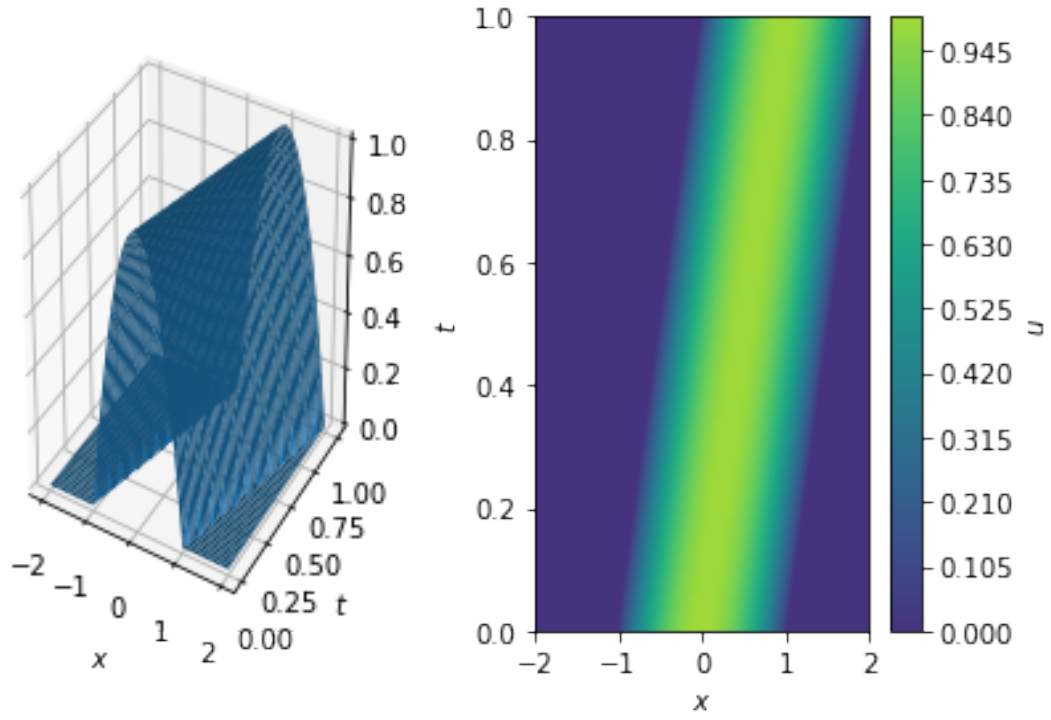
As expected, the solution blows up. This is becausethe flow of information is opposite to the direction of the wave velocity. The solution should depend on the value of u to the right of the grid point.



We now set $a = 1.0$ The CFL condition is satisfied and the scheme is stable

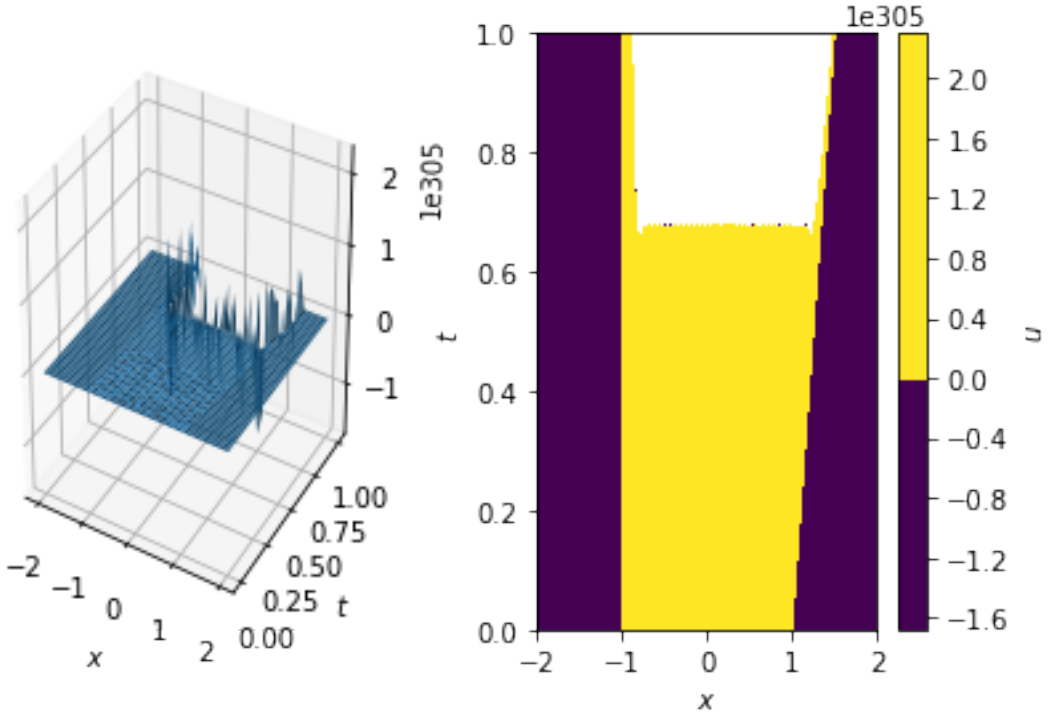


We now set $\Delta x = 1 \times 10^{-3}$ (The courant number becomes 1)



We now set $\Delta x = 5 \times 10^{-4}$ (The courant number becomes 2). The solution blows up as expected since the CFL condition is violated.

```
/home/foobar/code/ma540/solver.py:52: RuntimeWarning: overflow encountered in true_divide
  self.u_x[:,i] = ( self.u[:,i] - np.roll(self.u[:,i],1) ) / self.dx
/home/foobar/code/ma540/solver.py:69: RuntimeWarning: invalid value encountered in subtract
  self.u[:,i+1] = self.u[:,i] - self.a*(self.dt)*self.u_x[:,i]
/home/foobar/code/ma540/solver.py:52: RuntimeWarning: invalid value encountered in subtract
  self.u_x[:,i] = ( self.u[:,i] - np.roll(self.u[:,i],1) ) / self.dx
/usr/local/lib/python3.5/dist-packages/mpl_toolkits/mplot3d/axes3d.py:1691: RuntimeWarning: inv
  avgzsum = sum(p[2] for p in ps2)
/usr/local/lib/python3.5/dist-packages/mpl_toolkits/mplot3d/proj3d.py:73: RuntimeWarning: overi
  return np.sqrt(v[0]**2+v[1]**2+v[2]**2)
/usr/local/lib/python3.5/dist-packages/mpl_toolkits/mplot3d/axes3d.py:1758: RuntimeWarning: inv
  for n in normals])
/usr/local/lib/python3.5/dist-packages/mpl_toolkits/mplot3d/proj3d.py:141: RuntimeWarning: inva
  txs, tys, tzs = vecw[0]/w, vecw[1]/w, vecw[2]/w
```



Thus, the FTBS scheme is stable only when the wavespeed is positive and the cfl condition is satisfied

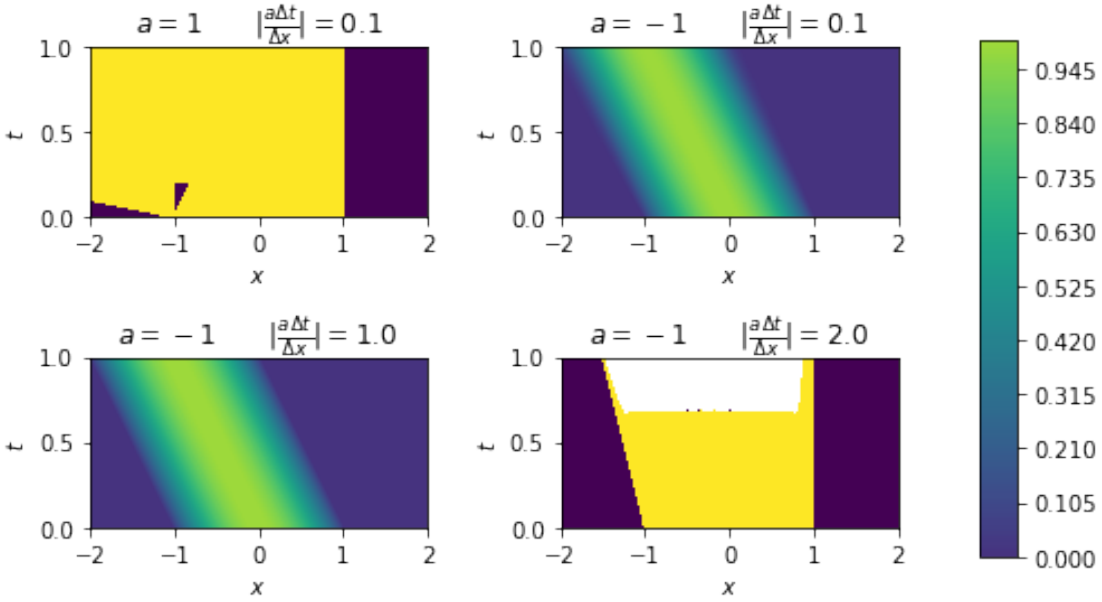
2.2 FT-FS

These are the results obtained by implementing the forward time-forward space scheme.

$$U_j^{k+1} = U_j^k - \frac{a\Delta t}{\Delta x}\left(U_{j+1}^k - U_j^k\right) \tag{9}$$

We run a similar set of simulations to the ones done for the FT-BS scheme

```
/home/foobar/code/ma540/solver.py:57: RuntimeWarning: overflow encountered in true_divide
  self.u_x[:,i] = (-1*self.u[:,i]+np.roll(self.u[:,i],-1))/self.dx
/home/foobar/code/ma540/solver.py:69: RuntimeWarning: invalid value encountered in subtract
  self.u[:,i+1] = self.u[:,i] - self.a*(self.dt)*self.u_x[:,i]
/usr/local/lib/python3.5/dist-packages/matplotlib/figure.py:1999: UserWarning: This figure incl
  warnings.warn("This figure includes Axes that are not compatible "
```



As expected, the solution is stable for $a < 0$ and $c < 1$

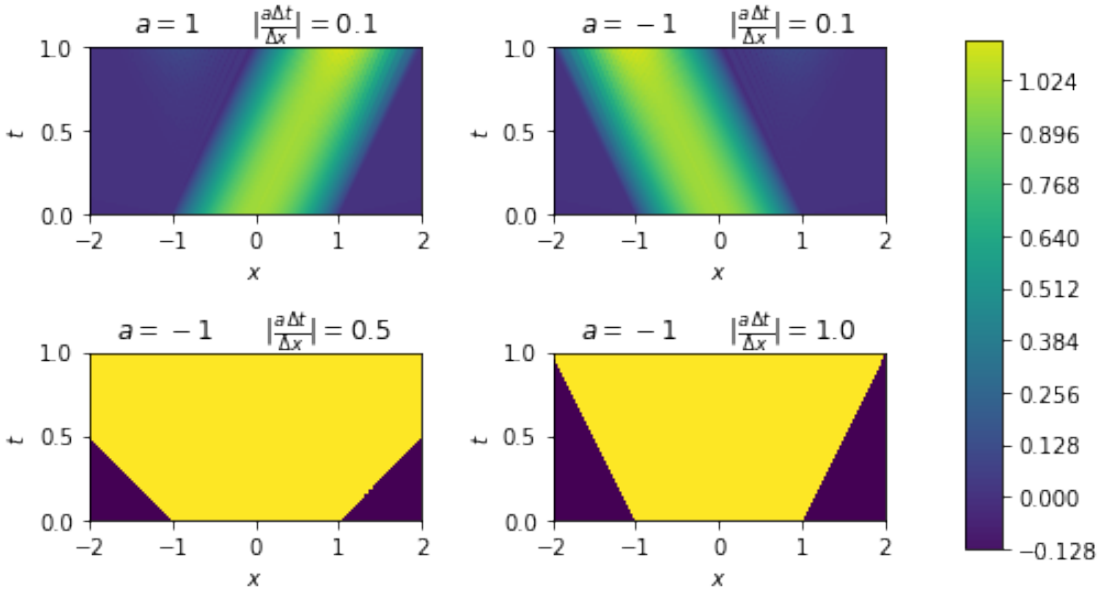
2.3 FT-CS

These are the results obtained by implementing the forward time-central space scheme.

$$U_j^{k+1} = U_j^k - \frac{a\Delta t}{2\Delta x}\left(U_{j+1}^k - U_{j-1}^k\right) \tag{10}$$

We run the similar set of simulations.

```
/usr/local/lib/python3.5/dist-packages/matplotlib/figure.py:1999: UserWarning: This figure incl
  warnings.warn("This figure includes Axes that are not compatible "
```



This scheme is unconditionall unstable. While it 'works' for both positive and negative values for wavespeed, the scheme blows up as time progresses. Even when the courant number is 0.1 we can see that the height of the peak is rising with time. It will eventually blow up

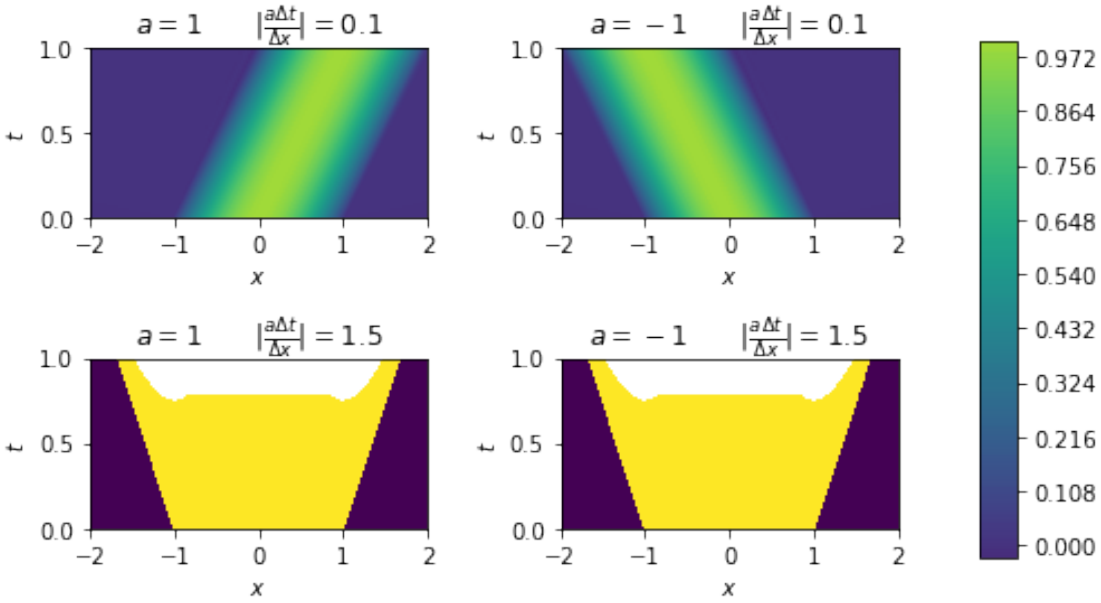
2.4 Leapfrog scheme

These are the results obtained by implementing the central time-central space scheme.

$$U_j^{k+1} = U_j^{k-1} - \frac{a\Delta t}{\Delta x}\left(U_{j+1}^k - U_{j-1}^k\right) \tag{11}$$

We run a similar set of simulations.

```
/home/foobar/code/ma540/solver.py:62: RuntimeWarning: overflow encountered in true_divide
  self.u_x[:,i] = (np.roll(self.u[:,i],-1) - np.roll(self.u[:,i],1))/(2*self.dx)
/home/foobar/code/ma540/solver.py:73: RuntimeWarning: invalid value encountered in subtract
  self.u[:,i+1] = self.u[:,i-1] - 2*self.a*(self.dt)*self.u_x[:,i]
/home/foobar/code/ma540/solver.py:62: RuntimeWarning: invalid value encountered in subtract
  self.u_x[:,i] = (np.roll(self.u[:,i],-1) - np.roll(self.u[:,i],1))/(2*self.dx)
/usr/local/lib/python3.5/dist-packages/matplotlib/figure.py:1999: UserWarning: This figure incl
  warnings.warn("This figure includes Axes that are not compatible "
```



Leapfrog scheme is thus stable(in both directions) when the CFL condition is satisfied. The scheme is unstable when the CFL condition is violated

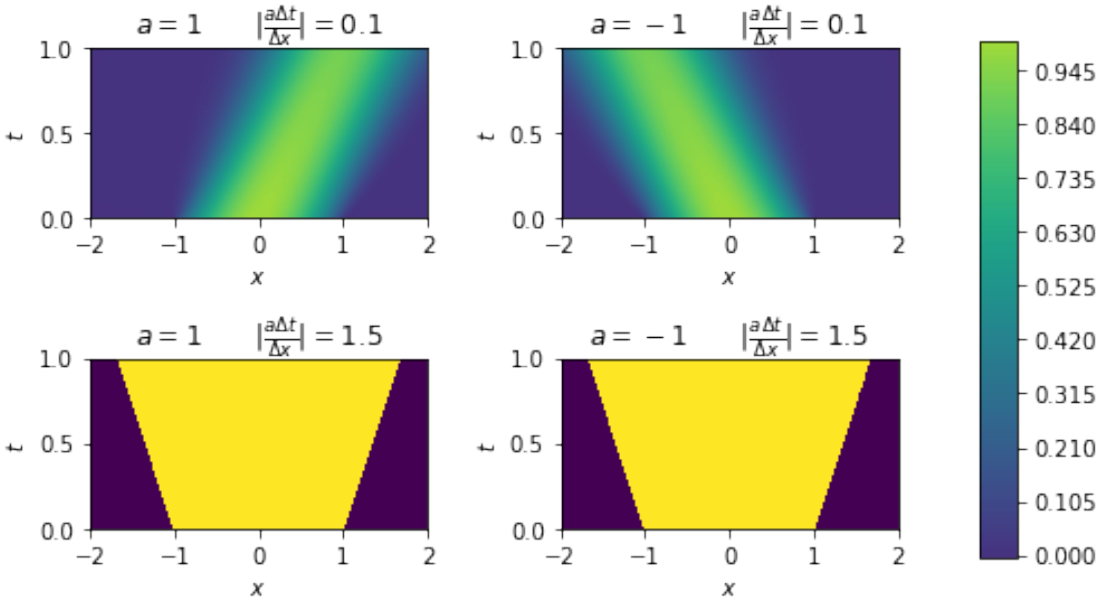
2.5 Lax Friedrich scheme

These are the results obtained by implementing the Lax-Friedrich scheme.

$$U_j^{k+1} = \frac{U_{j-1}^k + U_{j-1}^k}{2} - \frac{a\Delta t}{2\Delta x}\left(U_{j+1}^k - U_{j-1}^k\right) \tag{12}$$

We run a similar set of simulations.

```
/usr/local/lib/python3.5/dist-packages/matplotlib/figure.py:1999: UserWarning: This figure incl
  warnings.warn("This figure includes Axes that are not compatible "
```



Lax-Friedrich scheme is thus stable(in both directions) when the CFL condition is satisfied. The scheme is unstable when the CFL condition is violated.

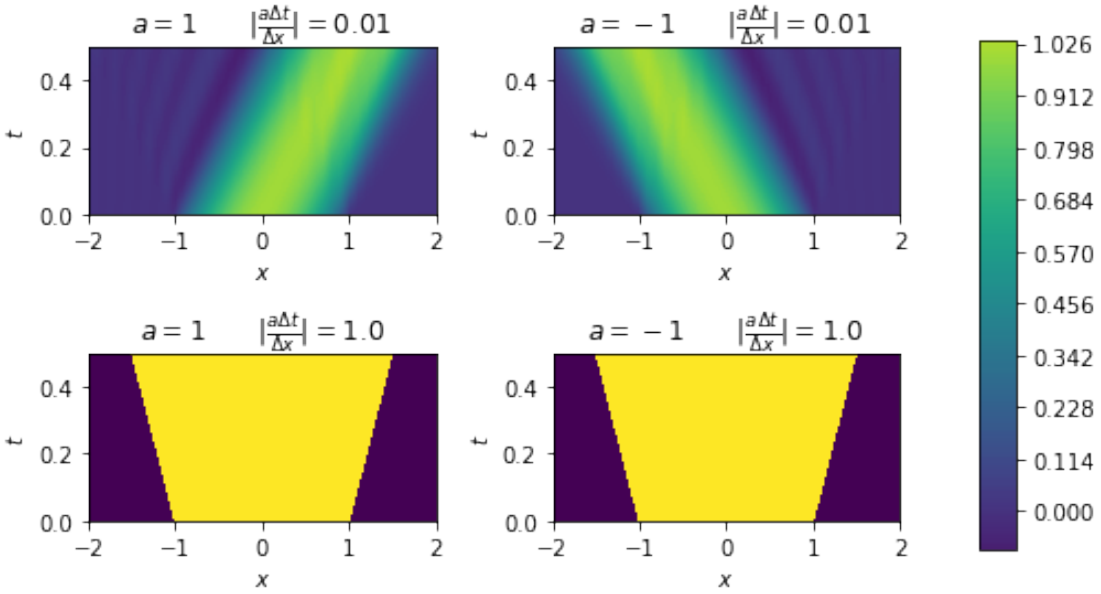
2.6 Lax Wendroff scheme

These are the results obtained by implementing the Lax-Wendroff scheme.

$$U_j^{k+1} = \left[1 - \left(\frac{a\Delta t}{\Delta x}\right)^2\right] U_j^k + \left[\frac{a\Delta t}{\Delta x} + \frac{1}{2}\left(\frac{a\Delta t}{\Delta x}\right)^2\right] U_{j-1}^k + \left[-\frac{a\Delta t}{\Delta x} + \frac{1}{2}\left(\frac{a\Delta t}{\Delta x}\right)^2\right] U_{j+1}^k \qquad (13)$$

We run a similar set of simulations.

```
/usr/local/lib/python3.5/dist-packages/matplotlib/figure.py:1999: UserWarning: This figure incl
  warnings.warn("This figure includes Axes that are not compatible "
```



This scheme also works for wave velocities in both directions given the CFL number is satisfied