M.Sc. in Computational Science

Fundamentals of Atmospheric Modelling

Peter Lynch, Met Éireann

Mathematical Computation Laboratory (Opp. Room 30)
Dept. of Maths. Physics, UCD, Belfield.

January–April, 2004.

Lecture 11

Time Integration Methods

Introduction

- We have studied various simple solutions of the shallow water equations by making approximations.
- In particular, by means of the perturbation method the equations have been linearised, making them amenable to analytical investigation.
- However, to obtain solutions in the general case, it is necessary to solve the full nonlinear system.
- In numerical weather prediction (NWP) the fully nonlinear primitive equations are solved by numerical means.
- In the atmosphere, the nonlinear advection process is a dominant factor.
- To get some idea of the methods used, we look at the simple problem of formulating time-integration algorithms for the solution of the simple advection equation.

- The development of accurate and efficient numerical schemes is a huge area of ongoing research, and we can do no more than to introduce the subject.
- Much of our research effort in Met Éireann over the past ten years has been in this area, and a brief outline of this work will be given.

Discretization Methods

There are several distinct approaches to the formulation of computer methods for solving differential equations. We will confine ourselves to the *finite difference methods*.

Other approaches include *finite element* method and the *spectral* method.

Discretization Methods

There are several distinct approaches to the formulation of computer methods for solving differential equations. We will confine ourselves to the *finite difference methods*.

Other approaches include *finite element* method and the *spectral* method.

The central idea of the finite difference approach is to *approximate the derivatives* in the equation by differences between adjacent points in space or time, and thereby reduce the differential equation to a difference equation.

- An *analytical* problem becomes an *algebraic* one.
- A problem with an *infinite* degree of freedom is replaced by one with a *finite* degree of freedom.
- A *continuous* problem goes over to a *discrete* one.

The Finite Difference Method

We start by looking at the *Taylor expansion* of $f(x)$:

$$f(x + \Delta x) = f(x) + f'(x).\Delta x + \tfrac{1}{2}f''(x)\Delta x^2 + [O(\Delta x^3)] \quad (1)$$

$$f(x - \Delta x) = f(x) - f'(x).\Delta x + \tfrac{1}{2}f''(x)\Delta x^2 + [O(\Delta x^3)] \quad (2)$$

The higher order terms, represented by $O(\Delta x^3)$, become less important as Δx becomes smaller.

The Finite Difference Method

We start by looking at the *Taylor expansion* of $f(x)$:

$$f(x+\Delta x) = f(x) + f'(x).\Delta x + \tfrac{1}{2}f''(x)\Delta x^2 + [O(\Delta x^3)] \qquad (1)$$

$$f(x-\Delta x) = f(x) - f'(x).\Delta x + \tfrac{1}{2}f''(x)\Delta x^2 + [O(\Delta x^3)] \qquad (2)$$

The higher order terms, represented by $O(\Delta x^3)$, become less important as Δx becomes smaller.
We can neglect these and use (1) or (2) to get an approximation for the derivative of $f(x)$ as follows:

$$f'(x) = \frac{f(x+\Delta x) - f(x)}{\Delta x} + O(\Delta x) = f_F + O(\Delta x)$$

$$f'(x) = \frac{f(x) - f(x-\Delta x)}{\Delta x} + O(\Delta x) = f_B + O(\Delta x)\,.$$

These are called the forward and backward differences.

The Finite Difference Method

We start by looking at the *Taylor expansion* of $f(x)$:

$$f(x+\Delta x) = f(x) + f'(x).\Delta x + \tfrac{1}{2}f''(x)\Delta x^2 + [O(\Delta x^3)] \qquad (1)$$

$$f(x-\Delta x) = f(x) - f'(x).\Delta x + \tfrac{1}{2}f''(x)\Delta x^2 + [O(\Delta x^3)] \qquad (2)$$

The higher order terms, represented by $O(\Delta x^3)$, become less important as Δx becomes smaller.
We can neglect these and use (1) or (2) to get an approximation for the derivative of $f(x)$ as follows:

$$f'(x) = \frac{f(x+\Delta x) - f(x)}{\Delta x} + O(\Delta x) = f_F + O(\Delta x)$$

$$f'(x) = \frac{f(x) - f(x-\Delta x)}{\Delta x} + O(\Delta x) = f_B + O(\Delta x)\,.$$

These are called the forward and backward differences.
Keeping only leading terms, we incur errors of order $O(\Delta x)$.

***We can do better than this*: subtracting (2) from (1) yields the following:**

$$f'(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} + O(\Delta x^2) = f_C + O(\Delta x^2)$$

which is seen to be of order $O(\Delta x^2)$, therefore *more accurate* for small Δx.

***We can do better than this*: subtracting (2) from (1) yields the following:**

$$f'(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} + O(\Delta x^2) = f_C + O(\Delta x^2)$$

which is seen to be of order $O(\Delta x^2)$, therefore *more accurate* for small Δx.

Adding (1) and (2) gives the corresponding expression for the second derivative:

$$f''(x) = \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2} + O(\Delta x^2)$$

These centered differences are of accuracy $O(\Delta x^2)$.

***We can do better than this*: subtracting (2) from (1) yields the following:**

$$f'(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} + O(\Delta x^2) = f_C + O(\Delta x^2)$$

which is seen to be of order $O(\Delta x^2)$, therefore *more accurate* for small Δx.

Adding (1) and (2) gives the corresponding expression for the second derivative:

$$f''(x) = \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2} + O(\Delta x^2)$$

These centered differences are of accuracy $O(\Delta x^2)$.

We can continue taking more and more terms, but obviously there is a trade-off between accuracy and efficiency. Fourth-order accurate schemes are sometimes used in NWP, but *second order accuracy is more popular*.

Exercise:

Consider the function $f(x) = A\sin(2\pi x/L)$**.**
We know that the derivative is $(2\pi/L)A\cos(2\pi x/L)$**.**

Exercise:

Consider the function $f(x) = A\sin(2\pi x/L)$.
We know that the derivative is $(2\pi/L)A\cos(2\pi x/L)$.

- **Show that a forward difference approximation gives**

$$f'_F(x) = (2\pi/L)A\cos[2\pi(x + \Delta x/2)/L] \cdot \left[\frac{\sin(\pi\Delta x/L)}{\pi\Delta x/L}\right]$$

  whereas the centered difference yields:

$$f'_C(x) = (2\pi/L)A\cos[2\pi x/L] \cdot \left[\frac{\sin(2\pi\Delta x/L)}{2\pi\Delta x/L}\right]$$

- **Compare these to the true derivative $f'(x)$ and investigate their behaviour for small Δx.**
- **Demonstrate thus that the centered difference is of higher order accuracy.**

Grid Resolution and Accuracy

The size of the gridstep Δx determines the accuracy of the numerical scheme. For the simple sine function the error depended on the ratio $(\Delta x/L)$.

For synoptic scale waves in the atmosphere a typical value of L is 1000 km. To make the ratio equal to 0.1 we need to have a grid size of about 100 km. This is larger than the typical gridsizes used in operational NWP.

The higher the resolution, that is, the smaller the grid-size, the heavier the computational burden. There is a *trade-off* between resolution and accuracy.

Linear Computational Instability

We consider the equation describing the conservation of a quantity $Y(x,t)$ following the motion of a fluid flow in one space dimension:

$$\frac{dY}{dt} \equiv \left(\frac{\partial Y}{\partial t} + u \frac{\partial Y}{\partial x} \right) = 0 \,.$$

Linear Computational Instability

We consider the equation describing the conservation of a quantity $Y(x,t)$ following the motion of a fluid flow in one space dimension:

$$\frac{dY}{dt} \equiv \left(\frac{\partial Y}{\partial t} + u \frac{\partial Y}{\partial x} \right) = 0 \,.$$

If the velocity is taken to be constant, $u = c$, or if we linearise about a mean flow $\bar{u} = c$, the equation becomes

$$\frac{\partial Y}{\partial t} + c \frac{\partial Y}{\partial x} = 0 \,.$$

Linear Computational Instability

We consider the equation describing the conservation of a quantity $Y(x,t)$ following the motion of a fluid flow in one space dimension:

$$\frac{dY}{dt} \equiv \left(\frac{\partial Y}{\partial t} + u\frac{\partial Y}{\partial x}\right) = 0\,.$$

If the velocity is taken to be constant, $u = c$, or if we linearise about a mean flow $\bar{u} = c$, the equation becomes

$$\frac{\partial Y}{\partial t} + c\frac{\partial Y}{\partial x} = 0\,.$$

This simple equation we will call the *linear advection equation*. It is analogous to a factor of the usual wave equation

$$\left(\frac{\partial^2}{\partial t^2} - c^2\frac{\partial^2}{\partial x^2}\right) Y = \left(\frac{\partial}{\partial t} + c\frac{\partial}{\partial x}\right)\left(\frac{\partial}{\partial t} - c\frac{\partial}{\partial x}\right) Y = 0\,,$$

and the general solution is $Y = Y(x - ct)$.

Since the advection equation is linear, we can construct a general solution from Fourier components

$$Y = a \exp[ik(x - ct)]\,; \qquad k = 2\pi/L\,.$$

We take the following *initial condition* for Y:

$$Y(x, 0) = a \exp[ikx]$$

Since the advection equation is linear, we can construct a general solution from Fourier components

$$Y = a \exp[ik(x - ct)]\,; \qquad k = 2\pi/L\,.$$

We take the following *initial condition* for Y:

$$Y(x, 0) = a \exp[ikx]$$

Next, we approximate the differential equation by a finite difference equation using centered differences for the space and time derivatives.

Since the advection equation is linear, we can construct a general solution from Fourier components

$$Y = a \exp[ik(x - ct)]\,; \qquad k = 2\pi/L\,.$$

We take the following *initial condition* for Y:

$$Y(x, 0) = a \exp[ikx]$$

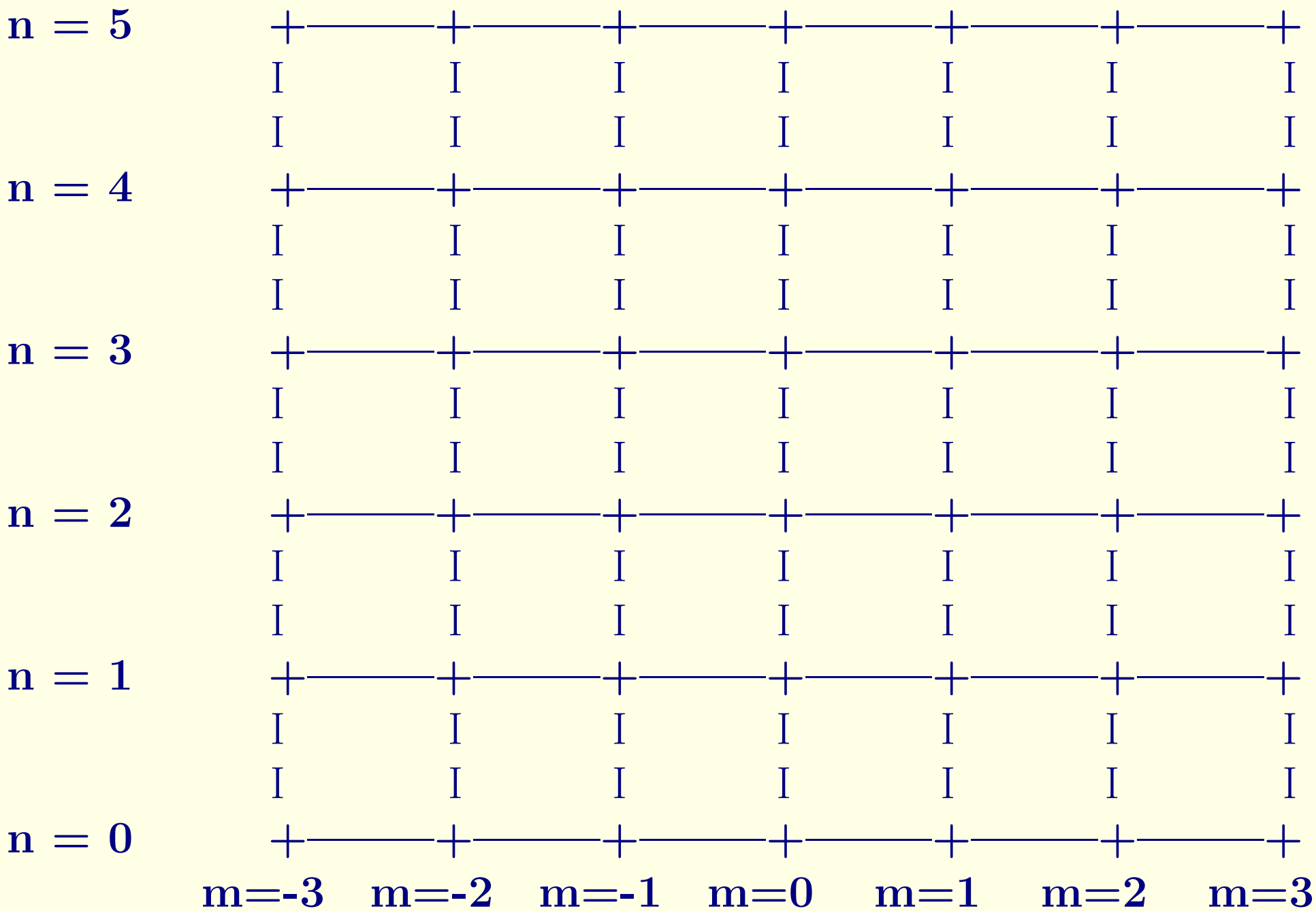
Next, we approximate the differential equation by a finite difference equation using centered differences for the space and time derivatives.

Let the variables x and t be represented by the horizontal and vertical axes. Positive time corresponds to the upper half plane. The initial data occur on the x-axis.

The continuous variables are replaced by discrete gridpoints at their integral values and the problem is solved on a finite difference grid.

Space-Time Grid:

Space axis horizontal
Time axis vertical



We denote the value of Y at a grid point by:

$$Y(m\Delta x, n\Delta t) = Y_m^n \,.$$

Then the finite difference approximation to the differential equation may be written as follows:

$$\left(\frac{Y_m^{n+1} - Y_m^{n-1}}{2\Delta t}\right) + c\left(\frac{Y_{m+1}^n - Y_{m-1}^n}{2\Delta x}\right) = 0 \,.$$

We denote the value of Y at a grid point by:

$$Y(m\Delta x, n\Delta t) = Y_m^n \,.$$

Then the finite difference approximation to the differential equation may be written as follows:

$$\left(\frac{Y_m^{n+1} - Y_m^{n-1}}{2\Delta t}\right) + c\left(\frac{Y_{m+1}^n - Y_{m-1}^n}{2\Delta x}\right) = 0 \,.$$

Solving for the value at time $(n+1)\Delta t$ gives

$$Y_m^{n+1} = Y_m^{n-1} - \left(\frac{c\Delta t}{\Delta x}\right)(Y_{m+1}^n - Y_{m-1}^n)$$

The value at the time $(n+1)\Delta t$ is obtained by adding a term to the value at $(n-1)\Delta t$; the method is known as the leapfrog method because of this jump over the time $n\Delta t$.

We denote the value of Y at a grid point by:

$$Y(m\Delta x, n\Delta t) = Y_m^n \,.$$

Then the finite difference approximation to the differential equation may be written as follows:

$$\left(\frac{Y_m^{n+1} - Y_m^{n-1}}{2\Delta t}\right) + c\left(\frac{Y_{m+1}^n - Y_{m-1}^n}{2\Delta x}\right) = 0 \,.$$

Solving for the value at time $(n+1)\Delta t$ gives

$$Y_m^{n+1} = Y_m^{n-1} - \left(\frac{c\Delta t}{\Delta x}\right)(Y_{m+1}^n - Y_{m-1}^n)$$

The value at the time $(n+1)\Delta t$ is obtained by adding a term to the value at $(n-1)\Delta t$; the method is known as the leapfrog method because of this jump over the time $n\Delta t$.
Note the ratio

$$\lambda \equiv \frac{c\Delta t}{\Delta x} \,.$$

The value of this will be found to be crucial.

Inter-dependency of Points

n + 1	+	**X**	+
n	**X**	**O**	**X**
n - 1	+	**X**	+
	m-1	**m**	**m+1**

The evaluation of the equation at point O involves values of the variable at points X. Solving for Y_m^{n+1} thus requires

$$Y_m^{n-1}, \qquad Y_{m-1}^n \qquad \textbf{and} \qquad Y_{m+1}^n .$$

Note how the leapfrog scheme *splits the grid into two independent sub-grids.*

Solving for the value at time $(n+1)\Delta t$ gives

$$Y_m^{n+1} = Y_m^{n-1} - \left(\frac{c\Delta t}{\Delta x}\right)(Y_{m+1}^n - Y_{m-1}^n)$$

Assuming that we know the solution up to time $n\Delta t$, the values at time $(n+1)\Delta t$ can be calculated, and the solution advanced one timestep in this way.
Then the whole procedure can be repeated to advance the solution to $(n+2)\Delta t$, and so on.

Solving for the value at time $(n+1)\Delta t$ gives

$$Y_m^{n+1} = Y_m^{n-1} - \left(\frac{c\Delta t}{\Delta x}\right)(Y_{m+1}^n - Y_{m-1}^n)$$

Assuming that we know the solution up to time $n\Delta t$, the values at time $(n+1)\Delta t$ can be calculated, and the solution advanced one timestep in this way.
Then the whole procedure can be repeated to advance the solution to $(n+2)\Delta t$, and so on.

Question: Under what conditions does the solution of the finite difference equation approximate that of the original differential equation?
Intuitively, we would expect that a good approximation would be obtained provided the grid steps Δx and Δt are small enough. However, it turns out that this is not enough, and that the value of the ratio $\lambda = c\Delta t/\Delta x$ is critical. This surprising result has important practical implications.

The CFL Stability Criterion

Let us assume a solution of the finite difference equation in the form

$$Y_m^n = a\, A^n \exp[ikm\Delta x]$$

The CFL Stability Criterion

Let us assume a solution of the finite difference equation in the form

$$Y_m^n = a\, A^n \exp[ikm\Delta x]$$

Substituting this in the equation and dividing by a common factor gives

$$A^2 + (2i\lambda \sin k\Delta x)A - 1 = 0$$

where $\lambda = c\Delta t/\Delta x$. This is a quadratic equation for the amplitude A, with solutions

$$A_\pm = -i\lambda \sin k\Delta x \pm \sqrt{1 - \lambda^2 \sin^2 k\Delta x}\,.$$

The CFL Stability Criterion

Let us assume a solution of the finite difference equation in the form

$$Y_m^n = a\, A^n \exp[ikm\Delta x]$$

Substituting this in the equation and dividing by a common factor gives

$$A^2 + (2i\lambda \sin k\Delta x)A - 1 = 0$$

where $\lambda = c\Delta t/\Delta x$. This is a quadratic equation for the amplitude A, with solutions

$$A_\pm = -i\lambda \sin k\Delta x \pm \sqrt{1 - \lambda^2 \sin^2 k\Delta x}\,.$$

The roots of a quadratic equation may be either real or complex, depending on the value of the coefficients.

We consider in turn the two possible cases.

Case I: $|\lambda| \leq 1$

The quantity under the square-root sign is positive, so the modulus of A is given by

$$|A|^2 = 1 - \lambda^2 \sin^2 k\Delta x + (\lambda \sin k\Delta x)^2 = 1\,.$$

The modulus of A is seen to be unity. Thus, we may write

$$A = \exp(i\psi)\,.$$

Case I: $|\lambda| \leq 1$

The quantity under the square-root sign is positive, so the modulus of A is given by

$$|A|^2 = 1 - \lambda^2 \sin^2 k\Delta x + (\lambda \sin k\Delta x)^2 = 1\,.$$

The modulus of A is seen to be unity. Thus, we may write

$$A = \exp(i\psi)\,.$$

The two values of the phase are

$$\psi_1 = -\arcsin(\lambda \sin k\Delta x)$$

and

$$\psi_2 = \pi - \psi_1\,.$$

For small λ, we have

$$\psi_1 \approx -\lambda k\Delta x = -kc\Delta t \qquad \textbf{and} \qquad \psi_2 \approx \pi + \lambda k\Delta x = \pi + kc\Delta t$$

Case I: $|\lambda| \leq 1$

The quantity under the square-root sign is positive, so the modulus of A is given by

$$|A|^2 = 1 - \lambda^2 \sin^2 k\Delta x + (\lambda \sin k\Delta x)^2 = 1\,.$$

The modulus of A is seen to be unity. Thus, we may write

$$A = \exp(i\psi)\,.$$

The two values of the phase are

$$\psi_1 = -\arcsin(\lambda \sin k\Delta x)$$

and

$$\psi_2 = \pi - \psi_1\,.$$

For small λ, we have

$$\psi_1 \approx -\lambda k\Delta x = -kc\Delta t \qquad \textbf{and} \qquad \psi_2 \approx \pi + \lambda k\Delta x = \pi + kc\Delta t$$

The solution of the equation may now be written

$$Y_m^n = \Big[D\exp(i\psi_1 n) + E\exp[i(-\psi_1 + \pi)n]\Big]\exp(ikm\Delta x)$$

where D and E are arbitrary constants.

Taking account of the initial conditions, we get the solution

$$Y_m^n = \underbrace{(a-E)\exp[ik(m\Delta x + \psi_1 n/k)]}_{\text{Physical Mode}} + \underbrace{(-1)^n E \exp[ik(m\Delta x - \psi_1 n/k)]}_{\text{Computational Mode}}$$

Exercise: Check the algebra leading to this solution.

Taking account of the initial conditions, we get the solution

$$Y_m^n = \underbrace{(a-E)\exp[ik(m\Delta x + \psi_1 n/k)]}_{\text{Physical Mode}} + \underbrace{(-1)^n E \exp[ik(m\Delta x - \psi_1 n/k)]}_{\text{Computational Mode}}$$

Exercise: Check the algebra leading to this solution.

The first term of this solution is called the physical mode and corresponds to the solution of the differential equation.

The second term is called the computational mode. It arises through the use of centered differences resulting in the approximation of a first order differential equation by a second order difference equation (with an extra solution).

Taking account of the initial conditions, we get the solution

$$Y_m^n = \underbrace{(a-E)\exp[ik(m\Delta x + \psi_1 n/k)]}_{\text{Physical Mode}} + \underbrace{(-1)^n E \exp[ik(m\Delta x - \psi_1 n/k)]}_{\text{Computational Mode}}$$

Exercise: Check the algebra leading to this solution.

The first term of this solution is called the physical mode and corresponds to the solution of the differential equation.

The second term is called the computational mode. It arises through the use of centered differences resulting in the approximation of a first order differential equation by a second order difference equation (with an extra solution).

If the ratio λ is small, the finite difference solution is given approximately by

$$Y \approx a \exp[ik(m\Delta x - cn\Delta t)]$$

which is just the analytical solution. Thus, we would expect good results from the finite difference approximation in this case.

The centered difference approximation cannot be used for the first timestep because it would involve values at time $t = -\Delta t$ which are not known. Instead, we must use another approximation for the first step, typically an uncentered forward time step. Thereafter, the leapfrog scheme can be used.

The centered difference approximation cannot be used for the first timestep because it would involve values at time $t = -\Delta t$ which are not known. Instead, we must use another approximation for the first step, typically an uncentered forward time step. Thereafter, the leapfrog scheme can be used.

In essence, this amounts to specifing another *"initial condition"*, the computational initial condition, at $t = \Delta t$. The value of this determines the amplitude of the computational mode. It should be chosen to minimize this.

The centered difference approximation cannot be used for the first timestep because it would involve values at time $t = -\Delta t$ which are not known. Instead, we must use another approximation for the first step, typically an uncentered forward time step. Thereafter, the leapfrog scheme can be used.

In essence, this amounts to specifing another *"initial condition"*, the computational initial condition, at $t = \Delta t$. The value of this determines the amplitude of the computational mode. It should be chosen to minimize this.

The above solution implies

$$\begin{aligned} Y_m^0 &= a\exp(ikm\Delta x) \\ Y_m^1 &= [a\exp(i\psi_1) - 2E\cos\psi_1]\exp(ikm\Delta x) \end{aligned}$$

Requiring $E = 0$, we find that $Y_m^1 = \exp(i\psi_1)Y_m^0$. In this simple case, we can eliminate the computational mode. In general, it is much more difficult.

Case II: $|\lambda| > 1$

Recall that the roots of the quadratic are

$$A_{\pm} = -i\lambda \sin k\Delta x \pm \sqrt{1 - \lambda^2 \sin^2 k\Delta x}\,.$$

If $|\lambda| > 1$, there will be some wavelengths for which

$$\lambda^2 \sin^2 k\Delta x > 1\,.$$

Case II: $|\lambda| > 1$

Recall that the roots of the quadratic are

$$A_{\pm} = -i\lambda \sin k\Delta x \pm \sqrt{1 - \lambda^2 \sin^2 k\Delta x}\,.$$

If $|\lambda| > 1$, there will be some wavelengths for which

$$\lambda^2 \sin^2 k\Delta x > 1\,.$$

Then the two roots of the quadratic are pure imaginary

$$A = i\left(-\lambda \sin k\Delta x \pm \sqrt{\lambda^2 \sin^2 k\Delta x - 1}\right)$$

and therefore either $|A_+| > 1$ or $|A_-| > 1$, i.e., the modulus of one of the roots will exceed unity.

Case II: $|\lambda| > 1$

Recall that the roots of the quadratic are

$$A_{\pm} = -i\lambda \sin k\Delta x \pm \sqrt{1 - \lambda^2 \sin^2 k\Delta x}\,.$$

If $|\lambda| > 1$, there will be some wavelengths for which

$$\lambda^2 \sin^2 k\Delta x > 1\,.$$

Then the two roots of the quadratic are pure imaginary

$$A = i\left(-\lambda \sin k\Delta x \pm \sqrt{\lambda^2 \sin^2 k\Delta x - 1}\right)$$

and therefore either $|A_+| > 1$ or $|A_-| > 1$, i.e., the modulus of one of the roots will exceed unity.

In that case the amplitude of the corresponding solution will increase without limit as time increases. That is, the amplitude of the solution of the finite difference equation will grow without bound for large time.
This phenomenon is called computational instability.

In case of *computational instability*, the solution of the finite difference equation cannot possibly resemble the physical solution.

The physical solution remains of constant amplitude for all time. The numerical solution grows without limit with time.

We thus require that $|\lambda| \leq 1$.

In case of *computational instability*, the solution of the finite difference equation cannot possibly resemble the physical solution.

The physical solution remains of constant amplitude for all time. The numerical solution grows without limit with time.

We thus require that $|\lambda| \leq 1$.

The condition for stability is known as the CFL Criterion:

$$\frac{c\Delta t}{\Delta x} \leq 1$$

after Courant, Friedichs and Lewy (1928), who first published the result.

It implies that, if we refine the space grid, that is, decrease Δx, we must also shorten the time step Δt.

Thus, halving the grid size in a two dimensional domain results in an eightfold increase in computation time.

Unconditionally Stable Schemes.

A large part of the research effort in Met Éireann recently has been devoted to the development of integration schemes which are free of the CFL constraint. The semi-Lagrangian scheme for advection is based on the idea of approximating the Lagrangian form of the time derivative. It is so formulated that the numerical domain of dependence always includes the physical domain of dependence. This necessary condition for stability is satisfied automatically by the scheme. The semi-Lagrangian algorithm has enabled us to integrate the primitive equations using a time step of 15 minutes. This can be compared to a typical timestep of 2.5 minutes for conventional schemes. The consequential saving of computation time means that the operational numerical guidance is available to the forecasters much earlier than would otherwise be the case.

We discuss semi-Lagrangian schemes in the next lecture.