CHAPTER 2 Basic Concepts

2.1 Finite-difference quotients

Consider the derivative $\frac{du}{dx}$, where $u = u(x)$, and x is the independent variable (which could be either space or time). Finite-difference methods represent the continuous function $u(x)$ by a set of values defined at a number of discrete points in a specified region. Thus, we usually introduce a "grid" with discrete points at which the variable u is carried, as shown in Fig. 2.1. Sometimes the words "mesh or "lattice" are used in place of "grid." The

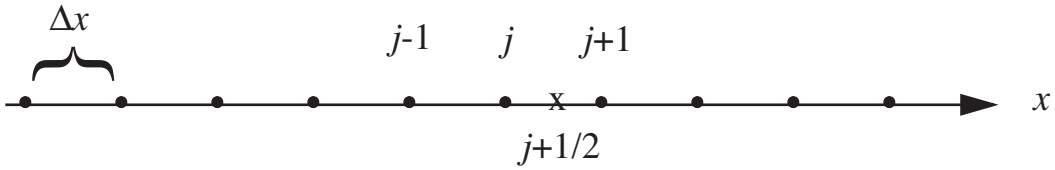

Figure 2.1: An example of a grid, with uniform grid spacing Δx. The grid points are denoted by the integer index *j*. Half-integer points can also be defined.

interval Δx is called the grid interval, grid size, mesh size, etc. We assume that the grid interval is constant for the time being; then $x_j = j\Delta x$, where j is the "index" used to identify the grid points. Note that u is defined only at the grid points denoted by the integers j, $j+1$, etc.

Using the notation $u_j = u(x_j) = u(j\Delta x)$, we can define the *forward difference* at the point j by

$$(\Delta u)_j \equiv u_{j+1} - u_j , \tag{2.1}$$

the *backward difference* at the point j by

$$(\nabla u)_j \equiv u_j - u_{j-1} \ , \tag{2.2}$$

and the *centered difference* at the point $j + \frac{1}{2}$ by

$$(\delta u)_{j+\frac{1}{2}} \equiv u_{j+1} - u_j \ . \tag{2.3}$$

Note that u itself is not defined at the point $j + \frac{1}{2}$.

From (2.1) - (2.3) we can define the following "finite-difference quotients:" the forward difference quotient at the point j:

$$\left(\frac{du}{dx}\right)_j \cong \frac{u_{j+1} - u_j}{\Delta x} \ ; \tag{2.4}$$

the backward difference quotient at the point j:

$$\left(\frac{du}{dx}\right)_j \cong \frac{u_j - u_{j-1}}{\Delta x} \ ; \tag{2.5}$$

and the centered difference quotient at the point $j + \frac{1}{2}$:

$$\left(\frac{du}{dx}\right)_{j+\frac{1}{2}} \cong \frac{u_{j+1} - u_j}{\Delta x} = \frac{(\delta u)_{j+\frac{1}{2}}}{\Delta x} \ . \tag{2.6}$$

In addition, the centered difference quotient at the point j can be defined by

$$\left(\frac{du}{dx}\right)_j \cong \frac{u_{j+1} - u_{j-1}}{2\Delta x} = \frac{1}{\Delta x}\frac{1}{2}\left[(\delta u)_{j+\frac{1}{2}} + (\delta u)_{j-\frac{1}{2}}\right] . \tag{2.7}$$

Since (2.4) and (2.5) employ the values of u at two points, they are sometimes referred to as two-point approximations. On the other hand, (2.6) and (2.7) are three-point approximations. When x is time, the time point is frequently referred to as a "level." In that case, (2.4) and (2.5) can be referred to as two-level approximations and (2.6) and (2.7) as three-level approximations.

How accurate are these finite-difference approximations? We now introduce the

concepts of accuracy and truncation error. As an example, consider the forward difference quotient

$$\left(\frac{du}{dx}\right)_j \cong \frac{u_{j+1} - u_j}{\Delta x} = \frac{u[(j+1)\Delta x] - u(j\Delta x)}{\Delta x}, \tag{2.8}$$

and expand u in a Taylor series about the point x_j to obtain

$$u_{j+1} = u_j + \Delta x\left(\frac{du}{dx}\right)_j + \frac{(\Delta x)^2}{2!}\left(\frac{d^2u}{dx^2}\right)_j + \frac{(\Delta x)^3}{3!}\left(\frac{d^3u}{dx^3}\right)_j + \ldots + \frac{(\Delta x)^{n-1}}{(n-1)!}\left(\frac{d^{n-1}u}{dx^{n-1}}\right)_j + \ldots, \tag{2.9}$$

which can be rearranged to

$$\frac{u_{j+1} - u_j}{\Delta x} = \left(\frac{du}{dx}\right)_j + \varepsilon, \tag{2.10}$$

where

$$\varepsilon \equiv \frac{\Delta x}{2!}\left(\frac{d^2u}{dx^2}\right)_j + \frac{(\Delta x)^2}{3!}\left(\frac{d^3u}{dx^3}\right)_j + \ldots + \frac{(\Delta x)^{n-2}}{(n-1)!}\left(\frac{d^{n-1}u}{dx^{n-1}}\right)_j + \ldots \tag{2.11}$$

is the error. The expansion (2.9) can be derived without any assumptions or approximations except that the indicated derivatives exist (Arfken, 1985; for a quick review see the Appendix on Taylor's Series). The terms of (2.10) the terms in (2.10) that lumped into ε are called the "truncation error." *The lowest power of Δx that appears in the truncation error is called the order of accuracy of the corresponding difference quotient*. For example, the leading term of (2.11) is of order Δx or $O(\Delta x)$, and so we say that (2.10) is a first-order approximation or an approximation of first-order accuracy. Obviously (2.5) is also first-order accurate.

Expansion of (2.6) and (2.7) similarly shows that they are of second-order accuracy. We can write

$$u_{j-1} = u_j + \left(\frac{du}{dx}\right)_j(-\Delta x) + \left(\frac{d^2u}{dx^2}\right)_j\frac{(-\Delta x)^2}{2!} + \left(\frac{d^3u}{dx^3}\right)_j\frac{(-\Delta x)^3}{3!} + \ldots. \tag{2.12}$$

Subtracting (2.12) from (2.9) gives

$$u_{j+1} - u_{j-1} = 2\left(\frac{du}{dx}\right)_j (\Delta x) + \frac{2}{3!}\left(\frac{d^3u}{dx^3}\right)_j (\Delta x)^3 + \ldots \text{ odd powers only}, \tag{2.13}$$

or

$$\left(\frac{du}{dx}\right)_j = \frac{u_{j+1} - u_{j-1}}{2\Delta x} - \left(\frac{d^3u}{dx^3}\right)_j \frac{(\Delta x)^2}{3!} + O[(\Delta x)^4]\,. \tag{2.14}$$

Similarly,

$$\left(\frac{du}{dx}\right)_{j+\frac{1}{2}} = \frac{u_{j+1} - u_j}{\Delta x} - \left(\frac{d^3u}{dx^3}\right)_{j+\frac{1}{2}} \frac{(\Delta x/2)^2}{3!} + O[(\Delta x)^4]\,. \tag{2.15}$$

In this way, we find that

$$\frac{\text{Error of } \left(\frac{du}{dx}\right)_j}{\text{Error of } \left(\frac{du}{dx}\right)_{j+\frac{1}{2}}} \cong \frac{\left(\frac{d^3u}{dx^3}\right)_j \frac{(\Delta x)^2}{3!}}{\left(\frac{d^3u}{dx^3}\right)_{j+\frac{1}{2}} \frac{(\Delta x/2)^2}{3!}} = \frac{4\left(\frac{d^3u}{dx^3}\right)_j}{\left(\frac{d^3u}{dx^3}\right)_{j+\frac{1}{2}}} \cong 4. \tag{2.16}$$

In other words, the error of $\left(\frac{du}{dx}\right)_j$ is four times as large as the error of $\left(\frac{du}{dx}\right)_{j+\frac{1}{2}}$, even though both finite-difference quotients have second-order accuracy. This makes the point that *the "order of accuracy" tells how rapidly the error changes as the grid is refined, but it does not tell how large the error is for a given grid size*. It is possible for a scheme of low-order accuracy to give a more accurate result than a scheme of higher-order accuracy, if a finer grid spacing is used with the low-order scheme.

Suppose that the leading term of the error has the form

$$\varepsilon \cong C(\Delta x)^p\,. \tag{2.17}$$

Then $\ln(\varepsilon) \cong p\ln(\Delta x) + \ln(C)$, and so

$$\frac{d[\ln(\varepsilon)]}{d[\ln(\Delta x)]} \cong p\,. \tag{2.18}$$

The implication is that if we plot $\ln(\varepsilon)$ as a function of $\ln(\Delta x)$ (i.e., plot the error as a function of the grid spacing on "log-log" paper), we will get (approximately) a straight line whose slope is p. This is a simple way to determine empirically the order of accuracy of a finite-difference quotient. Of course, in order to carry this out it is necessary to compute the error of the finite-difference approximation, and that can only be done when the exact derivative is known. Therefore, this approach is usually implemented by using an analytical "test function."

2.2 Difference quotients of higher accuracy

Suppose that we write

$$\frac{u_{j+2}-u_{j-2}}{4\Delta x} = \left(\frac{du}{dx}\right)_j + \frac{1}{3!}\left(\frac{d^3u}{dx^3}\right)_j (2\Delta x)^2 + \ldots \text{ even powers only} . \tag{2.19}$$

Here we have written a centered difference using the points $j+2$ and $j-2$ instead of $j+1$ and $j-1$, respectively. It should clear that (2.19) is second-order accurate, although for any given value of Δx the error of (2.19) is expected to be larger than the error of (2.14). We can combine (2.14) and (2.19) with a weight, w, so as to obtain a "hybrid" approximation to $\left(\frac{du}{dx}\right)_j$:

$$\begin{aligned}\left(\frac{du}{dx}\right)_j &= w\left(\frac{u_{j+1}-u_{j-1}}{2\Delta x}\right) + (1-w)\left(\frac{u_{j+2}-u_{j-2}}{4\Delta x}\right) \\ &- \frac{w}{3!}\left(\frac{d^3u}{dx^3}\right)_j (\Delta x)^2 - \frac{(1-w)}{3!}\left(\frac{d^3u}{dx^3}\right)_j (2\Delta x)^2 + O[(\Delta x)^4] \quad .\end{aligned} \tag{2.20}$$

Inspection of (2.20) shows that we can force the coefficient of $(\Delta x)^2$ to vanish by choosing

$$w + (1-w)4 = 0 \text{, or } w = 4/3 . \tag{2.21}$$

With this choice, (2.20) reduces to

$$\left(\frac{du}{dx}\right)_j = \frac{4}{3}\left(\frac{u_{j+1}-u_{j-1}}{2\Delta x}\right) - \frac{1}{3}\left(\frac{u_{j+2}-u_{j-2}}{4\Delta x}\right) + O[(\Delta x)^4] . \tag{2.22}$$

We have thus obtained a fourth-order scheme. In effect, this is a linear extrapolation of the value of the finite-difference expression to a smaller grid size, as schematically illustrated in Fig. 2.2.

Is there a more systematic way to construct schemes of any desired order of accuracy? The answer is "yes," and one such approach is as follows.

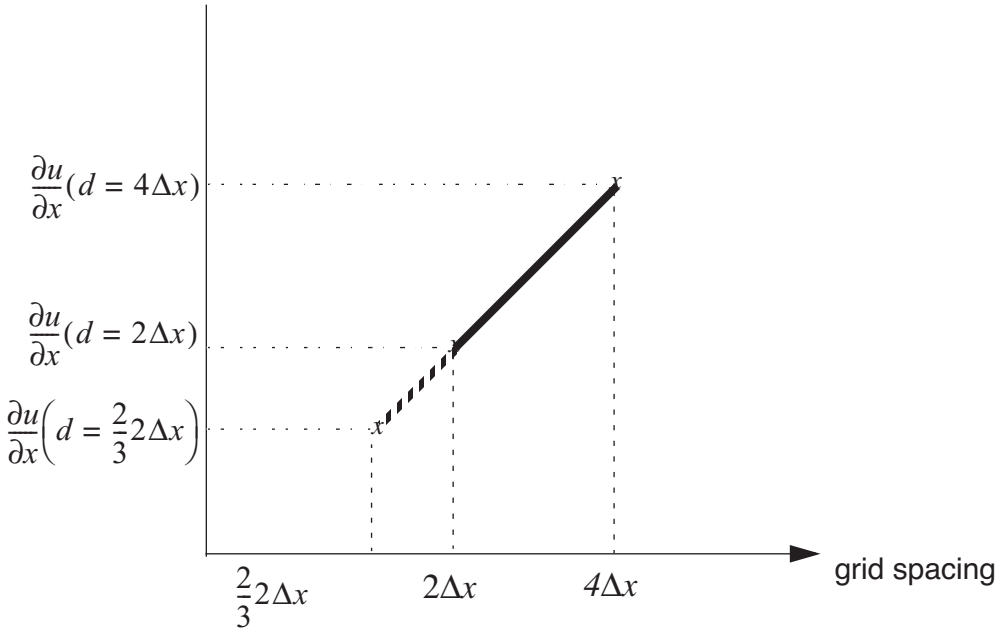

Figure 2.2: Schematic illustrating the interpretation of the fourth-order difference in terms of the extrapolation of the second-order difference based on a spacing of 4Δx, and that based on a spacing of 2Δx. The extrapolation reduces the effective grid size to (2/3)2Δx.

Suppose that we write a finite-difference approximation to $\left(\frac{df}{dx}\right)_j$ in the following somewhat generalized form:

$$\left(\frac{df}{dx}\right)_j \cong \frac{1}{\Delta x}\sum_{i=-\infty}^{\infty} a_i f(x_j + i\Delta x) \quad . \tag{2.23}$$

Here the a_i are coefficients or "weights," which are undetermined at this point. In most schemes, all but a few of the a_i will be zero, so that the sum in (2.23) will actually involve only a few non-zero terms. In writing (2.23), we have assumed for simplicity that Δx is a constant; this assumption can be relaxed, as will be shown below. The index i in (2.23) is a counter that is zero at our "home base" at grid point j. For $i < 0$ we count to the left, and for $i > 0$ we count to the right. According to (2.23), our finite-difference approximation to $\left(\frac{df}{dx}\right)_j$ has the form of a weighted sum of values of f at various grid points in the vicinity of point j. Every finite-difference approximation that we have considered so far does indeed have this form, but you should be aware that there are (infinitely many!) schemes that do not

have this form; a few of them will be discussed later.

Introducing a Taylor series expansion, we can write

$$f(x_j + i\Delta x) = f_j^0 + f_j^1 (i\Delta x) + f_j^2 \frac{(i\Delta x)^2}{2!} + f_j^3 \frac{(i\Delta x)^3}{3!} + \ldots \tag{2.24}$$

Here f_j^n is the nth derivative of f, evaluated at the point j. Using (2.24), we can rewrite (2.23) as

$$\left(\frac{df}{dx}\right)_j \cong \frac{1}{\Delta x} \sum_{i=-\infty}^{\infty} a_i \left[f_j^0 + f_j^1 (i\Delta x) + f_j^2 \frac{(i\Delta x)^2}{2!} + f_j^3 \frac{(i\Delta x)^3}{3!} + \ldots \right]. \tag{2.25}$$

By inspection of (2.25), we see that in order to have at least first-order accuracy, we need

$$\sum_{i=-\infty}^{\infty} a_i = 0 \text{ and } \sum_{i=-\infty}^{\infty} i a_i = 1 . \tag{2.26}$$

To have at least second-order accuracy, we must impose an additional requirement:

$$\sum_{i=-\infty}^{\infty} i^2 a_i = 0 . \tag{2.27}$$

In general, to have at least nth-order accuracy, we must require that

$$\sum_{i=-\infty}^{\infty} i^m a_i = \delta_{m,1} \text{ for } 0 \le m \le n . \tag{2.28}$$

Here $\delta_{m,1}$ is the Kronecker delta. In order to satisfy (2.28), we must solve a system of $n + 1$ linear equations for the $n + 1$ unknown coefficients a_i.

According to (2.28), a scheme of nth-order accuracy can be constructed by satisfying $n + 1$ equations. In particular, because (2.26) involves two equations, a first-order scheme has to involve at least two grid points, i.e., there must be at least two non-zero values of a_i. This is pretty obvious. Note that we could make a first-order scheme that used fifty grid points if we wanted to -- but then, why would we want to? A second-order scheme must involve at least three grid points. A scheme that is parsimonious in its use of points is called "*compact*."

Consider a simple example. Still assuming a uniform grid, a first order scheme for

$\left(\frac{df}{dx}\right)_j$ can be constructed using the points j and $j+1$ as follows. From (2.26), we get $a_0 + a_1 = 0$ and $a_1 = 1$. Obviously we must choose $a_0 = -1$. Substituting into (2.23), we find that the scheme is given by $\left(\frac{df}{dx}\right)_j \cong \frac{f(x_j + \Delta x) - f(x_j)}{\Delta x}$, i.e., the one-sided difference discussed earlier. Obviously we can also construct a first-order scheme using the points j and $j-1$, with a similar one-sided result. If we choose the points $j+1$ and $j-1$, imposing the requirements for first-order accuracy, i.e., (2.26), will actually give us the centered second-order scheme, i.e., $\left(\frac{df}{dx}\right)_j \cong \frac{f(x_j + \Delta x) - f(x_{j-1})}{2\Delta x}$, because (2.27) is satisfied "accidentally" or "automatically" -- we manage to satisfy three equations using only two unknowns. If we choose the three points $j-1$, j and $j+1$, and require second-order accuracy, we get exactly the same centered scheme, because a_0 turns out to be zero.

Next, we work out a generalization of the family of schemes given above, for the case of (possibly) non-uniform grid spacing. Eq. (2.23) is replaced by

$$\left(\frac{df}{dx}\right)_j \cong \sum_{i=-\infty}^{\infty} b_i f(x_{j+i}) \ . \tag{2.29}$$

Note that, since Δx is no longer a constant, the factor of $\frac{1}{\Delta x}$ that appears in (2.23) has been omitted in (2.29), and in view of this, in order to avoid notational confusion, we have replaced the symbol a_i by b_i. Similarly, Eq. (2.24) is replaced by

$$f(x_{j+i}) = f_j^0 + f_j^1 (x_{j+i} - x_j) + f_j^2 \frac{(x_{j+i} - x_j)^2}{2!} + f_j^3 \frac{(x_{j+i} - x_j)^3}{3!} + \ldots \tag{2.30}$$

Substitution of (2.30) into (2.29) gives

$$\left(\frac{df}{dx}\right)_j \cong \sum_{i=-\infty}^{\infty} b_i \left[f_j^0 + f_j^1 (x_{j+i} - x_j) + f_j^2 \frac{(x_{j+i} - x_j)^2}{2!} + f_j^3 \frac{(x_{j+i} - x_j)^3}{3!} + \ldots \right] \ . \tag{2.31}$$

To have first-order accuracy with (2.31), we must require that

$$\sum_{i=-\infty}^{\infty} b_i = 0 \text{ and } \sum_{i=-\infty}^{\infty} b_i (x_{j+i} - x_j) = 1 \ . \tag{2.32}$$

It may appear that when we require first-order accuracy by enforcing (2.32), the leading term of the error in (2.31), namely $\sum_{i=-\infty}^{\infty} b_i f_j^2 \frac{(x_{j+i}-x_j)^2}{2!}$, will be of order $(\Delta x)^2$, but this is not really true because, as shown below, $b_i \sim \frac{1}{\Delta x}$.

Similarly, to achieve second-order accuracy with (2.31), we must require, in addition to (2.32), that

$$\sum_{i=-\infty}^{\infty} b_i (x_{j+i}-x_j)^2 = 0 . \tag{2.33}$$

In general, to have at least nth-order accuracy, we must require that

$$\sum_{i=-\infty}^{\infty} (x_{j+i}-x_j)^m b_i = \delta_{m,1} \text{ for } 0 \le m \le n . \tag{2.34}$$

As an example, the first-order accurate scheme using the points j and $j+1$ must satisfy the two equations obtained from (2.32), i.e., $b_0 + b_1 = 0$ and $b_1 = \frac{1}{x_{j+i}-x_j}$, so that $b_1 = \frac{-1}{x_{j+i}-x_j}$. From (2.29), the scheme is $\left(\frac{df}{dx}\right)_j \cong \frac{f(x_{j+1})-f(x_j)}{x_{j+i}-x_j}$, which, clearly, is equivalent to the result that we obtained for the case of the uniform grid.

To obtain a second-order accurate approximation to $\left(\frac{df}{dx}\right)_j$ on an arbitrary grid, using the three points $j-1$, j and $j+1$, we must require, from (2.32) that

$$b_{-1} + b_0 + b_1 = 0 , \text{and } b_{-1}(x_{j-1}-x_j) + b_1(x_{j+1}-x_j) = 1 , \tag{2.35}$$

and from (2.34) that

$$b_{-1}(x_{j-1}-x_j)^2 + b_1(x_{j+1}-x_j)^2 = 0 . \tag{2.36}$$

The solution of this system of three equations is

$$b_{-1} = \left(\frac{-1}{x_{j+i}-x_{j-1}}\right)\left(\frac{x_{j+i}-x_j}{x_j-x_{j-1}}\right) , \tag{2.37}$$

$$b_0 = \left(\frac{1}{x_{j+i} - x_{j-1}}\right)\left[\left(\frac{x_{j+i} - x_j}{x_j - x_{j-1}}\right) - \left(\frac{x_j - x_{j-1}}{x_{j+i} - x_j}\right)\right] , \tag{2.38}$$

$$b_1 = \left(\frac{1}{x_{j+i} - x_{j-1}}\right)\left(\frac{x_j - x_{j-1}}{x_{j+i} - x_j}\right) . \tag{2.39}$$

For the case of uniform grid-spacing this reduces to the familiar centered second-order scheme.

Here is a simple but very practical question: Suppose that we use a scheme that has second-order accuracy on a uniform grid, but we apply it on a non-uniform grid. What happens? As a concrete example, we use the scheme

$$\left(\frac{df}{dx}\right)_j \cong \frac{f(x_{j+1}) - f(x_{j-1})}{x_{j+1} - x_{j-1}} . \tag{2.40}$$

By inspection, we have

$$b_{-1} = \frac{-1}{x_{j+i} - x_{j-1}} , \tag{2.41}$$

$$b_0 = 0 , \tag{2.42}$$

$$b_1 = \frac{1}{x_{j+i} - x_{j-1}} . \tag{2.43}$$

Eqs. (2.41)-(2.43) can be compared with (2.37)-(2.39). Obviously the scheme does not have second-order accuracy on a non-uniform grid, because (2.37)-(2.39) are not satisfied for a non-uniform grid. We note, however, that Eqs. (2.41)-(2.43) do satisfy both of the conditions in (2.35), even when the grid is non-uniform. *This means that the scheme does have first-order accuracy, even on the non-uniform grid*.

This example illustrates that a scheme that has been designed for use on a uniform grid, with second-order (or even better than second-order) accuracy, will reduce to a scheme of first-order accuracy when applied on a non-uniform grid. A special case has been used as an example here, but the conclusion is true quite generally.

Finally, we observe that a very similar approach can be used to derive approximations to higher-order derivatives of f. For example, to derive approximations to $\left(\frac{d^2 f}{dx^2}\right)$, on a (possibly) non-uniform grid, we write

$$\left(\frac{d^2f}{dx^2}\right)_j \cong \sum_{i=-\infty}^{\infty} c_i f(x_{j+i}) . \tag{2.44}$$

Obviously, it is going to turn out that $c_i \sim \frac{1}{(\Delta x)^2}$. Substitution of (2.30) into (2.44) gives

$$\left(\frac{d^2f}{dx^2}\right)_j \cong \sum_{i=-\infty}^{\infty} c_i \left[f_j^0 + f_j^1 (x_{j+i} - x_j) + f_j^2 \frac{(x_{j+i} - x_j)^2}{2!} + f_j^3 \frac{(x_{j+i} - x_j)^3}{3!} + \ldots \right]. \tag{2.45}$$

Keeping in mind that $c_i \sim \frac{1}{(\Delta x)^2}$, we see that a first-order accurate approximation is ensured if we enforce the three conditions

$$\sum_{i=-\infty}^{\infty} c_i = 0 , \text{ and } \sum_{i=-\infty}^{\infty} c_i (x_{j+i} - x_j)^2 = 2! \text{ and } \sum_{i=-\infty}^{\infty} c_i (x_{j+i} - x_j) = 0 . \tag{2.46}$$

To achieve a second-order accurate approximation to the second derivative, we must additionally require that

$$\sum_{i=-\infty}^{\infty} c_i (x_{j+i} - x_j)^3 = 0 . \tag{2.47}$$

In general, to have an *n*th-order accurate approximation to the second derivative, we must require that

$$\sum_{i=-\infty}^{\infty} (x_{j+i} - x_j)^m c_i = (2!)\delta_{m,2} \text{ for } 0 \le m \le n+1 . \tag{2.48}$$

Earlier we showed that, in general, a second-order approximation to the first derivative must involve a minimum of three grid points, because three conditions must be satisfied [i.e., (2.35) and (2.36)]. Now we see that a second-order approximation to the second derivative must involve a minimum of four grid points, because four conditions must be satisfied, i.e., (2.46) and (2.47). In the special case of a uniform grid, three points suffice. With a non-uniform grid, five points may be preferable to four, from the point of view of symmetry.

At this point, you should be able to see ("by induction") that on a (possibly) non-uniform grid, an *n*th-order accurate approximation to the lth derivative of f takes the form

$$\left(\frac{d^l f}{dx^l}\right)_j \cong \sum_{i=-\infty}^{\infty} d_i f(x_{j+i}) \tag{2.49}$$

where

$$\sum_{i=-\infty}^{\infty} (x_{j+i} - x_j)^m d_i = (l!)\delta_{m,l} \quad \text{for } 0 \le m \le n + l - 1 . \tag{2.50}$$

This is a total of $n + l$ requirements, so in general a minimum of $n + l$ points will be needed. It is straightforward to write a computer program that will automatically generate the coefficients for a compact nth-order accurate approximation to the lth derivative of f.

What happens for $l = 0$?

2.3 Extension to two dimensions

The approach presented above can be generalized to multi-dimensional problems. We will use the two-dimensional Laplacian operator as an example. Consider a finite-difference approximation to the Laplacian, of the form

$$(\nabla^2 f)_j \cong \sum_{k=-\infty}^{\infty} c_k f(x_{j+k}, y_{j+k}) \quad . \tag{2.51}$$

Here we use one-dimensional indices even though we are on a two-dimensional grid. The subscript j denotes a particular grid point ("home base" for this calculation), whose coordinates are (x_j, y_j). Similarly, the subscript $j + k$ denotes a different grid point whose coordinates are (x_{j+k}, y_{j+k}).

The two-dimensional Taylor series expansion is

$$
\begin{aligned}
f(x_{j+k}, y_{j+k}) &= f(x_j, y_j) + [(\delta x)_k f_x + (\delta y)_k f_y] \\
&+ \frac{1}{2!}[(\delta x)_k{}^2 f_{xx} + 2(\delta x)_k(\delta y)_k f_{xy} + (\delta y)_k{}^2 f_{yy}] + \dots \quad , \\
&+ \frac{1}{3!}[(\delta x)_k{}^3 f_{xxx} + 3(\delta x)_k{}^2(\delta y)_k f_{xxy} + 3(\delta x)_k(\delta y)_k{}^2 f_{xyy} + (\delta y)_k{}^3 f_{yyy}] \\
&+ \frac{1}{4!}[(\delta x)_k{}^4 f_{xxxx} + 4(\delta x)_k{}^3(\delta y)_k f_{xxxy} + 6(\delta x)_k{}^2(\delta y)_k{}^2 f_{xxyy} \\
&+ 4(\delta x)_k(\delta y)_k{}^3 f_{xyyy} + (\delta y)_k{}^4 f_{yyyy}] + \dots \} \quad ,
\end{aligned}
\tag{2.52}
$$

where

$$
(\delta x)_k \equiv x_{j+k} - x_j \quad \text{and} \quad (\delta y)_k \equiv y_{j+k} - y_j \ , \tag{2.53}
$$

and it is understood that all of the derivatives are evaluated at the point (x_j, y_j) .

Substituting from (2.52) into (2.51), we find that

$$
\begin{aligned}
(\nabla^2 f)_j \cong \sum_{k=-\infty}^{\infty} c_k \{ & f(x_j, y_j) + [(\delta x)_k f_x + (\delta y)_k f_y] \\
&+ \frac{1}{2!}[(\delta x)_k{}^2 f_{xx} + 2(\delta x)_k(\delta y)_k f_{xy} + (\delta y)_k{}^2 f_{yy}] + \dots \quad , \\
&+ \frac{1}{3!}[(\delta x)_k{}^3 f_{xxx} + 3(\delta x)_k{}^2(\delta y)_k f_{xxy} + 3(\delta x)_k(\delta y)_k{}^2 f_{xyy} + (\delta y)_k{}^3 f_{yyy}] \\
&+ \frac{1}{4!}[(\delta x)_k{}^4 f_{xxxx} + 4(\delta x)_k{}^3(\delta y)_k f_{xxxy} + 6(\delta x)_k{}^2(\delta y)_k{}^2 f_{xxyy} \\
&+ 4(\delta x)_k(\delta y)_k{}^3 f_{xyyy} + (\delta y)_k{}^4 f_{yyyy}] + \dots \} \quad .
\end{aligned}
\tag{2.54}
$$

To have first-order accuracy, we need

$$
\sum_{k=-\infty}^{\infty} c_k = 0 \ , \tag{2.55}
$$

$$\sum_{k=-\infty}^{\infty} c_k (\delta x)_k^2 = 2! \ , \tag{2.56}$$

$$\sum_{k=-\infty}^{\infty} c_k (\delta y)_k^2 = 2! \ , \tag{2.57}$$

$$\sum_{k=-\infty}^{\infty} c_k (\delta x)_k = 0 \tag{2.58}$$

$$\sum_{k=-\infty}^{\infty} c_k (\delta x)_k = 0 \text{ , and} \tag{2.59}$$

$$\sum_{k=-\infty}^{\infty} c_k (\delta x)_k (\delta y)_k = 0 \ . \tag{2.60}$$

From (2.56) and (2.57), it is clear that c_k is of order δ^2, where δ denotes δx or δy. Therefore, the quantities inside the sums in (2.58) and (2.59) are of order δ^{-1}, and the quantities inside the sum of (2.60) or of order one. This is why (2.58)-(2.60) are required, in addition to (2.55)-(2.57), to obtain first-order accuracy.

So far (2.55) - (2.60) involve only six equations, and so six grid points are needed. To get second-order (or higher) accuracy, we will need to add more points, unless we are fortunate enough to use a highly symmetrical grid that permits the conditions for higher-order accuracy to be satisfied automatically. For example, if we satisfy (2.55) - (2.60) on a square grid, we will get second-order accuracy "for free." More generally, with a non-uniform grid, we need the following four additional conditions to achieve second-order accuracy:

$$\sum_{k=-\infty}^{\infty} c_k (\delta x)_k^3 = 0 \ , \tag{2.61}$$

$$\sum_{k=-\infty}^{\infty} c_k (\delta x)_k^2 (\delta y)_k = 0 \ , \tag{2.62}$$

$$\sum_{k=-\infty}^{\infty} c_k(\delta x)_k(\delta y)_k^{\,2} = 0 \,, \tag{2.63}$$

$$\sum_{k=-\infty}^{\infty} c_k(\delta y)_k^{\,3} = 0 \,. \tag{2.64}$$

Therefore, in general a total of ten conditions must be satisfied to ensure second-order accuracy on a non-uniform grid.

For the continuous Laplacian on a closed or periodic domain, we can prove the following:

$$\int_A (\nabla^2 f)\, dA = 0 \,, \tag{2.65}$$

$$\int_A f(\nabla^2 f)\, dA \le 0 \,. \tag{2.66}$$

Here the integrals are with respect to area, over the entire domain. The corresponding finite-difference requirements are

$$\sum_{\text{all } j} (\nabla^2 f)_j A_j \cong \sum_{\text{all } j} \left[\sum_{k=-\infty}^{\infty} c_k f(x_{j+k}, y_{j+k}) \right] A_j = 0 \quad \text{, and} \tag{2.67}$$

$$\sum_{\text{all } j} f_j (\nabla^2 f)_j A_j \cong \sum_{\text{all } j} f_j \left[\sum_{k=-\infty}^{\infty} c_k f(x_{j+k}, y_{j+k}) \right] A_j \le 0 \quad , \tag{2.68}$$

where A_j is the area of grid-cell j. These requirements must hold for an arbitrary spatial distribution of f, so they actually represent conditions on the c_k. Very similar (but more complicated) requirements were discussed by Arakawa (1966), in the context of the Jacobian operator. Further discussion is given later.

2.4 An example of a finite difference–approximation to a differential equation

With these definitions and concepts, we proceed directly to a simple example of a partial differential equation. Consider the one-dimensional "advection" equation,

$$\left(\frac{\partial u}{\partial t}\right)_x + c\left(\frac{\partial u}{\partial x}\right)_t = 0\,, \tag{2.69}$$

where c is a constant, and $u = u(x, t)$. This is a first-order linear partial differential equation with a constant coefficient, namely c. Eq. (2.69) looks harmless, but it causes no end of trouble, as we will see.

Suppose that

$$u(x, 0) = F(x) \text{ for } -\infty < x < \infty\,. \tag{2.70}$$

This is an "initial condition." What is $u(x, t)$? This is a simple example of an initial value problem. We first work out the analytic solution, for later comparison with our numerical solution. Define

$$\xi \equiv x - ct\,. \tag{2.71}$$

We can write

$$\begin{aligned}\left(\frac{\partial u}{\partial x}\right)_\xi &= \left(\frac{\partial u}{\partial x}\right)_t + \left(\frac{\partial u}{\partial t}\right)_x \left(\frac{\partial t}{\partial x}\right)_\xi \\ &= \left(\frac{\partial u}{\partial x}\right)_t + \left(\frac{\partial u}{\partial t}\right)_x \frac{1}{c} \\ &= 0\quad .\end{aligned} \tag{2.72}$$

The first line of (2.72) can be understood by reference to Fig. 2.4. Similarly,

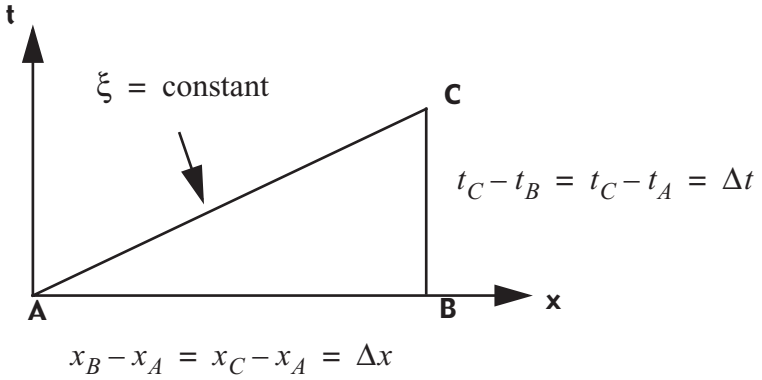

Figure 2.3: Figure used in the derivation of the first line of (2.72).

$$\begin{aligned}\left(\frac{\partial u}{\partial t}\right)_{\xi} &= \left(\frac{\partial u}{\partial t}\right)_{x} + \left(\frac{\partial u}{\partial x}\right)_{t}\left(\frac{\partial x}{\partial t}\right)_{\xi} \\ &= \left(\frac{\partial u}{\partial x}\right)_{t} + \left(\frac{\partial u}{\partial t}\right)_{x} c \\ &= 0 \quad .\end{aligned} \tag{2.73}$$

It follows that

$$u = f(\xi) \tag{2.74}$$

is the general solution to (2.69). At $t = 0$, $\xi \equiv x$ and $u(x) = f(x)$, i.e. the shape of f is determined by the initial condition. Eq. (2.74) means that u is constant along the line $\xi \equiv x - ct =$ constant. In order to satisfy the initial condition (2.70), we chose $f \equiv F$. Thus $u(\xi) = F(\xi) = F(x - ct)$ is the solution to the differential equation (2.69) which satisfies the initial condition. We see that an initial value simply "moves along" the lines of constant ξ. The initial "shape" of $u(x)$, namely $F(x)$, is just carried along by the wind. From a physical point of view this is obvious.

Keeping in mind this exact solution, we now investigate one possible numerical solution of (2.69). We construct a grid, as in Fig. 2.4. An example of a finite difference approximation to (2.69) is

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\left(\frac{u_j^n - u_{j-1}^n}{\Delta x}\right) = 0 \; . \tag{2.75}$$

Here we have used the forward difference quotient in time and the backward difference quotient in space. If $c > 0$, (2.75) is called the "upstream" difference scheme. Because

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} \to \frac{\partial u}{\partial t} \text{ as } \Delta t \to 0 \; , \tag{2.76}$$

and

$$\frac{u_j^n - u_{j-1}^n}{\Delta x} \to \frac{\partial u}{\partial x} \text{ as } \Delta x \to 0 \; . \tag{2.77}$$

we conclude that (2.75) does approach (2.69) as Δt and Δx both approach zero. The upstream scheme has some serious weaknesses, but it also has some very useful properties.

If we know u_j^n at some time level n for all j, then we can compute u_j^{n+1} at the next time level $n + 1$. Note that this scheme is one-sided or asymmetric in both space and time. It

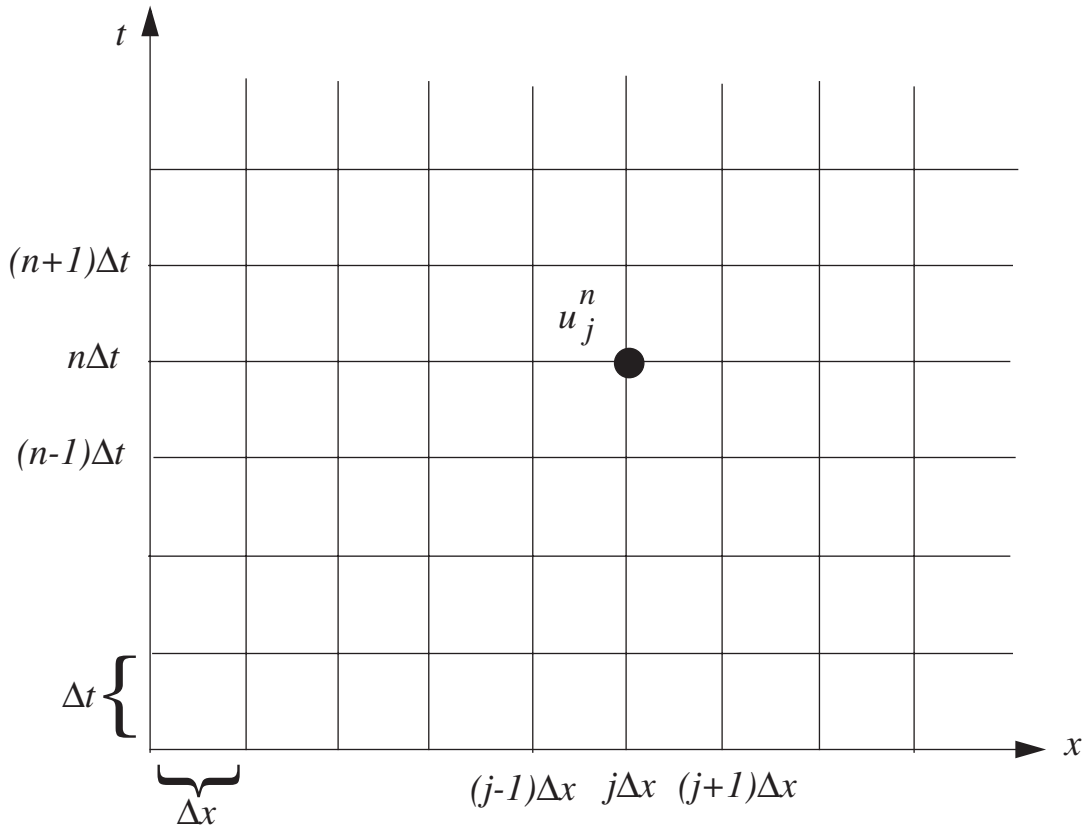

Figure 2.4: A grid for the solution of the one-dimensional advection equation.

seems naturally suited to modeling advection, in which air comes from one side and goes to the other as time passes by.

In view of (2.76) and (2.77), it may seem obvious that the *solution* of (2.75) approaches the *solution* of (2.69) as $\Delta x \to 0$ and $\Delta t \to 0$. This "obvious" conclusion is not necessarily true, as we shall see.

2.5 Accuracy and truncation error of a finite-difference scheme.

We have already defined accuracy and truncation error for finite difference quotients. Now we define truncation error and accuracy for a finite-difference scheme. Here we define a finite-difference scheme as a finite-difference equation which approximates, term-by-term, a differential equation.

It is easy to find an approximation to each term of a differential equation, and we have already seen that the error of such an approximation can be made as small as desired, almost effortlessly. This is not our goal, however. *Our goal is to find an approximation to the solution of the differential equation*. Now you might think that if we have a finite-difference

equation, F, that is constructed by writing down a good approximation to each term of a differential equation, D, then the solution of F will be a useful approximation to the solution of D. Wouldn't that be nice? Unfortunately, it isn't necessarily true.

Letting $u(x, t)$ denote the (exact) solution of the differential equation, we see that $u(j\Delta x, n\Delta t)$ is the value of this exact solution at the discrete point $(j\Delta x, n\Delta t)$ on the grid shown in Fig. 2.3, while u_j^n is the "exact" solution of a finite-difference equation, at the same point. In general, $u_j^n \neq u(j\Delta x, n\Delta t)$. We wish that they were equal!

A measure of the accuracy of the finite-difference *scheme* can be obtained by substituting the solution of the differential equation into the finite-difference equation. For the upstream scheme given by (2.75), we get

$$\frac{u[j\Delta x, (n+1)\Delta t] - u(j\Delta x, n\Delta t)}{\Delta t} + c\left\{\frac{u(j\Delta x, n\Delta t) - u[(j-1)\Delta x, n\Delta t]}{\Delta x}\right\} = \varepsilon , \quad (2.78)$$

where ε is called "truncation error" of the scheme. The truncation error of the scheme is a measure of how accurately the solution $u(x, t)$ of the original differential equation (2.69) satisfies the finite-difference equation (2.75).

Note that, since u_j^n is defined only at discrete points, it is not differentiable, and so we cannot substitute u_j^n into the differential equation. Because of this, we cannot measure how accurately u_j^n satisfies the differential equation.

If we obtain the terms in (2.78) from Taylor Series expansion of $u(x, t)$ about the point $(j\Delta x, n\Delta t)$, and use the fact that $u(x, t)$ satisfies (2.69), we find that

$$\varepsilon = \left(\frac{1}{2!}\Delta t\frac{\partial^2 u}{\partial t^2} + \ldots\right) + c\left(-\frac{1}{2!}\Delta x\frac{\partial^2 u}{\partial x^2} + \ldots\right). \quad (2.79)$$

We say this is a "first-order scheme" because the lowest powers of Δt and Δx in (2.79) are 1. The notations $O(\Delta t, \Delta x)$ or $O(\Delta t) + O(\Delta x)$ can be used to express this. *We say that a scheme is consistent with the differential equation if the truncation error of the scheme approaches zero as Δt and Δx approach zero*. The upstream scheme under consideration here is, therefore, consistent.

2.6 Discretization error and convergence

There are two sources of error in a numerical solution. One is the *round-off error*, which is the difference of a numerical solution from the "exact" solution of the finite

difference equation, u_j^n, and is a property of the machine being used (and to some extent the details of the program). The other source of error is the *discretization error* defined by $u_j^n - u(j\Delta x, n\Delta t)$. Round-off error can sometimes be a problem but usually it is not, and we will not consider it in this course.

The truncation error, discussed in the last section, can be made as small as desired by making Δx and Δt smaller and smaller, so long as the scheme is consistent and $u(x, t)$ is a smooth function. *A decrease in the truncation error does not necessarily guarantee that the discretization error will also become small, however*. This is demonstrated below.

We now analyze how the discretization error changes as the grid is refined (i.e., as Δt and $\Delta x \to 0$). If the discretization error approaches zero, then we say that the solution *converges*. Fig. 2.5 gives an example of a situation in which accuracy is increased but the

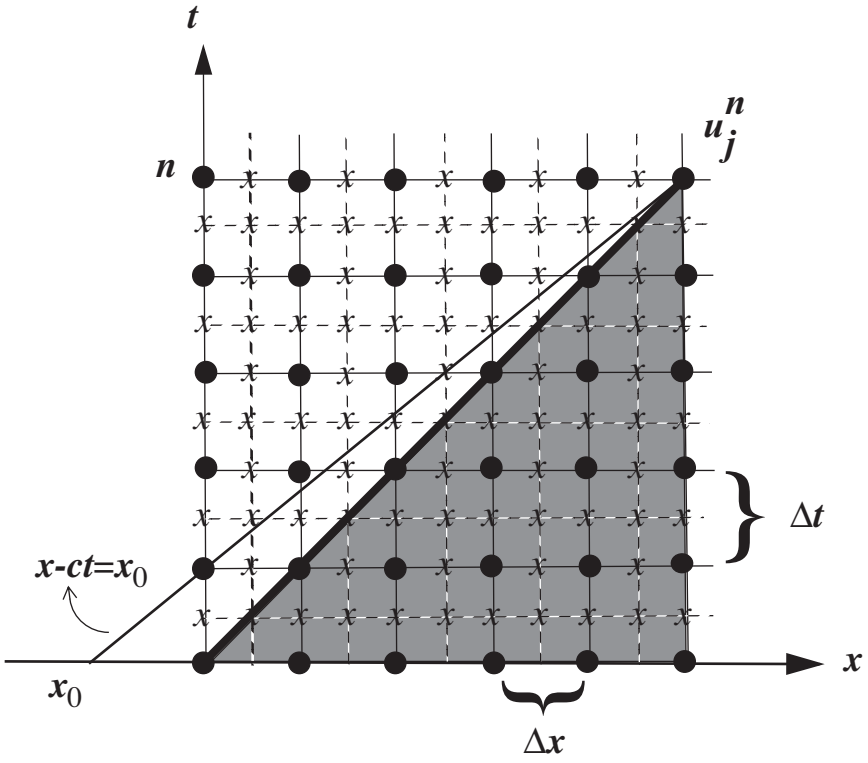

Figure 2.5: The shaded area represents the "domain of dependence" of the solution of the upstream scheme at the point $x = j\Delta x$, $t = n\Delta t$.

solution nevertheless does not converge. The thin diagonal line in the figure shows the characteristic along with u is "carried," i.e. u is constant along the line. This is the exact solution. To work out the numerical approximation to this solution, we first choose Δx and Δt such that the grid points are the dots in the figure. The domain consisting of the grid

points carrying values of u on which u_j^n depends is called the "*domain of dependence*." The shaded area in the figure shows this domain for the upstream scheme (2.75).

We could increase the accuracy of the scheme by cutting Δx and Δt in half, that is, by adding the points denoted by small x's forming a denser grid. Notice that the domain of dependence does not change, no matter how refined or dense the grid is, so long as the ratio $\frac{c\Delta t}{\Delta x}$ remains the same. This is a clue that $\frac{c\Delta t}{\Delta x}$ is an important quantity.

Suppose that the line through the point $(j\Delta x, n\Delta t)$, $x - ct = x_0$, where x_0 is a constant, does not lie in the domain of dependence. This is the situation shown in the figure. In general, there is no hope of obtaining smaller discretization error, no matter how small Δx and Δt become, so long as $\frac{c\Delta t}{\Delta x}$ is unchanged, because the true solution $u(j\Delta x, n\Delta t)$ depends only on the initial value of u at the single point $(x_0, 0)$ which cannot influence u_j^n. *You could change* $u(x_0, 0)$ *[and hence* $u(j\Delta x, n\Delta t)$*], but the computed solution* u_j^n *would remain the same*. In such a case, the error of the solution usually will not be decreased by refining the grid. This illustrates that if the value of c is such that x_0 lies outside of the domain of dependence, it is not possible for the solution of the finite-difference equation to approach the solution of the differential equation, no matter how fine the mesh becomes.

A finite-difference scheme for which the discretization error can be made small *for any initial condition* is called a *convergent* finite difference scheme. Therefore,

$$0 \le \frac{c\Delta t}{\Delta x} \le 1 \ , \tag{2.80}$$

is a *necessary* condition for convergence when the upstream scheme is used. Notice that if c is negative (giving what we might call a "downstream" scheme), it is impossible to satisfy (2.80). Of course, for $c < 0$ we can use

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\left(\frac{u_{j+1}^n - u_j^n}{\Delta x}\right) = 0 \ , \tag{2.81}$$

in place of (2.75). So our computer program can have an "if-test" that checks the sign of c, and uses (2.75) if $c \ge 0$, and (2.81) if $c < 0$. This is bad, though, because if-tests can cause slow execution on certain types of computers, and besides, if-tests are ugly. If we define

$$c_+ \equiv \frac{c + |c|}{2} \geq 0 \text{ , and } c_- \equiv \frac{c - |c|}{2} \leq 0 \text{ ,} \tag{2.82}$$

then the upstream scheme can be written as

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c_+ \left(\frac{u_j^n - u_{j-1}^n}{\Delta x} \right) + c_- \left(\frac{u_{j+1}^n - u_j^n}{\Delta x} \right) = 0 \text{ .} \tag{2.83}$$

This form avoids the use of if-tests and is also convenient for use in pencil-and-paper analysis, discussed later.

In summary:

Truncation error measures the accuracy of an approximation to a differential operator or operators. It is a measure of the accuracy with which a differential equation has been approximated.

Discretization error measures the accuracy with which the solution of the differential equation has been approximated.

Minimizing the truncation error is usually easy. Minimizing the discretization error can be much harder.

2.7 Interpolation and extrapolation

Referring to (2.75), we can rewrite the upstream scheme as

$$u_j^{n+1} = u_j^n (1 - \mu) + u_{j-1}^n \mu \tag{2.84}$$

where

$$\mu \equiv \frac{c \Delta t}{\Delta x} \text{ .} \tag{2.85}$$

This scheme has the form of either an *interpolation* or an *extrapolation,* depending on the value of μ . To see this, refer to Fig. 2.6. Along the line plotted in the figure,

$$u = u_{j-1}^n + \left(\frac{x - x_{j-1}}{x_j - x_{j-1}} \right) (u_j^n - u_{j-1}^n) = \left[1 - \left(\frac{x - x_{j-1}}{x_j - x_{j-1}} \right) u_{j-1}^n + \left(\frac{x - x_{j-1}}{x_j - x_{j-1}} \right) u_j^n \right] \tag{2.86}$$

which has the same form as our scheme if we identify

$$\mu \equiv \frac{x - x_{j-1}}{x_j - x_{j-1}} \tag{2.87}$$

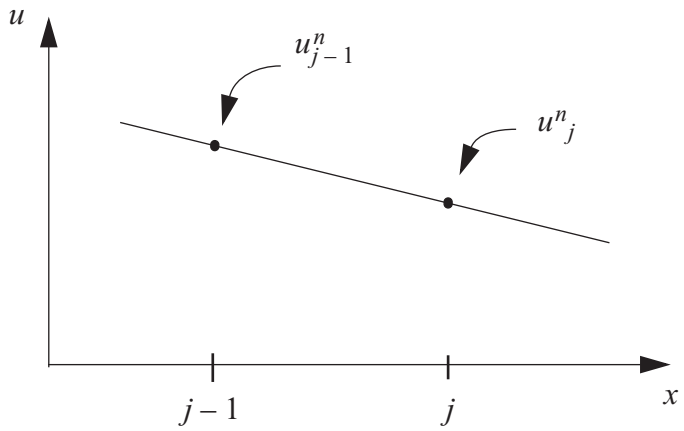

Figure 2.6: Diagram illustrating the concepts of interpolation and extrapolation. See text for details.

For $0 \le \mu \le 1$ we have *interpolation*. For $\mu < 0$ or $\mu > 1$ we have *extrapolation*. Note that for the case of interpolation, u_j^{n+1} will be intermediate in value between u_{j-1}^n and u_j^n. For instance, if u_{j-1}^n and u_j^n are both ≥ 0, then u_j^n will also be ≥ 0. For the case extrapolation, u_j^{n+1} will lie outside the range of u_{j-1}^n and u_j^n.

We use both interpolation and extrapolation extensively in this course.

2.8 Stability

We now investigate the behavior of the discretization error $\left|u_j^n - u(j\Delta x, n\Delta t)\right|$ as n increases, for fixed Δx and Δt. Does the error remain bounded for any initial condition? If so the scheme is said to be stable; otherwise it is unstable.

In most physical problems the true solution is bounded, at least for finite t, so that the solution of the scheme is bounded if the scheme is stable.

There are at least three ways in which the stability of a scheme can be tested. These are: 1) the *direct method*, 2) the *energy method*, and 3) *von Neumann's method*.

As an illustration of the direct method, consider the upstream scheme, as given by (2.84). Note that u_j^{n+1} is a weighted mean of u_j^n and u_{j-1}^n. If $0 \le \mu \le 1$ [the necessary condition for convergence according to (2.80)], we may write

$$\left|u_j^{n+1}\right| \le \left|u_j^n\right|(1-\mu) + \left|u_{j-1}^n\right|\mu \ . \tag{2.88}$$

Therefore,

$$max_{(j)}\left|u_j^{n+1}\right| \le max_{(j)}\left|u_j^n\right|(1-\mu) + max_{(j)}\left|u_{j-1}^n\right|\mu \ , \tag{2.89}$$

or if we assume that $max_{(j)}\left|u_j^n\right| = max_{(j)}\left|u_{j-1}^n\right|$, then

$$max_{(j)}\left|u_j^{n+1}\right| \le max_{(j)}\left|u_j^n\right| \ , \quad \text{provided that } 0 \le \mu \le 1. \tag{2.90}$$

We have shown that for $0 \le \mu \le 1$ the solution u_j^n remains bounded for all time. Therefore, $0 \le \mu \le 1$ is a *sufficient* condition for stability. For this scheme, the condition for stability has turned out to be the same as the condition for convergence. In other words, if the scheme is convergent it is stable, and vice versa.

This conclusion is actually obvious from (2.84), because when $0 \le \mu \le 1$, u_j^{n+1} is obtained by linear interpolation *in space*, from the available u_j^n to the point $x = j\Delta x - c\Delta t$. This is reasonable, since in advection the time rate of change at a point is closely related to the spatial variations in the neighborhood of that point.

Note that in the true solution of the differential equation for advection, the maxima and minima of u never change. They are just carried along to different spatial locations. So, for the exact solution, the equality in (2.90) would hold.

The direct method is not very widely applicable. It becomes intractable for complex schemes.

The second method, the energy method, is more widely applicable, even for some nonlinear equations. We illustrate it here by means of application to the scheme (2.75). With this method we ask: "Is $\sum_j (u_j^n)^2$ bounded after an arbitrary number of time steps?" Here the summation obviously must be taken over a finite number of grid points. This is not an important limitation because in practice we are always dealing with a finite number of grid points. If the sum is bounded, then each u_j^n must also be bounded. Whereas in the direct method we checked $max_{(j)}\left|u_j^{n+1}\right|$, here in the energy method we check $\sum_j (u_j^n)^2$. The two approaches are therefore somewhat similar.

Returning then to (2.84), squaring both sides, and summing over the domain, we obtain

$$\sum_j (u_j^{n+1})^2 = \sum_j [(u_j^n)^2(1-\mu)^2 + 2\mu(1-\mu)u_j^n u_{j-1}^n + \mu^2(u_{j-1}^n)^2]$$
$$= (1-\mu)^2\sum_j (u_j^n)^2 + 2\mu(1-\mu)\sum_j u_j^n u_{j-1}^n + \mu^2\sum_j (u_{j-1}^n)^2 \quad . \tag{2.91}$$

For simplicity, suppose that u is periodic in x, and consider a summation over one complete cycle. Then

$$\sum_j (u_{j-1}^n)^2 = \sum_j (u_j^n)^2 \, . \tag{2.92}$$

We note that

$$\text{if } \sum_j u_j^n u_{j-1}^n < 0 \text{ then } \sum_j u_j^n u_{j-1}^n < \sum_j (u_j^n)^2 \text{; and} \tag{2.93}$$

$$\text{if } \sum_j u_j^n u_{j-1}^n > 0 \text{ then } \sum_j u_j^n u_{j-1}^n \le \sum_j (u_j^n)^2 \, . \tag{2.94}$$

To derive (2.94) we have used Schwartz's inequality, i.e.,

$$\left(\sum_j a_j\right)^2\left(\sum_j b_j\right)^2 \le \left(\sum_j a_j^2\right)\left(\sum_j b_j^2\right) \tag{2.95}$$

for any sets of a s and b s, and (2.92), i.e.

$$\left[\sum_j u_j^n u_{j-1}^n\right]^2 \le \sum_j (u_j^n)^2 \sum_j (u_{j-1}^n)^2 = \left[\sum_j (u_j^n)^2\right]^2 . \tag{2.96}$$

Use of (2.92), (2.93) and (2.94) in (2.91) gives

$$\sum_j (u_j^{n+1})^2 \le [(1-\mu)^2 + 2\mu(1-\mu) + \mu^2]\sum_j (u_j^n)^2 \, , \tag{2.97}$$

provided that $\mu(1-\mu) \ge 0$, which is equivalent to $0 \le \mu \le 1$. The quantity in square brackets in (2.97) is equal to 1. We conclude that

$$\sum_j (u_j^{n+1})^2 \le \sum_j (u_j^n)^2 \, , \quad \text{provided that } 0 \le \mu \le 1. \tag{2.98}$$

As with the direct method, we conclude that $0 \le \mu \le 1$ is a sufficient condition for the scheme to be stable.

A very powerful tool for testing the stability of linear partial difference equations with constant coefficients is von Neumann's method. It is the method that will be used most often in this course. Solutions to linear partial differential equations can be expressed as superposition of waves, by means of Fourier series. Von Neuman's method simply tests the stability of each Fourier component.

To illustrate von Neumann's method, we return first to the advection equation, (2.69):

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \ . \tag{2.99}$$

We assume for simplicity that the domain is periodic. First, we look for a solution with the wave form

$$u(x, t) = Re[\hat{u}(t)e^{ikx}] \ , \tag{2.100}$$

where $|u(t)|$ is the amplitude of the wave. Here we consider a single wave number, for simplicity, but in general we could replace the right-hand side of (2.100) by a sum over all relevant wave numbers. Using (2.100), (2.69) becomes

$$\frac{du}{dt} + ikc\,\hat{u} = 0 \ . \tag{2.101}$$

By Fourier expansion we have converted the partial differential equation (2.69) into an ordinary differential equation, (2.101), whose solution is

$$\hat{u}(t) = \hat{u}(0)e^{-ikct} \ , \tag{2.102}$$

where $u(0)$ is the initial value of u. The solution to (2.69) is, from (2.100),

$$u(x, t) = Re[\hat{u}(0)e^{ik(x-ct)}] \ . \tag{2.103}$$

Note that (2.103) is a valid solution only for c = constant.

For a finite difference equation, we use in place of (2.100)

$$u_j^n = Re[\hat{u}^{(n)}e^{ikj\Delta x}] \ . \tag{2.104}$$

Then $|\hat{u}^{(n)}|$ is the amplitude of the wave at time-level n. Note that the shortest resolvable wave, with $L = 2\Delta x$, has $k\Delta x = \pi$, while longer waves have $k\Delta x < \pi$, so there is never any need to consider $k\Delta x > \pi$. *Define* λ, which may be complex, by

$$\hat{u}^{(n+1)} \equiv \lambda\hat{u}^{(n)} \ . \tag{2.105}$$

Then $\left|\hat{u}^{(n+1)}\right| = |\lambda|\left|\hat{u}^{(n)}\right|$. We call λ the "amplification factor." As shown below, we can work out the form of λ for a particular finite-difference scheme. In general λ depends on k, so we could write λ_k, but usually we suppress that urge for the sake of keeping the notation simple. Note that λ can also be defined for the exact solution to the differential equation; from (2.102), we simply have $\hat{u}(t+\Delta t) \equiv e^{-ikc\Delta t}\hat{u}(t)$, so that for the differential equation $\lambda = e^{-ikc\Delta t}$. For a particular scheme, we can compare the "exact" λ with the approximate λ defined by (2.105). Note that for the exact advection equation $|\lambda|$ is 1. For other problems, the exact $|\lambda|$ can differ from 1.

From (2.105) we see that after n time steps (starting from $n = 0$) the solution will be

$$\hat{u}^{(n)} = \hat{u}^{(0)}\lambda^n . \tag{2.106}$$

If we require that the solution remains bounded after arbitrarily many time steps, then we need

$$|\lambda| \le 1 . \tag{2.107}$$

This is the condition for the stability of mode k.

To check the stability of a finite-difference scheme, using von Neumann's method, we need to work out $|\lambda|$ for that scheme. We now illustrate the computation of $|\lambda|$ for the upstream scheme, which is given by (2.75). Substituting (2.104) into (2.75) gives

$$\frac{\hat{u}^{(n+1)} - \hat{u}^{(n)}}{\Delta t} + c\left(\frac{1 - e^{-ik\Delta x}}{\Delta x}\right)\hat{u}^{(n)} = 0 . \tag{2.108}$$

Notice that the true advection speed, c, is multiplied, in (2.108), by the factor $\left(\frac{1 - e^{-ik\Delta x}}{\Delta x}\right)$. Comparing with (2.101), we see that this factor is taking the place of ik in the exact solution. As $\Delta x \to 0$, the factor reduces to ik. This is a hint that the scheme gives an error in the advection speed. We return to this point later.

For now, we use the definition of λ, i.e. (2.105),together with (2.108), to infer that

$$\lambda = 1 - \mu(1 - \cos k\Delta x + i \sin k\Delta x) . \tag{2.109}$$

Note that λ is complex. Taking the modulus of (2.109), we obtain

$$|\lambda|^2 = 1 + 2\mu(\mu - 1)(1 - \cos k\Delta x) . \tag{2.110}$$

At $\mu = \frac{1}{2}$, for example, (2.110) reduces to

$$|\lambda|^2 = \frac{1}{2}(1 + \cos k\Delta x) \ . \tag{2.111}$$

According to (2.111), the amplification factor $|\lambda|$ depends on the wave number, k. Using $k \equiv \frac{2\pi}{2\Delta x}$ for $L = 2\Delta x$, $k = \frac{\pi}{2\Delta x}$ for $L = 4\Delta x$, etc., the various curves shown in Fig. 2.7 can be constructed. We see clearly that this scheme damps for $0 < \mu < 1$ and is unstable for $\mu < 0$ and $\mu > 1$.

Although λ depends on k, it does not depend on x (i.e., on j) or on t (i.e., on n). Why not? The reason is that our "coefficient" c, has been assumed to be independent of x and t. Of course, in realistic problems the advecting current varies in both space and time. We normally apply von Neumann's method to idealized versions of our equations, in which the various coefficients, such as c, are treated as constants. As a result, von Neumann's method can "miss" instabilities that arise from variations of the coefficients. The energy method does not suffer from this limitation. It is very important to understand that von Neumann's method can only analyze the stability of a linearized version of the equation. In fact, the equation has to be linear and with constant coefficients. This is an important weakness of the method, because the equations used in numerical models are typically nonlinear and/or have spatially variable coefficients -- if this were not true we would solve them analytically! The point is that von Neumann's method can sometimes tell us that a scheme is stable, when in fact it is unstable. In such cases, the instability arises from nonlinearity and/or through the effects of spatially variable coefficients. This kind of instability will be discussed in Chapter 6. If von Neumann's method tells us that a scheme is *unstable*, then it is unstable.

As mentioned above, in general, the solution u_j^n can be expressed as a Fourier series. For simplicity, let us assume that the solution is periodic in x with period L_0. Then u_j^n can be written as

$$u_j^n = Re\left[\sum_{m=-\infty}^{\infty} \hat{u}_m^{(n)} e^{imk_0 j\Delta x}\right] = Re\left[\sum_{m=-\infty}^{\infty} \hat{u}_m^{(0)} e^{imk_0 j\Delta x} (\lambda_m)^n\right], \tag{2.112}$$

where $k \equiv mk_0$

$$k_0 \equiv \frac{2\pi}{L_0}, \tag{2.113}$$

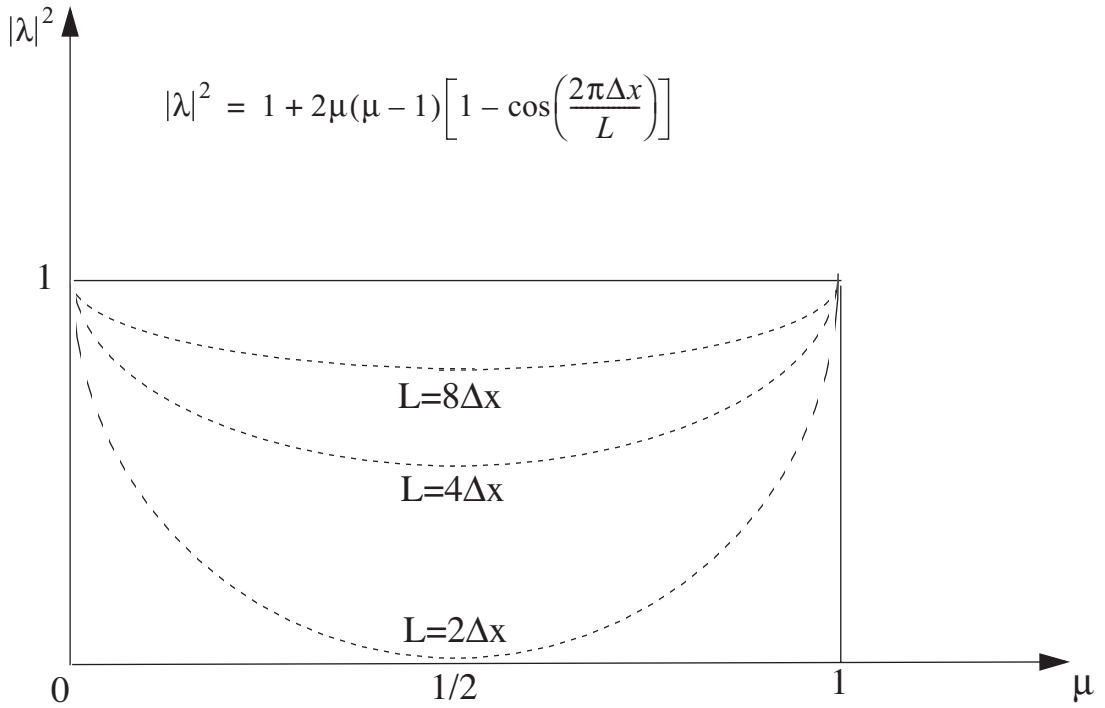

Figure 2.7: The amplification factor for the upstream scheme, plotted for three different wave lengths.

and m is an integer, which is analogous to what we call the "zonal wave number" in large-scale dynamics. In (2.112), the summation has been formally taken over all integers, although of course only a finite number of m's could be used in a real application. Note that $|\lambda_m|$ is the amplification factor for mode m. We have

$$\begin{aligned} |u_j^n| &\le \left| \sum_{m=-\infty}^{\infty} \hat{u}_m^{(0)} e^{imk_0 j\Delta x} (\lambda_m)^n \right| \\ &\le \sum_{m=-\infty}^{\infty} \left| \hat{u}_m^{(0)} e^{imk_0 j\Delta x} (\lambda_m)^n \right| \\ &= \sum_{m=-\infty}^{\infty} \left| \hat{u}_m^{(0)} \right| |\lambda_m|^n . \end{aligned} \tag{2.114}$$

If $|\lambda_m| \le 1$ is satisfied for all m, then

$$\left|u_j^n\right| \leq \sum_{m=-\infty}^{\infty} \left|\hat{u}_m^{(0)}\right| . \tag{2.115}$$

Therefore, $\left|u_j^n\right|$ will be bounded provided that $\sum_{m=-\infty}^{\infty} \hat{u}_m^{(0)} e^{imk_0 j\Delta x}$, which gives the initial condition, is an absolutely convergent Fourier series. *This shows that* $\left|\lambda_m\right| \leq 1$ *for all m is sufficient for stability*. It is also necessary, because if $\left|\lambda_m\right| > 1$ for a particular m, say $m = m_1$, then the solution for the initial condition $u_{m_1} = 1$ and $u_m = 0$ for all $m \neq m_1$ is unbounded.

From (2.109), λ_m for the upstream scheme is given by

$$\lambda_m = 1 - \mu(1 - \cos mk_0\Delta x + i \sin mk_0\Delta x) . \tag{2.116}$$

The amplification factor is

$$\left|\lambda_m\right| = [1 + 2\mu(1 - \cos mk_0\Delta x)(\mu - 1)]^{\frac{1}{2}} . \tag{2.117}$$

From (2.117) we can show that $\left|\lambda_m\right| \leq 1 \leq$ holds for all m, if and only if $\mu(\mu - 1) \leq 0$, or $0 \leq \mu \leq 1$. This is the necessary and sufficient condition for the stability of the scheme.

Finally, we can test stability using the "matrix method[1]," which is really just von Neumann's method with more general boundary conditions. The upstream scheme given by (2.75) (or by (2.112)) can be written in matrix form as

[1] Developed by K. Reeves.

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \cdots \\ u_{j-1}^{n+1} \\ u_j^{n+1} \\ u_{j+1}^{n+1} \\ \cdots \\ u_{J-1}^{n+1} \\ u_J^{n+1} \end{bmatrix} = \begin{bmatrix} 1-\mu & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & \mu \\ \mu & 1-\mu & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & \mu & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 1-\mu & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & \mu & 1-\mu & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & \mu & 1-\mu & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 & \mu & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 1-\mu & 0 \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & \mu & 1-\mu \end{bmatrix} \bullet \begin{bmatrix} u_1^n \\ u_2 \\ \cdots \\ u_{j-1}^n \\ u_j^n \\ u_{j+1}^n \\ \cdots \\ u_{J-1}^n \\ u_J^n \end{bmatrix}, \tag{2.118}$$

or

$$[u_j^{n+1}] = [A][u_j^n], \tag{2.119}$$

where $[A]$ is the matrix written out on the right-hand side of (2.118). In writing (2.118), the cyclic boundary condition

$$u_1^{n+1} = (1-\mu)u_1^n + \mu\, u_J^n \tag{2.120}$$

has been assumed. Recall from the definition of λ that $u_j^{n+1} = \lambda u_j^n$. This can be written in matrix form as

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \cdots \\ u_{j-1}^{n+1} \\ u_j^{n+1} \\ u_{j+1}^{n+1} \\ \cdots \\ u_{J-1}^{n+1} \\ u_J^{n+1} \end{bmatrix} = \begin{bmatrix} \lambda & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & \lambda & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & \lambda & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & \lambda & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 & \lambda & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & \lambda & 0 \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & \lambda \end{bmatrix} \bullet \begin{bmatrix} u_1^n \\ u_2 \\ \cdots \\ u_{j-1}^n \\ u_j^n \\ u_{j+1}^n \\ \cdots \\ u_{J-1}^n \\ u_J^n \end{bmatrix}, \tag{2.121}$$

or

$$[u_j^{n+1}] = \lambda[I][u_j^n], \tag{2.122}$$

where $[I]$ is the identity matrix. Comparing (2.119) and (2.122), we see that

$$([A] - \lambda[I])[u_j^n] = 0, \tag{2.123}$$

and this equation must hold regardless of the values of the u_j^n. It follows that the amplification factors, λ, are the *eigenvalues* of $[A]$, obtained by solving

$$|[A] - \lambda[I]| = 0. \tag{2.124}$$

For the current example, we can show that

$$\lambda = 1 - \mu\left(1 - e^{i\frac{2m\pi}{J}}\right), \quad m = 0, 1, 2, \ldots J-1. \tag{2.125}$$

This has essentially the same form as (2.109), and so we find that $0 \le \mu \le 1$ is the stability condition. *The advantage of the matrix method is that the boundary conditions can be directly included in the stability analysis, as in the example above.*

2.9 The effects of increasing the number of grid points

Recall that the upstream scheme

can be written as

$$u_j^{n+1} = u_j^n(1-\mu) + u_{j-1}^n\mu. \tag{2.126}$$

Also recall that for this scheme the amplification factor, λ, is

$$\lambda = 1 - \mu(1 - \cos k\Delta x + i\sin k\Delta x), \tag{2.127}$$

so that

$$|\lambda|^2 = 1 + 2\mu(\mu-1)(1-\cos k\Delta x). \tag{2.128}$$

The stability criterion is

$$0 \le \mu \le 1. \tag{2.129}$$

When (2.129) is satisfied, we have

$$|\lambda| \le 1. \tag{2.130}$$

Consider what happens when we increase the number of grid points, while keeping the domain size, D, the wind speed, c, and the wave number, k, of the advected signal the same. We consider grid spacing Δx, such that

$$D = J\Delta x . \tag{2.131}$$

As we decrease Δx, we increase J correspondingly, so that D remains the same, and

$$k\Delta x = \frac{kD}{J} . \tag{2.132}$$

Substituting this into (2.128), we obtain

$$|\lambda|^2 = 1 + 2\mu(\mu - 1)\left[1 - \cos\left(\frac{kD}{J}\right)\right] . \tag{2.133}$$

In order to maintain computational stability, we keep μ fixed as Δx decreases, so that

$$\begin{aligned} \Delta t &= \frac{\mu\Delta x}{c} \\ &= \frac{\mu D}{cJ} . \end{aligned} \tag{2.134}$$

The time required for the air to flow through the domain is

$$T = \frac{D}{c} . \tag{2.135}$$

Let N be the number of time steps needed for the air to flow through the domain, so that

$$\begin{aligned} N &= \frac{T}{\Delta t} \\ &= \frac{D}{c\Delta t} \\ &= \frac{D}{\mu\Delta x} \\ &= \frac{J}{\mu} . \end{aligned} \tag{2.136}$$

The total amount of damping that "accumulates" as the air moves across the domain is measured by

$$|\lambda|^N = (|\lambda|^2)^{N/2}$$
$$= \left\{1 - 2\mu(1-\mu)\left[1 - \cos\left(\frac{kD}{J}\right)\right]\right\}^{\frac{J}{2\mu}}. \tag{2.137}$$

Here we have used (2.133) and (2.136).

As we increase the resolution, J increases. This causes the cosine factor in (2.137) to approach 1, which weakens the damping associated with $|\lambda| < 1$; but on the other hand it also causes the exponent in (2.137) to increase, which strengthens the damping. Which effect dominates is not obvious. The answer can be seen in Fig. 2.8. Increasing the resolution leads

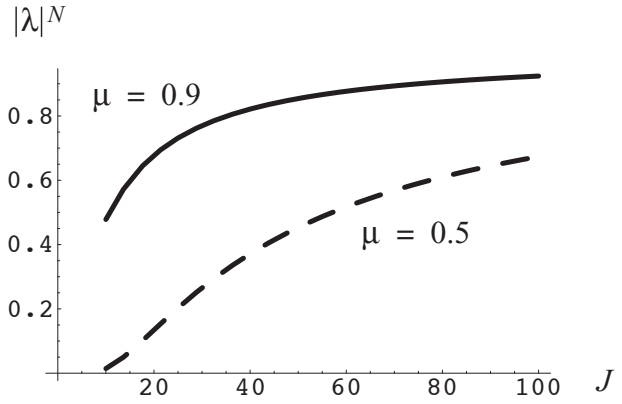

Figure 2.8: "Total" damping experienced by a disturbance crossing the domain, as a function *J*, the number of grid points across the domain. Here we have assumed that $D/L = 2$.

to less total damping, even though the number of time steps needed to cross the domain increases.

2.10 Summary

Suppose that we are given a non-linear partial differential equation and wish to solve it by means of a finite difference approximation. The usual procedure would be as follows:

- **Check truncation error.** Normally this is done by means of a Taylor series expansion. We are concerned with the lowest powers of the space and time grid-interval in the expansion of the independent variables.
- **Check linear stability** for a simplified (linearized, constant coefficients) version of the equation. Von Neumann's method is often used here.
- **Check nonlinear stability**, if possible. This can be accomplished, in some cases, by using the energy method. Otherwise, empirical tests are needed.

Increased accuracy does not always give a better scheme. For example, consider two schemes A and B, such that scheme A is first-order accurate but stable, while scheme B is second-order accurate but unstable. Given such a choice, the less accurate scheme is definitely better.

In general, "good" schemes have the following properties, among others:

- High accuracy.
- Stability.
- Simplicity.
- Computational economy.

Later we will extend this list to include additional desirable properties.

Almost always, the design of a finite-difference scheme is an exercise in trade-offs. For example, a more accurate scheme is usually more complicated and expensive than a less accurate scheme. We have to ask whether the additional complexity and computational expense are justified by the increased accuracy. The answer depends on the particular application.

Problems

1. Prove that a finite-difference scheme with errors of order n gives exact derivatives for polynomial functions of degree n or less. For example, a first-order scheme gives exact derivatives for linear functions.

2. Choose a simple differentiable function $f(x)$ that is *not* a polynomial. Find the exact numerical value of $\frac{df}{dx}$ at a particular value of x, say x_1. Then choose

   a) a first-order scheme, and

   b) a second-order scheme

   to approximate $\left(\frac{df}{dx}\right)_{x = x_1}$. Plot the log of the absolute value of the *error* of these approximations as a function of the log of Δx. By inspection of the plot, verify that the errors of the schemes decrease, at the expected rates, as Δx decreases.

3. Program the upstream scheme on a periodic domain with 100 grid points. Give a sinusoidal initial condition with a single mode such that exactly four wavelengths fit in the domain. Integrate for μ= -0.1, 0.1, 0.5, 0.9, 1 and 1.1. In each case, take enough time steps so that in the exact solution the signal will just cross the domain. Discuss your results.

4. **Using the energy method**, determine the stability of the **forward** time scheme as applied to the following pair of equations:

$$\frac{du}{dt} = fv \,, \tag{2.138}$$

$$\frac{dv}{dt} = -fu \,. \tag{2.139}$$

   Note: Solution following the energy method as required does not involve the use of complex variables.

5. Work out the form of the most compact second-order accurate approximation for $\left(\frac{d^2 f}{dx^2}\right)_j$ on a non-uniform grid. Also find the simpler form that applies when the grid is uniform.