Contents

1 1D Advection Equation

1.1 Description

Consider the 1D advection equation,

$$u_t + au_x = 0$$

defined on the domain $[a, b]$

With periodic boundary conditions, that is,

$$u(x + 1, t) = u(x, t)$$

1.2 Discretization

Partition the domain into N points, such that

$$x_i = i\Delta x$$

where, Δx is the partition size defined as,

$$\Delta x = \frac{b - a}{N - 1}$$

The time is partitioned so that,

$$t_n = n\Delta t$$

Where Δt is the time interval

Let U_i^n be the approximation to the function u, that is,

$$U_i^n \approx u(x_i, t_n)$$

By the finite difference method the time derivative is approximated using a forward difference scheme.

$$\left.\frac{\partial}{\partial t} u(x_i, t)\right|_{t=t_n} \Rightarrow \frac{U_i^{n+1} - U_i^n}{\Delta t}$$

The space derivative can be approximated in three different schemes:

1.2.1 Forward Difference

1. Description

$$\left.\frac{\partial}{\partial x} u(x, t_n)\right|_{x=x_i} \Rightarrow \frac{U_{i+1}^{n+1} - U_i^n}{\Delta x}$$

Thus we get,

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} + a\frac{U_{i+1}^{n+1} - U_i^n}{\Delta x} = 0$$

Define,

$$\sigma = \frac{a\Delta t}{\Delta x}$$

Rearranging we get,

$$U_i^{n+1} = (1+\sigma)\, U_i^n - \sigma U_{i+1}^n$$

where,

$$i = 0, 1, \ldots, N-2 \qquad n = 0, 1, 2, \ldots$$

With the periodic boundary condition we define,

$$U_{N-1}^{n+1} = (1+\sigma)\, U_{N-1}^n - \sigma U_1^n$$

2. Stability

Using Fourier Analysis we get, the solution

$$U_i^n = \beta^n e^{ilx_i}$$

Substituting in the scheme we get,

$$\beta^{n+1} e^{ilx_i} = (1+\sigma)\, \beta^n e^{ilx_i} - \sigma \beta^n e^{ilx_{i+1}}$$

Thus,

$$\beta = 1 + \sigma - \sigma e^{il\Delta x}$$

Clearly, $|\beta| > 1$. Hence, the forward difference scheme is unconditionally unstable.

1.2.2 Backward Difference

1. Description

$$\frac{\partial}{\partial x}u(x,t_n)\bigg|_{x=x_i} \Rightarrow \frac{U_i^{n+1}-U_{i-1}^n}{\Delta x}$$

   Thus we get,

$$\frac{U_i^{n+1}-U_i^n}{\Delta t} + a\frac{U_i^{n+1}-U_{i-1}^n}{\Delta x} = 0$$

   Define,

$$\sigma = \frac{a\Delta t}{\Delta x}$$

   Rearranging we get,

$$U_i^{n+1} = (1-\sigma)\,U_i^n + \sigma U_{i-1}^n$$

   where,

$$i = 1,2,\ldots,N-1 \qquad n = 0,1,2,\ldots$$

   With the periodic boundary condition we define,

$$U_0^{n+1} = (1-\sigma)\,U_0^n + \sigma U_{N-2}^n$$

2. Stability

   Using Fourier Analysis we get, the solution

$$U_i^n = \beta^n e^{ilx_i}$$

   Substituting in the scheme we get,

$$\beta^{n+1}e^{ilx_i} = (1-\sigma)\,\beta^n e^{ilx_i} + \sigma\beta^n e^{ilx_{i-1}}$$

   Thus,

$$\beta = 1-\sigma+\sigma e^{-il\Delta x}$$

$$\beta = 1-\sigma+\sigma cos(l\Delta x) - i\sigma sin(l\Delta x)$$

$$|\beta| = 1+2\sigma\,(\sigma-1)\,(1-cos(l\Delta x))$$

   $|\beta| < 1$ only when $\sigma < 1$. Hence the bacward difference scheme is stable only when $\sigma < 1$.

1.2.3 Central Difference

1. Description

$$\frac{\partial}{\partial x}u(x,t_n)\bigg|_{x=x_i} \Rightarrow \frac{U_{i+1}^{n+1}-U_{i-1}^{n}}{\Delta x}$$

   Thus we get,

$$\frac{U_i^{n+1}-U_i^n}{\Delta t}+a\frac{U_{i+1}^{n+1}-U_{i-1}^{n}}{\Delta x}=0$$

   Define,

$$\sigma=\frac{a\Delta t}{\Delta x}$$

   Rearranging we get,

$$U_i^{n+1}=U_i^n-\frac{\sigma}{2}U_{i+1}^n+\frac{\sigma}{2}U_{i-1}^n$$

   where,

$$i=1,2,\ldots,N-2 \qquad n=0,1,2,\ldots$$

   With the periodic boundary condition we define,

$$U_0^{n+1}=U_0^n-\frac{\sigma}{2}U_1^n+\frac{\sigma}{2}U_{N-2}^n$$

$$U_{N-1}^{n+1}=U_{N-1}^n-\frac{\sigma}{2}U_1^n+\frac{\sigma}{2}U_{N-2}^n$$

2. Stability

   Using Fourier Analysis we get, the solution

$$U_i^n=\beta^n e^{ilx_i}$$

   Substituting in the scheme we get,

$$\beta^{n+1}e^{ilx_i}=\beta^n e^{ilx_i}-\frac{\sigma}{2}\beta^n e^{ilx_{i+1}}+\frac{\sigma}{2}\beta^n e^{ilx_{i-1}}$$

   Thus,

$$\beta=1-\sigma/2e^{il\Delta x}+\sigma/2e^{-il\Delta x}$$
$$|\beta|=1+\sigma^2\sin^2(l\Delta x)$$

   Clearly, $|\beta|>1$. Hence, the central difference scheme is unconditionally unstable.

   But β is close to one if $\sigma<1$, meaning the solution slowly grows in amplitude for $\sigma<1$

1.3 Implementation

```
#include <iostream>
#include <vector>
```

1.3.1 Forward Difference

```
// This method returns the current state of the solution after n time steps
vector<double> nsol(const int n, const double sigma, const vector<double> &u)
{
    int size = u.size();
    vector<double> u_prev(size,0.0);
    u_prev = u;
    vector<double> u_next(size,0.0);
    for(unsigned int k=0;k<n;k++)
    {
        u_next[size-1] = (1+sigma)*u_prev[size-1] - sigma*u_prev[1];
        for(unsigned int i=0;i<size-1;i++)
        {
            u_next[i] = (1+sigma)*u_prev[i] - sigma*u_prev[i+1];
        }
        u_prev = u_next;
    }
    return u_next;
}
```

1.3.2 Backward Difference

```
// This method returns the current state of the solution after n time steps
vector<double> nsol(const int n, const double sigma, const vector<double> &u)
{
    int size = u.size();
    vector<double> u_prev(size,0.0);
    u_prev = u;
    vector<double> u_next(size,0.0);
    for(unsigned int k=0;k<n;k++)
    {
        u_next[0] = (1-sigma)*u_prev[0] + sigma*u_prev[size-2];
        for(unsigned int i=1;i<size;i++)
        {
            u_next[i] = (1-sigma)*u_prev[i] + sigma*u_prev[i-1];
```

```
        }
        u_prev = u_next;
    }
    return u_next;
}
```

1.3.3 Central Difference

```
// This method returns the current state of the solution after n time steps
vector<double> nsol(const int n, const double sigma, const vector<double> &u)
{
    int size = u.size();
    vector<double> u_prev(size,0.0);
    u_prev = u;
    vector<double> u_next(size,0.0);
    for(unsigned int k=0;k<n;k++)
    {
        u_next[size-1] = u_prev[size-1] - (sigma/2)*u_prev[1] + (sigma/2)*u_prev[size-
        u_next[0] = u_prev[0] - (sigma/2)*u_prev[1] + (sigma/2)*u_prev[size-2];
        for(unsigned int i=1;i<size-1;i++)
        {
            u_next[i] = u_prev[i] - (sigma/2)*u_prev[i+1] + (sigma/2)*u_prev[i-1];
        }
        u_prev = u_next;
    }
    return u_next;
}
```