PLU 85

Computational modelling flow and transport

Lecture Notes CT wa4340

by G.S. Stelling and N. Booij

Delft University of Technology

Faculty of Civil Engineering and Geosciences

	Aug. 1999			***320085***		***CT wa 4340***		

Contents:

Chapter 1 Preliminaries

1.1. Introduction

In this chapter the general outline of the course, wa4340, will be explained. Computational hydraulics is supposed to be a an applied science aiming at the simulation, with computers, of various processes as they take place in areas such as seas, estuaries, rivers, channels, lakes, etc.. An important prerequisite for simulation is the availability of a computational model. The construction of computational models is an important subject of this course. Computational models are getting more and more complicated, not only from the physical point of view, but also from the point of view of data processing. Models can be regarded as information systems, and some authors are redefining the discipline as "hydro informatics". Within the framework of this course however we prefer the expression "computational hydraulics". Whatever expression is used, the discipline is not an independent development, it is rather a synthesis of various disciplines such as physics, mathematical physics, mathematics, numerical analysis and informatics.

Computational models can be considered from various points of view. In section 1.2 we view modelling as a series of mappings from a certain part of the real world, via abstract number spaces, onto another part of the real world, namely the computer. in section 1.4.

1.2. What is a computational model?

A computational model in the sense of this section is supposed to be a mapping of a part of the real world onto a computer, see figure 1.1. In this way similarity with respect to various aspects, such as dynamics, shape or structure, between the real world and the computer is obtained. The most ideal situation would be if this mapping could be considered as an "isomorphism". The word isomorphism applies when two complex structures can be mapped onto each other, in such a way that to each part of that structure there is a corresponding part in the other structure, where corresponding means that the two parts play similar roles in their respective structures, see Hofstadter 1979. More precise, mathematical, definitions exist, see e.g. Roman, 1975. In general a computational model can never considered to be an isomorphism in the sense as described above because:

I A model will never describe every aspect of the real world.

II A model will contain artifacts that have no corresponding pattern in the real world.

To get insight into the shortcomings of a model we consider a model as a result of a series of mappings from one space to another, see figure 1.3.

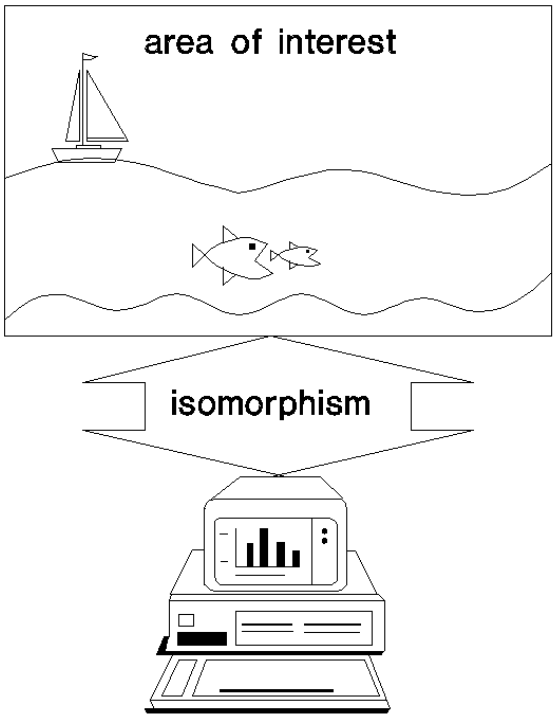


Figure 1.1

The first space contains that part of the real world of which a model is to be set up. In the sense of this course only waters such as rivers, lakes, estuaries and seas are to be considered, see figure 1.2.

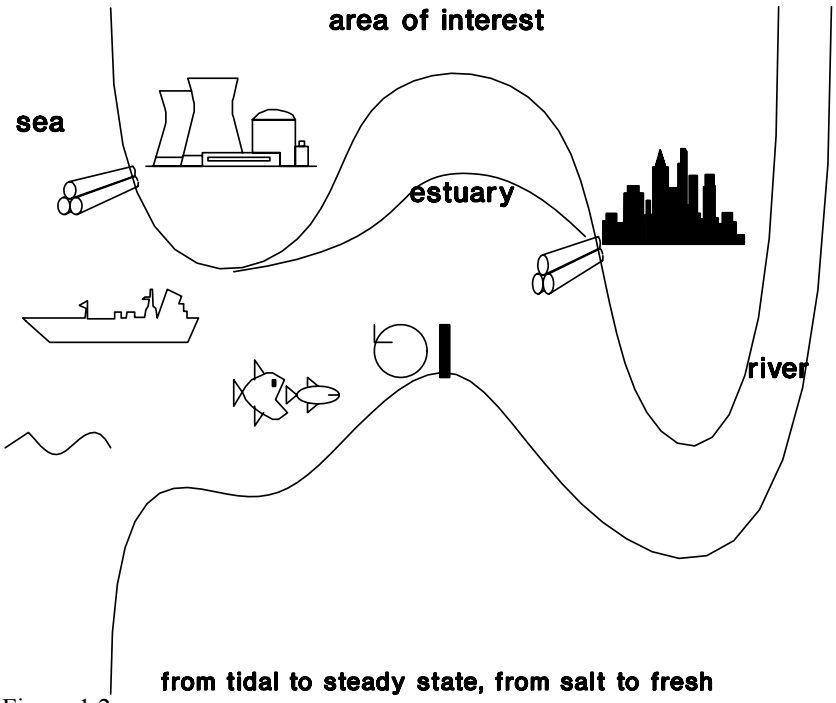


Figure 1.2

The processes that are considered are mainly flow and transport. To quantify processes, or to describe the "state of the system", we must define variables, such as velocity, water level, salinity, temperature, pressure, etc. to which we can assign numerical values. It is supposed that these processes are governed by physical laws such as conservation of mass and momentum. These laws can be expressed as:

$$\textit{Rate of change} = \textit{Input} - \textit{Output} \tag{1.1}$$

Conservation laws, can be represented as differential equations. Within the framework of this course we will consider continuity equations, momentum equations and equations for the transport of scalar quantities such as salinity and heat. Once differential equations are obtained with symbols that are connected to things in the real world we have created a mapping from the real world to a symbolic mathematical space. This is the second space that we consider:

In this space real numbers are represented as symbols. The space is structured by mathematical notions such as continuity, measurability, etc., see Yosida, 1963 or Roman 1975. A PDE can considered as a map from a domain to a co-domain that is given implicitly. Internally this space is governed by mathematical laws. For simulation only "well-posed problems" are meaningful, i.e. PDE's that, including boundary conditions, have unique solutions that depend continuously on the boundary conditions, see Garabedian, 1964. This mathematical space is governed by mathematical laws and not by physics. Well-posedness for example is a mathematical notion that follows only from fulfilling basic mathematical rules. The implicitness of the mapping between various sets in this mathematical space gives us a (initial and boundary value) problem. For simulation purposes this mapping must be given explicitly, in other words the PDE must be solved. This is only possible for rather trivial, mostly linear, problems. In general we can only solve this problem in an approximative sense. For this aim we have to consider another space:

This space is called a discrete space or a grid space. In this space everything is discrete, real numbers still exist but continuity and differentiability do not exist. Functions are represented as series of points in the discrete space. The only operations are addition, subtraction, multiplication, division and similar operations. PDE's are replaced by (or mapped onto) recurrence relations. This recurrent relations can be defined for a regular or an irregular grid. The mapping with regular grids in general takes place with by means finite difference methods while for irregular grids finite element methods are used. The solution of the recurrence relations is not always trivial, sometimes it is the case, in that case the numerical method is called explicit. In other, implicit, cases the recurrent relations are to be solved with the aid of various techniques such as matrix inversion methods or non-linear equation solvers. Finally,

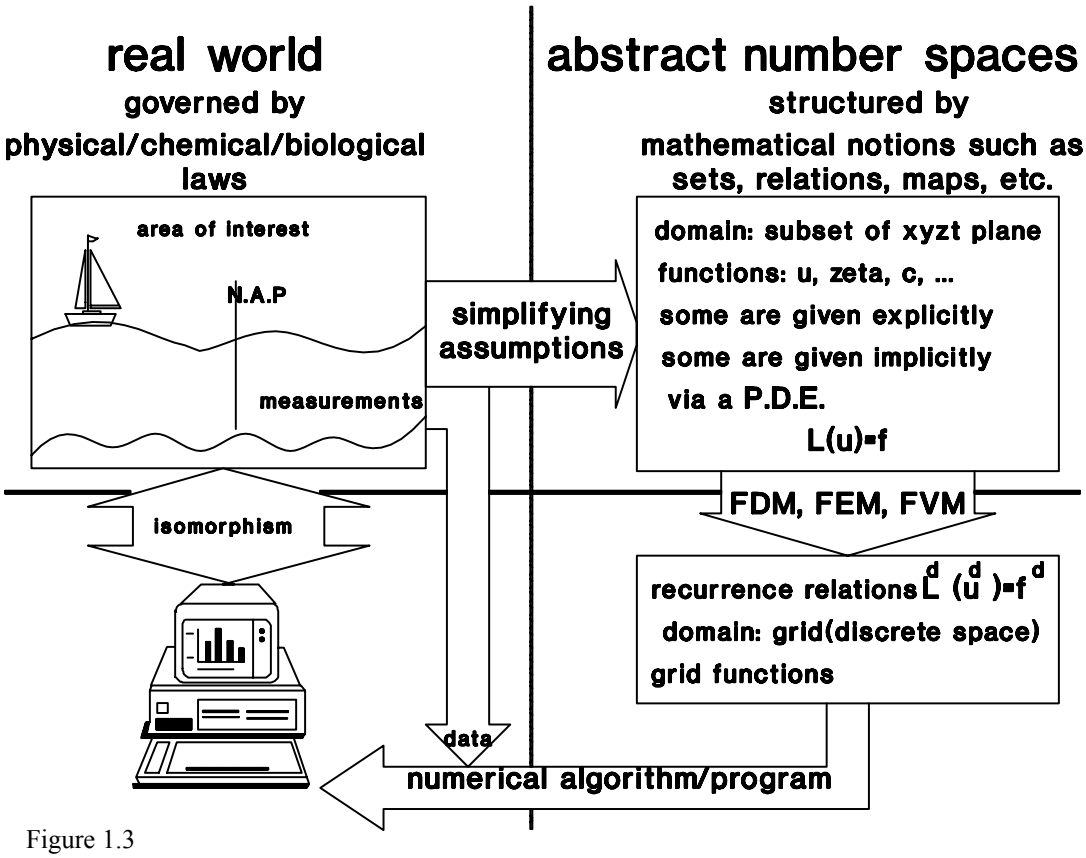


Figure 1.3

by a mixture of various numerical techniques, a numerical solution can be obtained in principle. At this stage however everything is still in symbolic form. To obtain actual numbers we need the computer:

The computer is the final stage of our sequence. The mixture of numerical methods can be synthesized into an algorithm described in a symbolic algorithmic language, such as FORTRAN, PASCAL etc., with which the computer can be controlled. At present algorithms are a part of a simulation system and they are not to be programmed for each new application. Only input parameters are to be changed. A computer as such is a part of the real world, its behaviour is also controlled by physical conservation laws, similar to fluid flow, see Potter, 1973. The quantities describing the state of its electronic circuits are something like voltage, current, etc. Digital semi-conductor technology however enables us to consider voltages as (digital) numbers and to perform elementary operations such as multiplication, addition, etc., at very high speeds. Graphical devices as part of computer systems permit us to represent numbers as graphs leading sometimes to animations that make the similarity between a computer model and the real world visible.

One of the main objectives of computational hydraulics is to obtain simulations of dynamical processes of flow and transport in open water bodies as detailed and as accurately as required within a predefined framework of specifications. Knowledge of aspects that control this accuracy is therefore of crucial importance, it will play an important role during this course. At each step of our mapping cycle errors, or limitations, are introduced. We will give some examples of restrictions that are introduced at each transition between various spaces:

The part of the real world that we consider is governed by conservation laws such as conservation of mass and momentum. Via assumptions we can express these laws as symbolic equations. The most important assumption for example is the hypothesis that water is a continuum which means perfectly continuous in structure. This assumption is not supposed to be very restrictive, in general however assumptions are a source of errors or limitations to the generality of the applicability of a model or as depicted by Simon, 1969: "A simulation is no better than the assumptions built into it." We will illustrate with a few examples, how only by assumptions, physical phenomena can be expressed as differential equations:

Consider a closed surface S whose position is fixed relative to co-ordinate axes and which encloses a volume V entirely occupied by fluid. Conservation mass for this volume V can then be expressed as:

$$\frac{d}{dt}\int \rho \, dV = -\int \rho \vec{u} \cdot \vec{n} \, dS \tag{1.2}$$

Where ρ is the density of the fluid while the left hand side of this equation expresses the rate of change or storage in V and the right hand side expresses the net rate at which mass is flowing outwards across the surface S. If we assume differentiability of the flow field we can apply the famous Gauss law or divergence theorem given by:

$$\int \vec{u} \cdot \vec{n} \, dS = \int \nabla \cdot \vec{u} \, dV \tag{1.3}$$

Where $\vec{n}$ denotes the outward normal vector. After application of divergence theorem to (1.2) we obtain:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \left(\rho \vec{u} \right) = 0 \tag{1.4}$$

Let us assume that the fluid is incompressible, this means that the density of the fluid is not affected by pressure changes. From this it follows that the rate of change of ρ following the motion is zero, that is:

$$\frac{D\rho}{Dt} = 0 \tag{1.5}$$

The mass-conservation equation then takes the simple form:

$$\nabla \cdot \vec{u} = 0 \tag{1.6}$$

This relation is less complicated than (1.2) but it has lost generality. Again for water this assumption is not a real limitation. For simulation of flow not only do we need a continuity equation but also momentum equations based upon Newtonian mechanics. In the form of (1.1), in this case conservation of momentum, these equations are given by:

$$\int \frac{\partial \left(u_i \rho \right)}{\partial t} = -\int \rho \, u_i u_j n_j \, dS + \int F_i \, \rho \, dV + \int \sigma_{ij} n_j \, dS \tag{1.7}$$

where F_i is a component of a body force vector (in our case the earth's gravitational field) and $\sigma_{i,j}$ a component of a stress tensor, see e.g. Batchelor, 1967. As such these equations are of little use for us, first we need expressions for $\sigma_{i,j}$. After this one obtains the so-called Navier-Stokes equations. For our areas of interest, where the flow is always turbulent, these equations can neither be solved nor approximated. By introducing several, mostly simplifying, assumptions we will arrive at the so-called "shallow water equations". These equations can be given in 3,2 or 1 dimensional form. The derivation will be provided in section 3. Various assumptions, that do introduce real restrictions on the applicability, are needed to obtain or complete these equations, such as:

Hydrostatic pressure
Boussinesq approximation
Turbulent closure assumptions, the concept of eddy viscosity
Laws of the wall, e.g. perfect-slip
Assumptions that the flow can be considered as 2 or 1 dimensional
Limited domains including open (water/water) boundary conditions

Not only momentum equations are based upon assumptions limiting the generality, also transport equations have there restrictive concepts such as dispersion formulations. In general transport of matter is described by:

$$\frac{\partial}{\partial t}\int C \, dV = -\int \vec{f} \cdot \vec{n} \, dS \tag{1.8}$$

where $\vec{f}$ denotes a flux vector. The divergence theorem yields:

$$\frac{\partial C}{\partial t} + \nabla \cdot \vec{f} = 0 \tag{1.9}$$

The fluxes might contain both advective and diffusive transport of matter, which we can denote as:

$$f_i = u_i C + D_{ij} \frac{\partial C}{\partial x_j} \tag{1.10}$$

where D_{ij} denotes a transport coefficient. Especially this coefficient is, for turbulent, 2 or 1 dimensional flows, subject to assumptions limiting the generality. It is often assumed that:

$$D_{ij} = -D\delta_{ij} \tag{1.11}$$

In some cases spatial variation of the concentration is neglected at all. Then we obtain so-called "box models", which are initial value problems of ordinary differential equations. Consider the following example:

Consider a lake with a volume V as a section of a river. The river enters the lake with a discharge Q_1 and leaves the river with a discharge Q_0. The lake contains a dissolved matter with a concentration C. Since C is a biological substance, its mass decreases at a rate VC/T_r where T_r is the time scale for degradation. The inflow Q_1 contains a concentration C_1.

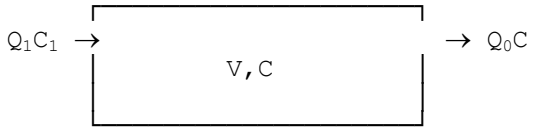


figure 1.4

At this point we introduce the following assumptions:

- V is constant from which it follows that $Q_1=Q_0=Q$
- The concentration C is constant in V from which it follows that the amount of concentration leaving the lake is QC.

Eq. (1.1) then becomes:

$$V\frac{dC}{dt} = Q C_1 - \left(C\frac{V}{T_r} + Q C \right) \tag{1.12}$$

This equation cannot be solved without a given initial value for C given by C_0. Due to this and due to the limited domain a model is not only less than reality, it will also contain effects that will not be found in reality. Unrealistic initial values for example, e.g. constant water levels and zero velocities, will generate oscillations of unrealistic length scales due to the location of artificial boundary conditions.

After having defined a boundary value problem which is supposed to be well-posed this problem has to be mapped onto a set of recurrent relations or difference equation. For example a recurrent relation for the approximation of our simple box model may look like:

$$\frac{C^{k+1} - C^k}{\Delta t} = \frac{Q}{V} C_1 - \left(\frac{1}{T_r} + \frac{Q}{V} \right) C^k \tag{1.13}$$

where Δt denotes the grid size or the time step.

The solution of these recurrent relations is only an approximation of the true solution we are looking for. The present state of the art of numerical analysis does not enable us to find numerical solutions with guaranteed error bounds, and therefore awareness of these errors is of crucial importance. A useful numerical technique must at least fulfil requirements derived from notions such as consistency, convergence and stability, see Richtmyer and Morton, 1968. These notions will be described in section 5 of this course. Fulfilling these requirements will not guarantee numerical solutions without errors. It only secures that given a sequence of grids of which the grid size tends to zero in the limit that related to this sequence there is a sequence of numerical solutions of which the error, i.e. the difference with the

true solution, tends to zero. In general the grid sizes that are chosen are far from being sufficiently small. The choice is often more based upon considerations of available computational resources than based upon considerations of sufficient accuracy. Varying the grid size might be a practical approach to gain an indication of the numerical error. In general it seems reasonable to assume that numerical solutions will always contain errors. Some errors lead to non-realistic solutions from a physical point of view. An example is a numerical solution that produces mass instead of conserving it, despite of an underlying conservation law. By applying special numerical techniques these type of errors can be avoided. Another example involves negative solutions for dissolved matters. Numerical techniques exist to guarantee positivity and monotonicity, see e.g. Hirsch, 1991. By applying such techniques numerical solutions can be constructed that look reasonable from a physical point of view, however by no means this implies smaller errors.

Finally the algorithm as part of a computational system will be executed on a computer, we will leave the abstract symbolic world, symbols will be replaced by actual numbers. Again we encounter error sources:

Measurement errors

The model that we apply assumes certain variables to be as computed while others are assumed to be as given. An example of a variable that is often to be supposed as given is the depth. Depth values are obtained by measurements that contain errors. Moreover the location were measurements are given are different from the grid point locations. Interpolation is needed, also leading to errors.

Round off errors

The computer performs calculations that are affected by round off errors. This error arises because the machine hardware can only represent a subset of the real numbers, see Golub and Van Loan, 1983.

Programming errors

Computational systems contain programming errors, increasingly more complex systems will have increased risks on programming errors, see Van Vliet, 1988.

1.3. Outline of the course

The remainder of the course deals with the various aspects mentioned in this introductory chapter in more detail. Chapter 1 is again introductory, it briefly focuses on the question of the role of models within engineering activities. Chapter 2 describes the basic equations, derived from balance principles, that are mostly used, often in simplifies form, throughout this course. Chapter 3 focuses on linear difference equations and ordinary differential equations. Chapter 4 is dealing with partial difference equations while the chapters 5 and 6 describe the practical application of the DUFLOW package.

Chapter 2 Derivation of equations using balance principles

In many branches of physics differential equations of phenomena are derived using balance principles. A balance equation describes the conservation of a physical quantity in a certain control volume; this does not necessarily mean that we use balance principles only if a quantity is exactly conserved; there can be production or destruction of the quantity.

For many physical quantities we have conservation laws, e.g.: mass, momentum, energy, heat etc.

In applying the balance principle we first choose a control volume; this can be very small compared with the total computational region which we consider, or it can be the same as this region. The balance principle states:

$$\frac{dM}{dt} = S_i - S_o + P \qquad (2.1)$$

where M is the quantity of a material in the control volume, S_i is the inflow of the material and S_o the outflow over the boundary of the control volume, and P is the production of the material within the control volume.

Note that destruction is simply negative production. Since there is mention of rate of change we see that we usually deal with time-dependent phenomena. In the rare stationary cases the term on the left-hand side simply is zero.

In this chapter we will first discuss the box models; in a box model the control volume is large, it is equal to the whole region, or the region is divided into a very small number of control volumes. For instance in an ecological study of Lake Balaton (Somlyody and Van Straten, 1986) the entire lake was split into 4 control volumes, in each of which the concentrations of algae and other biological and chemical materials were assumed to be homogeneous.

Later on we will consider computational regions consisting of a channel such as a river or canal stretch, a narrow lake or estuary etc. In that case the control volume is a short section of the channel, so short that the relevant quantities can be considered to be constant in this section.

2.1. Box models

2.1.1. Box models with one state variable

As an example of a box model we consider a lake on which a electricity plant is discharging its excess heat. It is assumed that there is river flowing through the lake. The situation is sketched in figure 2.1.

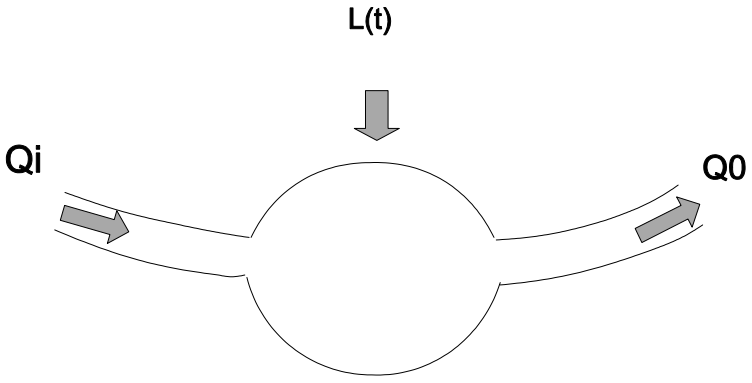


Figure 2.1. Lake with electricity plant.

The conserved quantity in this case is the amount of heat in the lake (measured in kJ, the unit of energy); note that the conserved quantity is not the temperature; the temperature is a density related to the heat; it is the amount of heat per unit mass divided by the specific heat coefficient c_t (which is approximately 4.2 kJ/kg/^{o}C). We introduce the coefficient $\mu = \rho c_t$.

The control volume is the entire lake; it is assumed that the temperature is roughly constant over the lake. Therefore we characterize the temperature in the lake by one variable T(t), which is a function of time. We will use T as the state variable, that is the parameter which we use to characterize the state of the system (within the limitations of the mathematical model); the equation will be formulated in terms of the state variable. Very often the state variable is a density.

When we consider the balance principle we follow the terms mentioned in the previous section:

- The amount of heat in the lake is equal to: μVT, where V is the volume of water in the lake; both V and T are functions of time.
- There is no internal production or destruction of heat in the lake, it is assumed.
- Inflows of heat are the following: the amount of heat discharged by the plant into the lake per unit time: L(t) (dimension kJ/s or kW), the amount of heat brought into the lake by the river: $\mu Q_i T_i$, where Q_i is the flow in the river (dimension m^3/s) times the temperature in the upstream river T_i.
- Outflows are: $Q_o T$, where Q_o is the flow in the river downstream of the lake, and where T is the temperature in this part of the river; this temperature is assumed to be equal to the temperature in the lake itself. A second outflow of heat is the exchange of heat with the atmosphere above the lake; we assume it to be $\mu C_T A(T-T_a)$, where A is the surface of the lake, T_a is the air temperature, and C_T is a heat exchange coefficient.

The whole model for the temperature change thus becomes:

$$\frac{d}{dt}\mu VT = L(t) + \mu Q_i T_i - \mu Q_0 T - \mu C_T A\left(T - T_a\right) \tag{2.2}$$

This is one equation, so we can solve one unknown, the lake temperature in this case. Thus the volume V, the air temperature T_a, the incoming water temperature T_i, the the river inflow Q_i and outflow Q_o must be known, probably from another model or from measurements.

Exercise 2.A:

Determine the dimension of the coefficient C_T. Check that the dimension of all the terms in equation (2.2) is the same.

Exercise 2.B:

Make a model for the determination of V assuming that Q_i is a given function of time and that Q_o is a given function of the water level H in the lake. H can be used as state variable.

The equations for the heat in the lake and for the volume are ordinary differential equations of first order with time as the independent variable. Each such equations requires one initial condition. The equation itself gives only the change of the quantity in the control volume, so additionally we need to know the quantity itself at some instant. We must prescribe the value of the state variable at the time when we start the computation (often designated as t=0), in other words T(0) and H(0) respectively.

Exercise 2.C:

In the example of the temperature model (2.2) assume that V etc. are constant; then the remaining equation is linear in T with constant coefficients. Try to find the general solution of the equation and sketch a graph of the temperature as a function of time.

The most simple box model (one equation, linear, constant coefficients) always leads to an equation of the form:

$$\frac{dC}{dt} + rC = P \tag{2.3}$$

This equation will be used as the prototype equation for box models in chapter 3. Assuming that P is constant its solution is:

$$C(t) = \frac{P}{r} + \left[C(0) - \frac{P}{r}\right]e^{-rt} \tag{2.4}$$

where C(0) is the given initial value; we assume that we start the computation at time t=0. We see that the function C(t) goes to a limit (depending on the value of P) for large t, and that the time required to reach the limit depends on r. The relaxation time T_r is defined as the inverse of r (see figure 2.2).

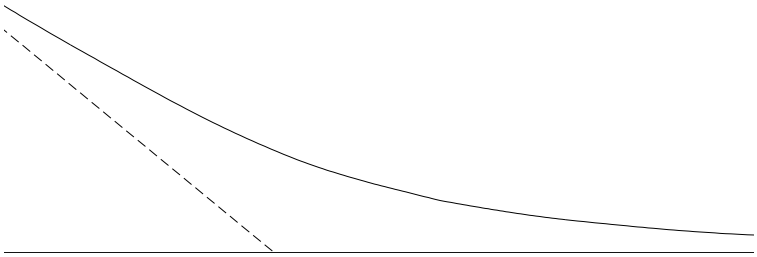


Figure 2.2. The analytic solution C(t)

Exercise 2.D:

Try to find an analytic solution for the equation (2.3) for the case that P varies sinusoidally, i.e.:

$$P=P_0 sin(\omega t)$$

where ω is the frequency of the sine function. Hint: for $C(t)$ try a function of similar form as for P, so also a sine function with unknown amplitude and phase; collect terms with $\sin \omega t$ and $\cos \omega t$ and suppose that both sets of terms are 0 because the expression must be 0 for all t.

Exercise 2.E:

Apply the result of the previous exercise for the case of the cooling-water problem; simulate the daily variation of energy demand by sinusoidal variation of the discharge $L(t)$. Use the model to determine how large the lake must be in order to keep the temperature within a specified range.

2.1.2. Box models with more state variables, stiffness

So far the examples treated involve only on unknown state variable $C(t)$. Often the state of a system has to be described with more than one state variable i.e. $Cj(t)$, where $j=1,...,k$. An example of such a system is the mass-and-spring system where the state is described by two variables i.e. the position of the mass (called $x=C_1(t)$) and its velocity ($v = C_2(t)$). The equation for the change of the velocity is obtained by considering the balance of momentum for the mass:

$$m\frac{dv}{dt} = -Kx - Fv$$

where m is the mass, K is the stiffness of the spring and F is the friction coefficient. The second equation which describes the change of position follows from the definition of v, i.e.

$$\frac{dx}{dt} = v$$

In general a system of differential equations for $C_j(t)$ is written as

$$\frac{dC_j}{dt} = F_j\left(C_1, C_2, \ldots, C_k, t\right) \quad j = 1, \ldots, k \tag{2.5}$$

There are many computer packages for the numerical solution of such systems of differential equations. Section 4.1 mentions a few of them.

A system of linear differential equations for $C_j(t)$ is written as

$$\frac{dC_j}{dt} = \sum_{m=1}^{k} A_{jm} C_m \quad j = 1, \ldots, k \tag{2.6}$$

Such a system of equations (both 2.5 and 2.6) is called a system of order k. The order is equal to the number of initial values needed to calculate a solution. Note that a single differential equation involving higher derivatives of C up to the k-th derivative can be brought in the form of (2.5) and thus also is a system of order k.

Exercise 2.F:

Determine the matrix coefficients A for the mass-and-spring system described above.

The general solution for a homogeneous system of linear differential equations with constant coefficients is:

$$C_j = \mathrm{Re}\left(D_j e^{rt}\right) \tag{2.7}$$

where D_j is some (complex) constant and r is a complex number which is the same for all j. By substitution of (2.7) into (2.6) we obtain the following system of linear equations in D_j:

$$rD_j - \sum_{m=1}^{k} A_{jm} D_m = 0 \qquad j = 1,...,k \tag{2.8}$$

This set of equations has non-trivial solutions if r is an eigenvalue of the system of equations. We will find k such eigenvalues each of which is a complex number corresponding to a certain "mode" of motion of the system. The real part of r (which should be negative or zero) represents the damping; the imaginary part represents the oscillation:

$$\mathrm{Re}\left(e^{rt}\right) = e^{\mathrm{Re}(r)t} \cos\left(\mathrm{Im}\left(r\right)t\right)$$

The damping time of the mode is $1/Re(r)$ and the oscillation period is $2\pi/Im(r)$.

Exercise 2.G:

Find the eigenvalues of the mass-and-spring system by making the determinant of the following matrix equal to zero:

A_{11}-r A_{12}

A_{21} A_{22}-r

and conclude how the damping time and the oscillation period depend on the coefficients m, K and F.

In higher order systems one often encounters the notion of stiffness. This means that there are widely different eigenvalues in the system. In other words there are "modes" with widely different reaction times. An example of a stiff system is a double mass-and-spring system in which one spring is much stiffer than the other (see figure 2.3).

Let us assume that spring b is much stiffer than spring a while both masses are roughly the same. Then there is a mode in which both masses oscillate with the same phase (i.e. move in the same direction) and which has a low frequency, and there is a mode in which the masses oscillate with phase difference of 180^o and which has a high frequency.

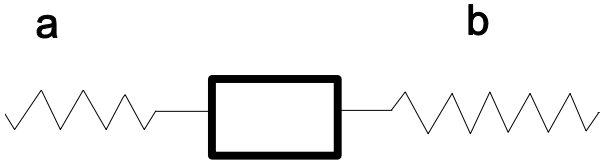


Figure 2.3. A double mass-and-spring system

Stiff systems are difficult to compute numerically. The interesting motion are usually the low frequency modes, so these must be computed accurately. At the same time the numerical method must be able to cope with the high frequency modes, but these do not have to be computed accurately. Stiff systems occur frequently in ecological modelling because in ecological systems there are often interactions with widely differing reaction times, but also in the numerical computation of partial differential equations.

2.2. The 1-D transport equation

The transport of a pollutant (including salt, heat etc.) in a one-dimensional channel is described by a partial differential equation involving time t and axial coordinate x as independent variables. The derivation of this equation is a fairly straightforward extension of the derivation of the box model. The main difference is now that the control volume is a portion with length Δx of the channel. We define C to be the concentration, i.e. the amount of pollutant per unit volume; consequently the unit of concentration often is kg/m^3. We define A_b to be the total wetted cross-section of the channel; so the volume of the control volume is $A_b \Delta x$. Finally S is the transport, i.e. the amount of pollutant passing a cross-section per unit time, and P is the production of pollutant per unit time and per unit volume. The balance equation reads:

$$\frac{\partial A_b C}{\partial t} + \frac{\partial S}{\partial x} = A_b P \qquad (2.9)$$

The first term multiplied by Δx (i.e. $\partial(A_b \Delta x C)/\partial t$) is the rate of change of the mass of the pollutant in the control volume; the second comes from the difference in transport at both ends of the control volume: $S(x+\Delta x)-S(x)$; the right hand side (also multiplied by Δx) gives the production of pollutant in the volume. If we consider for instance the concentration of algae, part of the production term (a negative contribution) comes from the fact that algae are eaten by zooplankton; if we consider heat, the production term is used to model the loss of heat from the water to the atmosphere through the surface.

Exercise 2.H:

Write the balance equation for heat transport in a channel.

In the balance equation (2.9) A_b is assumed to be known as a function of x and t, and P is a given function of the concentration and perhaps on other quantities. The unknowns in the equation are C and S, both functions of x and t. Since there are two unknowns, there must be a second equation involving C and S. This is the transport expressed in C. The transport consists of two contributions, viz. the advective part and the diffusive part (first and second terms of the right hand side of equation 2.10 resp.):

$$S = QC - A_s K \frac{\partial C}{\partial x} \tag{2.10}$$

where Q is the discharge, i.e. the amount of water passing a cross-section per unit time, A_s is the flow cross-section, i.e. the part of the total cross-section in which the flow takes place, and K is the diffusion coefficient. Q, A_s and K are assumed to be known as function x and t. In fact the values of the parameters Q, A_b and A_s are the result of a flow model; this model is considered in the next section. K depends on the flow conditions, such as depth and velocity, and is therefore also determined from the results of the flow model.

The effect of the advective term is to move the mass in x-direction (without changing the total mass), the effect of the diffusive term is to spread the pollutant over a longer distance (without changing the total mass), and the effect of the production term is to increase or decrease the total mass.

The essential properties of the transport model can be found in simplified versions of the equation. Many of the properties of the numerical approximations of the transport equation are also derived for simplified versions.

In the simplified versions of the transport equation it is assumed that Q, A_s, A_b and K are constant; furthermore P is assumed to be 0. After eliminating S the linear convection-diffusion-equation results:

$$\frac{\partial C}{\partial t} + v \frac{\partial C}{\partial x} - K \frac{\partial^2 C}{\partial x^2} = 0 \tag{2.11}$$

The transport velocity or propagation velocity of the pollutant is $v = Q / A_b$

Note: numerical models (like DUFLOW, see chapter 5) are always based on the general equations with variable values of Q, A_s, A_b and K. The above simplification is only done to derive analytical solutions.

Exercise 2.I:

Show that the following expression is an analytical solution of the convection-diffusion equation (2.11):

$$C(x,t) = \frac{M_0}{A_b \sqrt{(4\pi K t)}} e^{-(x-vt)^2/4Kt} \tag{2.12}$$

Make graphs of C as a function of x for two different values of t. You will see that a combination of translation and spreading of the pollutant occurs. The translation speed is v, the "wavelength" is proportional to $\sqrt{Kt}$ *, for instance* $4\sqrt{Kt}$ *.*

Exercise 2.J:

Show that the following expression is an analytical solution of the convection-diffusion equation (2.11) in the stationary case:

$$C = C_0 + C_1 e^{+xv/K} \tag{2.13}$$

Make a graph of C as a function of x with given boundary conditions at x=0: C(0)=1 and at x=X=10.K/v: C(X)=2. You will see that the influence of the downstream boundary condition is felt only in a small region; the length of this region is of the order of K/v.

Simple-wave equation

In practice sometimes the diffusion is negligible, for instance when the concentration profile is very smooth already. In that case the diffusion term is neglected and the convection-diffusion equation reduces to the so-called simple wave equation:

$$\frac{\partial C}{\partial t} + v \frac{\partial C}{\partial x} = 0 \tag{2.14}$$

Exercise 2.K:

Show that the solution of the simple wave equation (2.14) is a simple translation of the initial condition with transport velocity v; in other words: if the value of C at a time t=0 is: C(x,0)=F(x), the value of C for arbitrary x and t is: $C(x,t)=F(x-vt)$.

The result of exercise 2.K shows that the simple wave equation is a typical propagation equation; v plays the role of propagation velocity. The simple wave equation is used in chapter 4 to demonstrate the most important properties of various numerical schemes used for propagation phenomena.

Just like ordinary differential equations partial differential equations need one or more initial conditions, and because of the presence of partial derivatives with respect to x, also boundary conditions. We can say that initial conditions are needed because we choose to compute only a finite time interval, and the initial condition represents the influence of history; similarly we need boundary conditions because we carry out computations only in a finite region, and the boundary conditions represent the influence of the outside world. Note that

> **the requirements regarding initial and boundary conditions are a consequence of the nature of the (partial) differential equation.**

A consequence of this is: how many initial and/or boundary conditions are needed, has nothing to do with numerical approximations. Also note that

> **the mathematical nature of a (partial) differential equation is determined by its highest order derivatives.**

In the case of the simple wave equation, or the more general transport equation without diffusion term we can deduce how many initial and boundary conditions are needed using the concept of characteristic curves, or simply "characteristics". The transport equation without diffusion reads:

$$\frac{\partial A_b C}{\partial t} + \frac{\partial QC}{\partial x} = A_b P \tag{2.16}$$

or:

$$A_b \left(\frac{\partial C}{\partial t} + v \frac{\partial C}{\partial x} \right) + C \left(\frac{\partial A_b}{\partial t} + \frac{\partial Q}{\partial x} \right) = A_b P$$

from which the second term of the left hand side disappears due to the conservation of mass of the water (see also the next section); so:

$$\frac{\partial C}{\partial t} + v \frac{\partial C}{\partial x} = P$$

This partial differential equation reduces to an ordinary differential equation for C along characteristic curves. The characteristic equations read:

$$\frac{dC}{dt} = P \tag{2.17}$$

which is valid along curves in the x-t-plane which obey:

$$\frac{dx}{dt} = \frac{Q}{A_b} = v \tag{2.18}$$

For an ordinary differential equation such as (2.17) we need one initial condition; this initial condition is the value of C at the point where the characteristic curve enters the computational domain. The consequence is that also the partial differential equation needs a boundary condition at each point where a characteristic enters the domain; the following rule also holds in cases where there are more than one family of characteristics (see for instance the shallow water equation in the next section):

> **the number of initial or boundary conditions is equal to the number of characteristics entering the computational domain.**

In the case of the transport equation with diffusion term this rule does not help us very much since there are no characteristics. It can be derived that there is a boundary condition needed at each end of the computational domain. Since in the equation there appears only one first order derivative with respect to t one initial condition is sufficient.

2.3. The shallow water equation

The shallow water equation describes unsteady motion in channels; it is assumed that the depth is very small compared with the wavelength. The shallow water equation consists of two equations for two unknowns (state variables), viz. the water level h and the discharge Q. In some of the computer programs for the shallow water equation the average velocity U is chosen as state variable together with h. We will use Q and h. The water level h is taken with respect to a horizontal datum (reference level). The mass conservation equation or continuity equation is one of the two equations. We use the version where the density of the water ρ is assumed to be constant. The continuity equation is very similar to equation (2.9):

$$\frac{\partial A_b}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{2.19}$$

The total wetted cross-section A_b is a known function of x and h (in every place along the channel it is known how A_b depends on h). The partial derivative of A_b with respect to h is also known as the width of storage b. It appears in the continuity equation if we transform the equation such that h is the unknown instead of A_b:

$$b\frac{\partial h}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{2.20}$$

The second equation is the equation of motion; it is derived applying the balance principle to the amount of momentum in the control volume. Furthermore it is assumed that (a) the pressure distribution is hydrostatic and (b) the flow changes slowly in time so that the bottom friction can be modeled as if the flow were steady. It is assumed that the flow velocity is equal to U in the flow cross-section A_s and 0 in the remainder of the cross-section ($A_s \leq A_b$). In other words the discharge is $Q=A_sU$.

$$\frac{\partial Q}{\partial t} + \frac{\partial QU}{\partial x} + g A_s \frac{\partial h}{\partial x} + C_{fr} \frac{|Q|Q}{A_s R} + \frac{F_W}{\rho} = 0 \tag{2.21}$$

C_{fr} is the friction coefficient and F_W is the wind force (more precisely: the component of the wind stress vector in the direction of the channel axis).

As in the case of the transport equation there are simplified versions of the shallow water equation for which we can find analytical solutions. The equations (2.20) and (2.21) are linearized; it is assumed that there is a small variation of h and Q on top of a uniform flow situation. (2.20) is linear already because b is constant if we consider only small variations of h. The small variations of Q and h are called q and h, resp., i.e. $Q=Q_0+q'$ and $h=h_0+h'$. From (2.20) and (2.21) we get:

$$b\frac{\partial h'}{\partial t} + \frac{\partial q'}{\partial x} = 0 \tag{2.22}$$

$$\frac{\partial q'}{\partial t} + 2U\frac{\partial q'}{\partial x} - U^2 b_s \frac{\partial h'}{\partial x} + gA_s \frac{\partial h'}{\partial x} - gI_b b_s h' + 2C_{fr}\frac{|U|}{d}q' - 2C_{fr}\frac{|U|Q_0}{d^2}h' = 0 \tag{2.23}$$

where b, $A_s=b_sd$, $U=Q_0/A_s$ etc. are assumed to be constant.

Exercise 2.M:

Find the propagation velocity of a long wave from eq. (2.22) and (2.23), assuming that the initial shape of the wave is sinusoidal, with given wavelength. hint: The solution is of the form

$$h = h_c e^{i\omega t - ikx}$$

$$q = q_c e^{i\omega t - ikx}$$

where k is given because the initial condition is given, and where ω has to be determined; the imaginary part of ω gives the damping and Re(ω)/k is the propagation velocity of the wave. In general you should find two waves, one in the direction against the permanent flow, one in the same direction as the flow. To simplify the matter carry out the analysis for higher frequencies, i.e. neglect all terms not involving derivatives.

Exercise 2.N:

Find the propagation velocity of a flood wave from eq. (2.22) and (2.23), in the same way as in exercise 2.M, but this time for very long frequencies, i.e. in (2.23) all terms are neglected which contain derivatives. In this case you should find one wave in the same direction as the flow.

Exercise 2.O:

In a harbour having the shape of a prismatic channel with rectangular cross-section, oscillations occur. For such oscillations a strongly simplified version of the shallow water equation can be used in which the advective acceleration term and the friction term are neglected:

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} = 0$$
$$\frac{\partial h}{\partial t} + d\frac{\partial u}{\partial x} = 0 \qquad (2.23')$$

in which g and d can be assumed to be constant.
At the seaward boundary a sinusoidal variation of the water level with small amplitude is prescribed. If the period of this variation is such that the wavelength is 4 times the length of the harbour, large fluctuations will occur in the harbour due to resonance . At the end of the harbour Q=0. Show that a standing wave develops, and that for certain wavelengths resonance occurs. A standing wave is of the form:

$$h = h_0 \cos(\omega t - \psi_1)\cos(ikx - \psi_2)$$
$$q = q_0 \cos(\omega t - \psi_3)\cos(ikx - \psi_4)$$

where h_o and q_o are real constants. In this analysis neglect the resistance term and the advective acceleration term.

Exercise 2.P:

Find a solution of the equations (2.22) and (2.23) for the stationary case. Assume that the time derivatives are 0, and substitute an exponential function of x. The result should be that h goes to 0 when travelling upstream, and that the relaxation length is of the order of d/I_b where I_b is the bottom slope. This result is important because it shows that the region of influence of a downstream boundary condition in a river is limited to 2 or 3 times d/I_b. Note that this conclusion is only true in the case of subcritical flow, i.e. $U < \sqrt{gA/B}$.

2.4. Floodwave in a river

If one considers the flood wave in a river there is a simpler version of the equation of motion. In very gradual floods, i.e. the length of the flood wave is large compared with d/I_b, the derivatives in this equation are negligible, and the only terms that remain in the equation of motion represent uniform flow:

$$g A_s I_b + C_{fr}\frac{|Q|Q}{A_s R} = 0$$

This equation represents a relationship between Q and h.

Often Q is expressed in terms of h:

$$Q = \left(\frac{g A_s^2 R I_b}{C_{fr}}\right)^{1/2} \qquad (2.24)$$

It is noted that, although h does not appear explicitly in the right hand side of (2.24), it is a function of h since A_s, R and C_{fr} are given as functions of h. When (2.24) is substituted in the continuity equation (2.19) a first order partial differential equation results which read:

$$b\frac{\partial h}{\partial t}+\frac{dQ}{dh}\frac{\partial h}{\partial x}=0 \tag{2.25}$$

The characteristic velocity or propagation velocity can be seen to be:

$$\frac{dx}{dt}=c_f=\frac{1}{b}\frac{dQ}{dh} \tag{2.26}$$

This propagation velocity is of the order of the particle velocity of the water; it has the same direction as the particle velocity. In a channel with simple rectangular cross-section $c_f=1.5\ U$.

It is seen from (2.25) that h is constant along the characteristics defined by (2.26). In contrast to the simple wave equation (2.14) the characteristics are not parallel, so the flood wave is deformed during propagation.

Exercise 2.Q:

Using a simple graphical solution method based on the characteristics, determine whether in a flood wave the front of the wave becomes steeper in time or more gradual.

We conclude from the nature of the characteristics that the flood wave equation needs one initial condition and one boundary condition at the upstream side of the river.

Exercise 2.R:

Find the propagation velocity of a long wave from eq. (2.24), assuming that the initial shape of the wave is sinusoidal, with given wavelength. hint: The analysis is simpler when using the complex representation of sin and cos. This time you should find only one wave in the same direction as the flow.

2.5. Characteristics and the relation with initial and boundary conditions

The shallow-water equation can also be written in characteristic form. This is not immediately clear from the equations (2.20) and (2.21). The derivation of the characteristic equations can be found in many textbooks on open channel hydraulics (Dronkers, 1964; Mahmood and Yevyevich, 1975; Ven te Chow, 1983). There are two ways to carry out the derivation; one is based on the notion that discontinuities in the derivatives of Q and h travel along characteristics, the other attempts to bring the equations into a form such as (2.14) by taking a proper linear combination of (2.20) and (2.21).

In the latter derivation (2.21) is rewritten such that we have single derivatives of Q and h:

$$\frac{\partial Q}{\partial t}+2U\frac{\partial Q}{\partial x}-U^2 b_s\frac{\partial h}{\partial x}+gA_s\frac{\partial h}{\partial x}+W=0 \tag{2.27}$$

here W represents the combined effect of friction and wind. We then add (2.27) and (2.20) multiplied by an as yet unknown factor m:

$$mb\frac{\partial h}{\partial t}+(gA_s-U^2b_s)\frac{\partial h}{\partial x}+\frac{\partial Q}{\partial t}+(2U+m)\frac{\partial Q}{\partial x}+W=0 \tag{2.28}$$

Now we choose m such that Q and h are differentiated in the same direction in the x-t-plane. The propagation direction v will be:

$$v=\frac{gA_s-U^2b_s}{mb}=\frac{2U+m}{1}$$

This is a quadratic equation in m; there are two solutions

$$m_{1,2}=-U\ \pm\ \sqrt{\frac{gA_s}{b}+U^2\left(1-\frac{b_s}{b}\right)}$$

The result is that there are <u>two</u> families of characteristics with the propagation velocities:

$$\frac{dx}{dt} = v_{1,2} = U \overset{+}{-} \sqrt{\frac{g A_s}{b} + U^2(1 - \frac{b_s}{b})} \tag{2.29}$$

From (2.28) it is seen that along the characteristic curves the following relations hold:

$$\frac{dQ}{dt} + (v - 2U)b\frac{dh}{dt} + W = 0 \tag{2.30}$$

In the method of characteristics along a characteristic the development of a combination of Q and h (such a combination is known as a *Riemann invariant*) is given by (2.30); for the strongly simplified shallow water equation (2.23') the Riemann invariants are:

$$Q \overset{+}{-} b\sqrt{gd}\, h$$

The method of characteristics has been used in computer programs for the shallow water equation but it has the disadvantage that the intersection points of characteristics are at arbitrary points in the x-t-plane. The method characteristics is at its best in cases of sudden transitions.

In this book we consider only the finite difference method with fixed points in space (see chapter 4).

The propagation velocity associated with the characteristic curves is the same as that of the harmonic wave with high frequency.

In sub-critical flow $(U^2 < gA_s/b)$ the two propagation velocities have different signs. Based on the rule concerning the relation between boundary conditions needed and the number of incoming characteristics we can conclude that in sub-critical flow always two initial conditions are needed, and one boundary condition at each end of the computational region, because at an end there is always one incoming characteristic.

Since in deriving the characteristics we brought the shallow-water equation into the form of the simple wave equation (2.14) we can transfer properties of (2.14) to the shallow-water equation; we must take into account that the propagation velocity (2.26) now takes the place of the transport velocity v in equation (2.14). In particular conclusions regarding numerical accuracy and stability derived for the simple wave equation will be used also for the shallow-water equation.

The difference between the propagation velocity of the flood wave equation (2.26) and that of the shallow-water equation may seem puzzling. In exercise 2.R a propagation velocity for a sinusoidal wave of finite wavelength was derived; one can check that this propagation velocity is dependent on the wavelength, and that the limit for very short wavelength is (2.29), and the limit for very large wavelength is (2.26), the propagation velocity of the flood wave.

Equations like the shallow water equation have characteristics and can therefore be classified as propagation phenomena.

Partial differential equations are generally divided into three categories:

(I)	parabolic equations	two equal values for m
(II)	hyperbolic equations	one or two different real values for m
(III)	elliptic equations	two complex values for m

For simple linear equations the classification is simple and can be found in many textbooks on computational techniques for the approximation of PDE's, see e.g. Abbott and Basco (1989), Fletcher (1988), Hirsch (1991), but we consider this classification beyond the scope of our lecture notes. We will restrict ourselves to a few examples of each category:

The simplest example of an hyperbolic equation is the convection equation, given by:

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} = 0$$

As we have seen in section 2.2 this equation has one family of characteristics. This implies that disturbances are travelling in the x,t space with a finite speed along characteristics given by $x\text{-}Ut=x_0$. Here x_0 is a constant that is equal to the position of a disturbance at t=0. This type of equations is characterized by the fact that gradients do not

change as time proceeds. This means the shape of some initial condition will not change as time proceeds but will only change its position. Think for example of a travelling solitary wave. Convective transport of matter is described by this type of equations as we have pointed out in chapter 2. This equation is referred to as the simple convection equation.

An example of the parabolic type of equations is the convection-diffusion equation (2.11). The simplest example of a parabolic equation is the diffusion equation, given by:

$$\frac{\partial c}{\partial t} - K\frac{\partial^2 c}{\partial x^2} = 0 \qquad (2.30')$$

here K denotes a diffusion coefficient.

Parabolic equations are characterized by dissipative or smoothing behaviour of their solutions. Generally spoken the parabolic aspect of this type of equations describes the decrease of gradients of the solution as the time proceeds. Think for example of heat conduction. From a hot spot in a cold environment heat will be conducted to other parts in the domain R under consideration. In other words the sharp temperature gradient near the hot spot decreases until the gradients are zero everywhere in the domain under consideration. Of course external effects such as heating on one side and cooling on another might prevent the system from having gradients to be zero throughout the domain.

The most common example of the elliptic type is the Laplace equation:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0\,. \qquad (2.30'')$$

Here ϕ denotes some potential function. Equations of this type describe equilibrium situations, an expression also used by Abbott and Basco (1989), such as potential flow problems and groundwater flow with isotropic and constant permeability. This means that the situation that is described is not varying in time but only in space.

Equations as given above are to be completed with boundary conditions. The following boundary conditions are used, depending on the equations:

(A) *Dirichlet* condition, e.g. $c=f$ on ∂R, ∂R is the boundary of R.

(B) *Von Neumann* condition, e.g. $\partial c/\partial n=f$, n is in the direction normal to ∂R

(C) mixed or *Robin* condition, e.g. $\partial c/\partial n+\mu C=f$ on ∂R.

exercise 2.R:

Try to find the values of m for the Laplace equation; to apply the procedure used for the shallow water equation first write it as a system of two first-order equations for the quantities:

$$p = \partial\phi/\partial x$$

$$q = \partial\phi/\partial y$$

The two equations are:

$$\frac{\partial p}{\partial x} + \frac{\partial q}{\partial y} = 0$$

$$\frac{\partial p}{\partial y} - \frac{\partial q}{\partial x} = 0$$

You should obtain: $m_{1,2} = \pm i$.

Elliptic equations are entirely different from hyperbolic equations; the state in a point (x,y) is influenced by all surrounding points. Elliptic problems are therefore typically boundary value problems, usually with two (or three) space coordinates as independent variables. Because we will only deal with time dependent problems the numerical treatment of this type of equation will not be dealt with.

Hyperbolic (and parabolic) equations are typically initial value problems with x and t as independent variables.

The nature of the partial differential equation is also reflected in the numerical solution method. In initial value problems the numerical solution also progresses step by step through the time domain. In boundary value problems the (simplest) solution method is iterative: the value of the unknown in each point is updated a number of times (taking account of the values of surrounding points) until sufficient accuracy is reached.

2.6. Absorbing boundary conditions

Users of simulation programs usually try to locate the model boundaries at clear points: a weir or pumping station, the end of a channel, the point where a river flows into the sea etc.

This is not always efficient. If a computational domain has to end in the middle of a channel one applies a boundary at a place where in reality waves approaching from inside the computational domain will propagate through the boundary undisturbed. In such a case we often make use of a wave absorbing boundary condition. An absorbing boundary condition can be applied both in the shallow-water equation and in a transport computation.

There are several ways to derive absorbing boundary conditions; some of them are based on characteristics. In the derivation presented here we choose the boundary condition such that we impose that there is undisturbed wave propagation at the boundary. We know that for wave propagation without deformation the simple wave equation (2.14) holds, i.e.

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} = 0 \tag{2.31}$$

here v is the propagation velocity of the wave; its value depends on the physics. The sign of v must be such that the wave is leaving the computational domain; an absorbing boundary condition can never be applied to waves entering the computational domain. If the value of v is inaccurate the wave approaching the boundary will not be absorbed perfectly. In many physical systems (e.g. the shallow water equation) v is dependent on the wavelength, so then one has to assume a representative wavelength and choose v accordingly. Waves with a different wavelength will then be reflected partially.

In the case of the transport equation the propagation velocity is v, the transport velocity. Equation (2.31) can be generalized to

$$\frac{\partial A_b C}{\partial t} + \frac{\partial QC}{\partial x} = A_b P \tag{2.33}$$

i.e. the transport equation without diffusion term. In other words: we obtain the absorbing boundary condition by assuming that the influence of diffusion is negligible at the boundary. Because we can use the absorbing boundary condition only for outgoing waves we can apply this boundary condition only at a place where outflow occurs. At an inflow boundary always C or S must be prescribed.

In the case of the shallow-water equation we have seen (exercise 2.O) that a boundary condition where either Q or h is given, always reflects waves. Therefore an absorbing boundary condition must be some relation between Q and h. An absorbing boundary condition can for instance be obtained by applying equation (2.31) to Q. We get

$$\frac{\partial Q}{\partial t} + v\frac{\partial Q}{\partial x} = 0 \tag{2.34}$$

Using the continuity equation this can be rewritten into

$$\frac{\partial Q}{\partial t} - bv\frac{\partial h}{\partial t} = 0 \tag{2.35}$$

We see (by integrating eq. 2.35) that at an absorbing boundary Q and h are related; the values of Q and h themselves are still unknown.

The value of v that we use in the boundary condition depends on the wavelength. For flood waves (friction dominating over acceleration) v should be chosen equal to c_f (see equation 2.26). In this particular case equation (2.35) can be integrated analytically; the result turns out to be equation (2.24). Therefore we conclude that for a flood

wave on a river the equation for uniform flow is the absorbing boundary condition. For the same reason as in the case of the transport equation we can use this boundary condition only at an outflow boundary.

For shorter waves where acceleration is not negligible another value of v must be used. For waves where acceleration dominates over friction the value of v is equal to the characteristic propagation velocity of the shallow-water equation, eq. (2.29).

Since in the shallow-water equation there are two waves, one incoming and one leaving the computational domain, there is a need of handling the combination in the boundary condition. Equation (2.35) can be used if there is no incoming wave; if there is one, (2.35) is valid only for the outgoing wave.

Let v_i be the propagation velocity of the incoming wave, and v_o the velocity of the outgoing wave. If there is an incoming wave, it characterized by h_i and Q_i; the two are related by:

$$\frac{\partial Q_i}{\partial t} - bv_i \frac{\partial h_i}{\partial t} = 0$$

For the outgoing wave, characterized by $h_o=h-h_i$ and $Q_o=Q-Q_i$, we have equation (2.35), i.e.:

$$\frac{\partial (Q - Q_i)}{\partial t} - bv_0 \frac{\partial (h - h_i)}{\partial t} = 0 \qquad (2.36)$$

This equation can be used as the absorbing boundary condition for the shallow water equation.

2.7. Equations for an aquifer

In most aquifers the flow is predominantly horizontal. In that case the equation for groundwater flow is derived in much the same way as the shallow water equation. In contrast to open channel aquifers extend in two dimensions; however, if we assume that the conditions in the aquifer are independent of one horizontal coordinate y, we can derive a partial differential equation in x closely resembling the shallow water equation.

The mass conservation equation is the same apart from the fact that in rising water table the water fills only part of the volume, namely the part which is not taken by the grains; we introduce the coefficient of porosity ε. In the equation of motion we assume Darcy's law whereby the flow velocity is linear with the slope of the piezometric head.

Exercise 2.T:

Derive the equations for unsteady groundwater flow in a horizontal aquifer, i.e. the equation for conservation of mass:

$$\frac{\partial q}{\partial x} + \varepsilon^{-1} \frac{\partial h}{\partial t} = N$$

and the equation of motion:

$$q = k(h - Z) \frac{\partial h}{\partial x}$$

Here q is the discharge per unit width, ε *is the porosity, i.e. the part of the volume not taken by the grains; k is the permeability, i.e. the coefficient relating the flow velocity and the gradient of the piezometric head. Z is the level of the impermeable base. N is the precipitation, i.e. the amount of water falling on the ground per unit time and per unit surface. It is assumed that the precipitation reaches the aquifer immediately.*

Exercise 2.U:

Determine how many initial and boundary conditions are needed to solve the equations derived in the previous exercise. What is the type of this system of equations?

Chapter 3 Initial value problems of ordinary differential equations

3.1. Introduction

Initial value problems of systems of ordinary differential equations are often generated by application of the Method of Lines for the approximation of Partial Differential Equations. As an example we consider a simple diffusion equation given by:

$$\frac{\partial c}{\partial t} - K\frac{\partial^2 c}{\partial x^2} = 0$$

After replacing the spatial derivatives by a finite difference approximation given by:

$$\frac{\partial^2 c}{\partial x^2} \approx \frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2}$$

we obtain:

$$\frac{dc_i}{dt} - K\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2} = 0$$

where $i=1,...,I-1$. We assume boundary conditions prescribed at x_0 and x_I.

These type of sytems of ODE's are the main motivation of this chapter however in this chapter we assume the system of ODE's as given and we study only the consequences of the numerical integration in time. It is to be noted however that in practical applications in computational hydraulics also ODE's are applied. Examples are:

Box models, as given in the introduction, or chapter 2, eq.(2.2). The simplest example is given by:

$$V\frac{dC}{dt} = QC_I - \left(C\frac{V}{T_r} + QC\right)$$

Predator prey relations in reservoirs or lakes, for ecological simulations, given by for example:

$$\frac{d\text{PRED}}{dt} = \text{GROWTH1} * \text{PREY} * \text{PRED} - \text{DEATH1} * \text{PRED}$$

$$\frac{d\text{PREY}}{dt} = -\text{CONSUM} * \text{PREY} * \text{PRED} + \text{GROWTH2} * \text{PREY}$$

where:

	PRED	Predator concentration per unit volume
	PREY	Prey concentration per unit volume
	DEATH1	Death rate of predators
	GROWTH1	Growth rate of predators
	CONSUM	Consumption(death) rate of preys
	GROWTH	Constant growth rate of preys

Predator-prey type of relations are often applied for ecological modelling, to model food chains see e.g. Thomann (1987).

Water surface curves, such as the back water curve, for river applications, for example given by:

$$\frac{dH}{dx} = \frac{I_b - Q^2/\left(C^2AR\right)}{1 - b_s Q^2/\left(gA^3\right)}$$

Where:		
	A	Wet cross-section
	b_s	Surface width
	C	Chezy coefficient
	I_b	Bottom slope
	H	Waterdepth
	R	Hydraulic radius
	Q	Discharge

A mass spring system, for example given by:

$$\frac{dv}{dt} + \frac{A}{m} v + \frac{K}{m} x = 0$$

$$\frac{dx}{dt} - v = 0$$

where m is the mass, K denotes the stiffness and A is the friction coefficient.

Numerical integration in time implies the almost infinite repetition of recurrent relations or difference equations, therefore we start this chapter with these equations as such. Then we treat the numerical integration of a simple ODE, including aspects such as stability, consitency and convergence. Finally we will treat systems of ODE's including the numerical problems due to different timescales such as the problem of stiffness.

3.2. Difference Equations

Computers can only perform simple mathematical operation such as addition, subtraction, multiplication and division[1] on digital numbers. Modern computers can perform these simple operations at a very high speed due to its floating point processor[2]. Therefore a computer can repeat a simple formula almost an infinite number of times. Every numerical simulation with digital computers, no matter the kind of physics that is simulated and no matter what numerical method is used, is based upon a large number of repetitions of such formula's. Sometimes these formula's are called numerical recipes, an expression that we took from Press et al.(1988). They are derived from mathematical expressions that we call "difference equations" or "recurrent relations". For the analysis of computational results it is often not sufficient to understand the physics with which the computer simulation is supposed to be consistent. In fact the almost infinite repetition of algebraic formula's is governed by algebraic laws. Of these laws some basic understanding might help when results sometimes seem to have no physical justification or can make one to be always critical towards computational results no matter how plausible results might look like.

The simplest numerical recipe that one can imagine is given by:

$$c_n = a \cdot c_{n-1} \tag{3.1}$$

It means that each new value c_n is computed from a previous value c_{n-1}. Obviously to start this "numerical recipe" a starting value for some index n has to be given, e.g. for n=0:

$$c_0 = 1 \tag{3.2}$$

---

[1] And even these simple operations are performed by computers only with finite accuracy. This is primarily due to the finite number of digits that computers have available to represent real numbers. This causes round off errors. Also the way in which computers execute multiplications contributes to round off errors. A discussion on this topic is not given in these lecture notes, but especially for matrix problems they may be important, see e.g. Golub and Van Loan (1983).

[2] Some computers have a "peak perfomance" of the order of "giga flops". This means 1,000,000,000 floating point operations, such as multiplications per second. This speed is increasing almost every day, with new computers entering the market.

Such a set of formulas has to be instructed to a computer via a computer language. Often the computer language FORTRAN 77 is used for this job. FORTRAN is an acronym for "formula translation". In FORTRAN 77 (3.1) and (3.2) could look like:

```
      PROGRAM FORMULA
      DIMENSION C(0:100000)
      READ *,C(0),A,NLAST
      DO 10 N=1,NLAST
        C(N)=A*C(N-1)
10    CONTINUE
      END
```

In a slightly different notation this "recipe" can be denoted as the following set of "recurrent relations" or "difference equations":

$$\begin{aligned} c_0 &= 1 \\ c_{n+1} - ac_n &= 0, \quad n = 0,...,N \end{aligned} \tag{3.3}$$

The general solution of (3.3) is given by:

$$c_n = a^n \tag{3.4}$$

In fig.(3.1) we examine (3.4) for several values of a:

case(i) a=-1.1
case(ii) a=-1.0
case(iii) a=-0.8
case(iv) a=0.8
case(v) a=1.0
case(vi) a=1.1

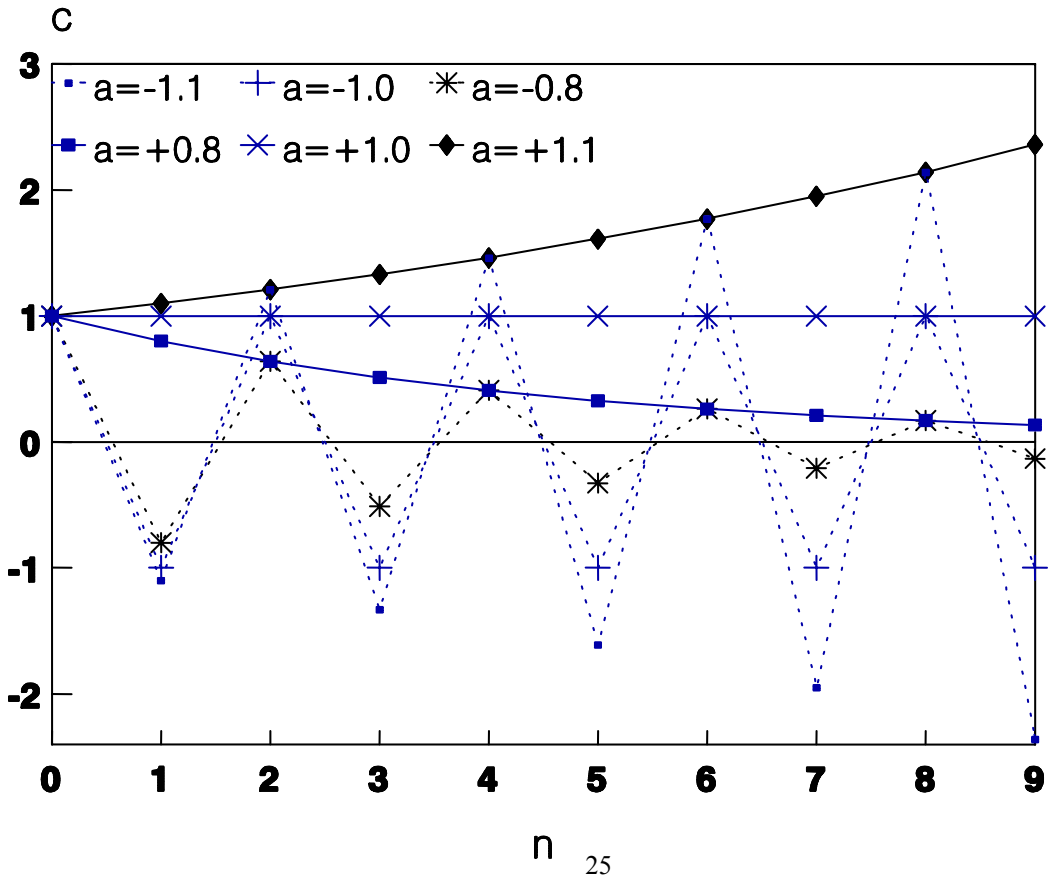


Figure 3.1

What we see is the following:

case(i) : c is oscillating and will become infinite if (3.1) is repeated an infinite number of times. In this case we say that (3.1) is unstable.

case(ii) : c is oscillating, but the amplitude remains constant, i.e. the solution is stable.

case(iii) : c decays and will approach 0 in the limit, i.e. in the case that (3.1) is repeated an infinite number of times. In this case (3.1) is stable and dissipative. But the solution is oscillating, and has negative solutions.

case(iv) : c decays and will approach 0 in the limit, i.e. in the case that (3.1) is repeated an infinite number of times. In this case (3.1) is stable and dissipative. The solution is also non-oscillating, in this case we call the numerical recipe monotonic.

case(v) : c remains constant also if (3.1) is repeated an infinite number of times. In this case (3.1) is still absolutely stable.

case(vi) : c is growing with a constant rate, and it will grow to infinity, if (3.1) is repeated an infinite number of times. If this infinite growth occurs then (3.1) is considered to be absolutely unstable.

Note that here stability implies the boundedness of c_n for arbitrary large values of n. Later on we will give more precise definitions.

Recipes of the kind as given by (3.1) will be used as formula's to simulate physical processes such as flow or the behaviour of dissolved or suspended matters. In general we will simulate stable physical processes, i.e. processes which are not sensitive to small disturbances. Some "resemblance" between the physical reality and the phenomena to be simulated seems to be a necessary condition for a numerical recipe in order to be useful for simulation. Obviously for the cases (i) and (vi) useful results are not to be expected. Also if numerical solutions are oscillating and have negative results, then sometimes useless result can be produced. Assume for example that c represents a concentration value or the temperature simulation given by (2.2), in such a case negative results are not realistic.

(3.1) is certainly not the only possible recipe that can be instructed to the computer via a computer language. An other example is given by the following formula:

$$c_n = a \cdot c_{n-1} + b \cdot c_{n-2} \tag{3.7}$$

In this case the result of c_n depends on two previous values of c. Now in order to start (3.5) we need two starting values, for example:

$$\begin{aligned} c_0 &= C_0 \\ c_1 &= C_1 \end{aligned} \tag{3.8}$$

We now denote (3.7) and (3.8) as:

$$\begin{aligned} c_0 &= C_0 \\ c_1 &= C_1 \\ c_{n+2} - ac_{n+1} - bc_n &= 0, \quad n = 2,3,...,N \end{aligned} \tag{3.9}$$

A solution of (3.9) differs from (3.4). Suppose that we are looking for a solution of the following form:

$$c_n = r^n, \quad n = 2,3,...,N$$

Substitution of this assumption into (3.9) gives:

$$r^{n+2} - ar^{n+1} - br^n = 0$$

Division by r^n gives:

$$r^2 - ar - b = 0 \tag{3.10}$$

Eq. (3.10) is called the "characteristic equation" of (3.9) (Note that this is entirely different from the characteristic curves as defined in chapter 2). It has two solutions given by:

$$r_1 = \frac{a + \sqrt{a^2 + 4b}}{2}, \; r_2 = \frac{a - \sqrt{a^2 + 4b}}{2} \tag{3.11}$$

We suppose that r_1 is different from r_2. Since (3.9) is linear it follows that the solution of (3.9) can be denoted as the superposition of two elementary solutions:

$$c_n = d_1 r_1^n + d_2 r_2^n \tag{3.12}$$

d_1 and d_2 denote constants that are determined by the starting values of (3.9) as follows:

$$\begin{aligned} d_1 + d_2 &= C_0 \\ d_1 r_1 + d_2 r_2 &= C_1 \end{aligned} \tag{3.13}$$

If $r_1=r_2$ then nr_1^n is also a solution of (3.9), as can be verified by substitution. In this case the solution is:

$$c_n = \alpha r_1^n + \beta n r_1^n$$

Eq. (3.3) is an example of a first order difference equation while (3.9) is an example of a second order difference equation. Both equations are homogeneous and linear. For an analysis, such as the example given above, of the behaviour of such a two step method we have to consider both roots r_1 and r_2. For stability it is now required that both $|r_1| \le 1$ and $|r_2| \le 1$, if r_1 has multiplicity 2, i.e. $r_1=r_2$ then we must require $|r_1| < 1$.

A general k^{th} order linear difference equation with constant coefficients is given by:

$$\gamma_k c_{n+k} + \gamma_{k-1} c_{n+k-1} + \ldots + \gamma_0 c_n = \phi_n, \quad n = 0,1,\ldots \tag{3.14}$$

where γ_j, $j=0,1,\ldots,k$ are constants independent of n, and $\gamma_k \neq 0$, $\gamma_0 \neq 0$. A solution of such a difference equation will consist of a sequence $c_0, c_1, \ldots$, which we shall indicate by $\{c_n\}$. Let $\{\_n\}$ be the general solution of the corresponding homogeneous difference equation:

$$\gamma_k \hat{c}_{n+k} + \gamma_{k-1} \hat{c}_{n+k-1} + \ldots + \gamma_0 \hat{c}_n = 0, \quad n = 0,1,\ldots \tag{3.15}$$

If $\{Y_n\}$ is any particular solution of (3.14), then the general solution of (3.14) is $\{c_n\}$, where $c_n = \_n + Y_n$. The general solution of the homogenous equation can be denoted in terms of roots of the characteristic equation given by:

$$P(r) \equiv \gamma_k r^k + \gamma_{k-1} r^{k-1} + \ldots + \gamma_0 = 0 \tag{3.16}$$

In general, if P(r) has roots r_j, $j=1,2,\ldots,p$, and the root r_j has multiplicity μ_j, where $\mu_1 + \ldots + \mu_p = k$, then the general solution of (3.15) is $\{c_n\}$, where

$$\begin{aligned} c_n = {} & [d_{1,1} + d_{1,2} n + d_{1,3} n(n-1) + \ldots + d_{1,\mu_1} n(n-1)\ldots(n-\mu_1+2)] r_1^n \\ & + [d_{2,1} + d_{2,2} n + d_{2,3} n(n-1) + \ldots + d_{2,\mu_2} n(n-1)\ldots(n-\mu_2+2)] r_2^n \\ & + \ldots \\ & + [d_{p,1} + d_{p,2} n + d_{p,3} n(n-1) + \ldots + d_{p,\mu_p} n(n-1)\ldots(n-\mu_p+2)] r_p^n + \psi_n , \end{aligned}$$

the k constants $d_{j,l}$, $l=1,2,\ldots,\mu_j$, $j=1,2,\ldots,p$, being arbitrary.

Note that if ϕ_n is independent of n then we can choose as particular solution of (3.14):

$$\psi_n = \frac{\phi}{\sum_{j=0}^{k} \gamma_j}$$

Exercise 3.A:

Find the solution of the difference equation

$$c_{n+4} - 4c_{n+3} + 5c_{n+2} - 4c_{n+1} + 4c_n = 4, n = 0,1,...,$$
$$c_0 = 5, c_1 = 0, c_2 = -4, c_3 = -12$$

Numerical recipes not only consist of simple scalar equations. In most cases matrices are involved. A simple numerical recipe could have the following form:

$$c_{1,n+1} = a_{1,1} \cdot c_{1,n} + a_{1,2} \cdot c_{2,n} + ... + a_{1,M} \cdot c_{M,n}$$
$$\vdots$$
$$c_{M,n+1} = a_{M,1} \cdot c_{1,n} + a_{M,2} \cdot c_{2,n} + ... + a_{M,M} \cdot c_{M,n}$$

Such a set of formula's is denoted as:

$$\vec{c}_{n+1} = A\vec{c}_n$$

where

$$\vec{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_M \end{bmatrix} \text{ and } A = \begin{bmatrix} a_{1,1} \cdots a_{1,M} \\ \vdots \\ a_{M,1} \cdots a_{M,M} \end{bmatrix}.$$

If one assumes that A has M distinct eigenvalues then the general solution of this equation is given by:

$$\vec{c}_n = \sum_{m=1}^{M} d_m \lambda_m^n \vec{e}_m$$

where λ_m denotes an eigenvalue of A, $\vec{e}_m$ is an eigenvector and d_m, m=1,...,M are constants which are determined by the starting values or initial conditions of c. Obviously stability imposes a restriction on the complete set of eigenvalues or "spectrum" of the matrix A.

So far in this paragraph we have only dealt with linear equations. Nonlinear equations are much more difficult to analyze. This is considered to be beyond the scope of these lecture notes. An example of a non-linear formula is the following:

$$c_{n+1} = ac_n(1 - c_n)$$

The analysis of non-linear formula's is often based upon numerical experiments. The formula given above for example shows all kinds of interesting phenomena, see e.g Lauwerier (1989). Its stability not only depends on the values of a, but also of c_0.

Recurrent relations of the type described in this chapter have all kinds of interpretations. Examples are : geometrical interpretations leading sometimes, in the non-linear case, to interesting figures such as fractals, physical, chemical and many more. In each case the formula's must be consistent with some abstract model of an aspect of reality. In the next paragraphs we will explore this concept of consistency in a bit more detail.

3.3. Multistep methods for the approximation of initial value problems of ordinary differential equations.

The physics with which we will try to construct consistent computer formula's is described in terms of initial and boundary value problems of partial differential equations. It is concerned with flow including dissolved or suspended matters, such as salt, heat, silt, turbulent kinetic energy, heavy metals, the clearance of the water, BOD, etc. If the concentrations are be supposed to be well mixed, or if there is lack of detailed information or measurements such that averaged models that describe only averaged concentrations are acceptable then the behaviour of the concentrations is described in terms of initial value problems of ordinary differential equations.

Such an equation, in scalar form, can be denoted as:

$$\frac{dc}{dt} = f(c,t), \quad c(t_0) = C_0 \tag{3.17}$$

We seek for a solution in the range $t_0 \le t \le T$. We assume that (3.17) has a unique solution. Note that this assumption imposes some requirements on f(c,t) such as continuous differentiability, continuity and the Lipschitz condition. All these concepts are impossible to implement on a digital computer, where everything is finite and countable. So instead of solving (3.17) exactly we will try to construct another equation, that can be solved by a computer, by application of formula's of the type described in the previous chapter, and that gives an approximate solution of (3.17).

For this aim consider the sequence of points:

$$\{t_n\} \text{ defined by } t_n = t_0 + n\Delta t,\ n=0,1,2,...$$

The parameter Δt, that will be considered as a constant, is called steplength or if t denotes time it is also called the timestep. We do not seek an approximate solution on continuous interval $t_0 \le t \le T$ but on the discrete set $\{t_n | n=0,1,...,(T-t_0)/\Delta t\}$. Let c_n be an approximation to the theoretical solution at t_n, i.e. to $c(t_n)$, and let $f_n \equiv f(c_n,t_n)$. If a computational method for determining the sequence $\{c_n\}$ takes the form of a linear relationship between $c_{n+j}, f_{n+j}, j=0,1,...,k$, we call it a "linear multistep method of step number k", or a "linear k step method". The general linear multistep method can be written as:

$$\sum_{j=0}^{k} \alpha_j \frac{c_{n+j}}{\Delta t} = \sum_{j=0}^{k} \beta_j f_{n+j} \tag{3.18}$$

Now the problem of finding a solution c(t) of (3.17) has been replaced by finding a sequence $\{c_n\}$ which satisfies the difference equation (3.18). Note that, since $f_n = f(c_n,t_n)$ is in general a non-linear function of c_n, (3.18) is in general a non-linear difference equation, given by:

$$c_{n+k} - \frac{\Delta t \beta_k}{\alpha_k} f(c_{n+k}, t_{n+k}) = RHS$$

where RHS is a known function of the previously calculated values $c_{n+j}, f_{n+j}, j=0,1,2,...,k-1$.

The solution of such an equation might imply the application of iterative methods, such as Newton Raphson, see e.g. Press et al.(1988). If f_n is a linear function then (3.18) is straightforward to solve, and if f_n is not a scalar, as is assumed throughout this paragraph, but a matrix, then the solution of (3.18) implies the inversion of a matrix.

If $\beta_k=0$ then there is no difficulty in solving (3.18). A value for c_{n+k} can be computed directly from values of RHS, i.e. from values of c_{n+j}, f_{n+j}, j=0,1,...,k-1. We say that (3.18) is "explicit" if $\beta_k=0$, and "implicit" if $\beta_k \neq 0$.

Note that (3.18) needs starting values, if k=1 then just the initial condition of (3.17) are used for that, however for k>1 additional starting values are needed. In general this problem can be solved by using methods of lower stepsize during the start.

There are various ways to find the coefficients α_j, β_j appearing in (3.18). Based on Taylor series expansions, interpolation and numerical integration, see Lambert (1990). We limit ourselves to the requirements, such as

"consistency", "convergence" and "stability", that methods should fulfil once the coefficients α_j,β_j are found. In the following paragraphs we will define and elaborate these important notions.

3.4. Consistency, local truncation error.

A necessary condition for a finite difference equation for the approximation of a differential equation is that it is consistent with that equation. This means that the finite difference equation is exactly equivalent to the differential equation, at each grid point, in the limiting case that $\Delta t \to 0$.

Before we give a more precise definition of this notion we first define the difference operator $D_{\Delta t}$, associated with the linear multistep method (3.18), as follows:

$$D_{\Delta t}[c(t)] = \sum_{j=0}^{k} \left\{ \frac{\alpha_j c(t + j\Delta t)}{\Delta t} - \beta_j f[c(t + j\Delta t), t + j\Delta t] \right\} \tag{3.19}$$

where $t_0 \le t \le T$, c(t) satisfies (3.17) and is continuously differentiable on $(t_0,T]$.

A linear multistep method is said to be "consistent" if and only if:

$$\lim_{\Delta t \to 0} D_{\Delta t}[c(t)] = 0$$

Example:

Consider "Euler's rule", the simplest of all linear multistep methods, given by:

$$\frac{c_{n+1} - c_n}{\Delta t} = f_n$$

According to our definition consistency requires:

$$\lim_{\Delta t \to 0} \left\{ \frac{c(t + \Delta t) - c(t)}{\Delta t} - f[c(t), t] \right\} = 0$$

This expression can be written as:

$$\lim_{\Delta t \to 0} \frac{c(t + \Delta t) - c(t)}{\Delta t} - f[c(t), t] = 0$$

By definition of the differential operator *d/dt* in (3.17) this expression is exactly equivalent with (3.17) from which it follows that Euler's rule is consistent.

In general consistency is less trivial to verify. Verification of consistency is usually based upon Taylor's series expansion given by:

$$c(t + \Delta t) = c(t) + \Delta t c^{(1)}(t) + \frac{\Delta t^2}{2!} c^{(2)} + \frac{\Delta t^3}{3!} c^{(3)} + \ldots$$

where

$$c^{(q)}(t) = \frac{d^q c}{dc^q}, \quad q = 1, 2, \ldots$$

This expansion is substituted into (3.19) together with $f(c,t)=c^{(1)}(t)$. This yields after some collecting:

$$D_{\Delta t}\left[c(t)\right] = \frac{b_0}{\Delta t} c(t) + b_1 c^{(1)}(t) + b_2 \Delta t c^{(2)}(t) + \ldots + b_q \Delta t^{q-1} c^{(q)}(t) + \ldots \tag{3.20}$$

where b_q are constants.

Consistency can now be translated into the requirement that in (3.20) at least $b_0=0$ and $b_1=0$.

Exercise 3.B:

Verify that for a linear multistep method the following relations must hold:

$$\sum_{j=0}^{k}\alpha_j = B_0 = 0$$

$$\sum_{j=1}^{k} j\alpha_j - \sum_{j=0}^{k}\beta_j = B_1 = 0$$

Hint: try this first for a step number k=1 and then 2.

We can also define the order of consistency:

> **The difference operator (3.19), and the associated linear multistep method (3.18) is said to be consistent of order q if, in (3.20), $b_0=b_1=...=b_q=0$, $b_{q+1}\neq 0$.**

Example

For Euler's rule we get the following:

$$\begin{aligned} D_{\Delta t}[c(t)] &= \frac{c(t+\Delta t) - c(t)}{\Delta t} - f(c,t) \\ &= \frac{c(t+\Delta t) - c(t)}{\Delta t} - c^{(1)}(t) \\ &= \frac{\sum_{q=0}^{\infty}\frac{\Delta t^q}{q!}c^{(q)} - c(t)}{\Delta t} - c^{(1)}(t) \\ &= \sum_{q=2}^{\infty}\frac{\Delta t^{q-1}}{q!}c^{(q)} \end{aligned}$$

In other words Euler's rule is consistent of order 1.

Related to order of consistency we can define the local truncation error or discretization error:

The "local truncation error" at t_{n+k}, denoted by E_{n+k}, of method (3.18) is defined to be the expression $D_{\Delta t}[c(t_n)]$ given by (3.19), where c(t) satisfies (3.17). From (3.20) it follows that when the consistency is of order p the local truncation error is given by:

$$D_{\Delta t}[c(t_n)] = \sum_{q=p+1}^{\infty} b_q \Delta t^{q-1} c^{(q)}(t_n)$$

The first term of this series expansion, given by:

$$b_{p+1}\Delta t^p c^{(p+1)}(t_n)$$

is called the "principal local truncation error".

Some authors multiply E_{n+k} by $\Delta t/\alpha_{n+k}$, see e.g. Lambert (1990), and call the result the local truncation error. But with Richtmyer and Morton (1967) we prefer our definition because we consider the local truncation error as a measure how well the finite difference equation (3.18) replaces the differential equation (3.17) and not how well the exact solution $c(t_n)$ is approximated by c_n, although there is a relation, see Lambert (1990).

Note that the discretization error gives us only an idea of the order of magnitude of this expression since we do not know c(t), and thereby $c^{(q)}(t)$, exactly. It is in fact an asymptotic expression that gives us only information on the asymptotic behaviour as $\Delta t \to 0$.

Examples

We conclude this paragraph with a few examples of LM methods:

Euler's explicit rule: $$\frac{c_{n+1} - c_n}{\Delta t} = f_n, \quad E_{n+1} = \frac{\Delta t}{2} c^{(2)}(t_n) + O(\Delta t^2)$$

Euler's implicit rule: $$\frac{c_{n+1} - c_n}{\Delta t} = f_{n+1}, \quad E_{n+1} = -\frac{\Delta t}{2} c^{(2)}(t_n) + O(\Delta t^2)$$

Trapezoidal rule: $$\frac{c_{n+1} - c_n}{\Delta t} = \frac{1}{2} f_n + \frac{1}{2} f_{n+1}, \quad E_{n+1} = -\frac{\Delta t^2}{12} c^{(3)}(t_n) + O(\Delta t^3)$$

Midpoint rule: $$\frac{c_{n+2} - c_n}{2\Delta t} = f_{n+1}, \quad E_{n+1} = \frac{\Delta t^2}{6} c^{(3)}(t_n) + O(\Delta t^3)$$

2nd order Backward differencing: $$\frac{3c_{n+2} - 4c_{n+1} + c_n}{2\Delta t} = f_{n+2}$$

θ method: $$\frac{c_{n+1} - c_n}{\Delta t} = \theta f_{n+1} + (1 - \theta) f_n$$

(applied by many commercial packages for simulation of 1 dimensional flows, θ=0 implies Euler explicit, θ=½ Trapezoidal and θ=1 Euler implicit)

Exercise 3.C:

Verify the local truncation errors when given above and find the truncation errors if not given.

3.5. Global error, convergence, zero stability, equivalence theorem,absolute stability

A basic property that any acceptable discretization method must have is that the solution $\{c_n\}$ generated by the method must converge in some sense to the theoretical solution c(t) as the steplength tends to zero.

Before we give a more precise definition of this concept we first define the global error, e_n, as the difference between the theoretical solution c(t) and the numerical solution c_n i.e. $e_n=c(t_n)-c_n$.

We define this as follows:

The linear multistep method (3.18) is said to be "convergent" if we have that

$$\lim_{\substack{\Delta t \to 0 \\ n=\frac{t-t_0}{\Delta t}}} e_n = 0$$

holds for all $t \in (t_0, T]$.

Note that the essential characteristic of this limiting process is that n tends to infinity while T has a fixed value.

Related to this definition of convergence is the following condition:

$$\lim_{\substack{\Delta t \to 0 \\ n=\frac{t-t_0}{\Delta t}}} |c_n| \le K_0 |c_0|$$

where K_0 denotes some constant, c_n denotes any member of the set of all possible solutions of (3.18) and c_0 denotes some initial condition.

This limiting process is the same as for the definition of convergence, but it considers only solutions of the linear multistep method. In other words in this definition there is no relation with the exact solution of the differential equation. If this condition is fulfilled by (3.18) then (3.18) is said to be *zero stable*.

Let us consider (3.18) again, but for the trivial case that f=0. After multiplication with Δt this yields:

$$\sum_{j=0}^{k} \alpha_j c_{n+j} = 0$$

According to (3.15) and (3.16) the general solution of this equation is given by:

$$c_n = \sum_{j=1}^{k} d_j r_j^n$$

where r_j are roots of the following polynomial:

$$p_1(r) \equiv \sum_{j=0}^{k} \alpha_j r^j = 0$$

This polynomial is called the "first characteristic polynomial" of the linear multistep method (3.18). The second characteristic polynomial is defined by:

$$p_2(r) \equiv \sum_{j=0}^{k} \beta_j r^j$$

If we assume that the roots of the first characteristic polynomial are simple then for zero-stability we must have that the roots of the first characteristic polynomial have roots with a modulus not greater than one.

Roots with a multiplicity greater than one must have a modulus smaller than one.[3]

We can denote the numerical solution c_n as $c_n=c(t_n)-e_n$. The exact solution $c(t_n)$ is supposed to be bounded or finite on the interval (0,T]. From this it follows that boundedness of c_n implies finiteness of e_n. Although the definition of zero stability has no relation with the exact solution c(t), yet the definition of zero-stability is equivalent to the requirement that the set of all possible errors e_n remains finite for the limiting process $\Delta t \to 0$, where $n=(t-t_0)/\Delta t$ and t remains fixed. Based on this consideration the following theorem is probably not very surprising:

> **The necessary and sufficient conditions for a linear multistep method to be convergent are that it be consistent and zero-stable.**

The importance of zero-stability lies in this theorem. It is due to Dahlquist, a proof can be found in Henrici (1962). Note that for partial differential equations a similar theorem, called the *Lax equivalence theorem*, exists. We will treat that later on.

The importance of this theorem lies in the fact that now consistency and zero-stability are sufficient to ensure convergence, while both properties are relatively easy to verify.

In most cases simulations are executed of stable physical phenomena, i.e. of phenomena that remain finite for a long period of time. An example of such a phenomenon is the tidal elevation. Differential equations that describe this kind of physical processes have solutions that remain finite also in the limiting case $T \to \infty$. A relevant question is therefore what happens with the numerical solution c_n, $n=1,...,N$, $n=(t-t_0)/\Delta t$ (i.e. $N=(T-t_0)/\Delta t$) in the limiting case $T \to \infty$ while Δt remains fixed. This consideration leads us to the following definition:

A linear multistep method (3.18) is called *absolutely stable* if the following condition is satisfied:

$$\lim_{\substack{T\to\infty \\ n=1,\ldots,\frac{T-t_0}{\Delta t}}} |c_n| \le K_1 |c_0|$$

where K_1 and Δt are constants.

Note that a method which has no interval of absolute stability has no practical value.

---

[3]Note that in literature on LM methods zero-stability is defined as the moduli of roots of the first characteristic polynomial to be no greater than one or less than one for roots that are not simple. We follow a different approach since we also use this definition later on for the approximation of partial differential equations. In essence however both definitions are equivalent.

In most cases absolute stability is studied only for linear problems, i.e. for problems for which f(c,t)=λc. By this assumption we get the so-called linear "test problem" given by:

$$\frac{dc}{dt} = \lambda c,\ c(t_0) = C_0 \tag{3.21}$$

When (3.18) is applied for the approximation of (3.21) one obtains:

$$\sum_{j=0}^{k} \alpha_j \frac{c_{n+j}}{\Delta t} = \lambda \sum_{j=0}^{k} \beta_j c_{n+j}$$

This equation can be rewritten as:

$$\sum_{j=0}^{k} \alpha_j c_{n+j} - \Delta t \lambda \sum_{j=0}^{k} \beta_j c_{n+j} = 0$$

The general solution can be written as:

$$c_n = \sum_{j=0}^{k} d_j r_j^n$$

Here r_j are roots of the polynomial equation given by:

$$P(r) \equiv p_1(r) + \Delta t p_2(r) = \sum_{j=0}^{k} \alpha_j r^j - \lambda \Delta t \sum_{j=0}^{k} \beta_j r^j = 0 \tag{3.22}$$

Here P(r) is the characteristic or *stability polynomial* of (3.18).

Obviously for absolute stability one must have that $|r_j| \le 1$ if r_j is a single root of (3.22) and $|r_j| < 1$ if r_j is a multiple root.

In general absolute stability is a more severe restriction than zero-stability. Absolute stability imposes restrictions on the timestep where as zero stability is a condition that is fulfilled or not. In other words if a method is not zero-stable then changing the timestep will not help to change that situation. We will see later on that for the numerical approximation of partial differential equations this situation is entirely different. There also zero-stability might depend upon the stepsize.

The regions of absolute stability are often displayed in the complex λΔt plane. This way we cover also the cases in which the eigenvalue λ of the system is complex, as it is with oscillating systems. Figure 3.2 shows the region of absolute stability of the explicit Euler method.

Note that because of our assumption that for absolute stability we only consider problems with a bounded solution for an infinite interval of t only the left half of the complex λΔt plane is of interest because the solution of our test-equation is given by $c(t)=C_0e^{\lambda t}$.

If we consider the stability polynomial we will see that in case Δt=0 then $r_1=1$ must be a root of the equation P(r)=0. This is because consistency requires that:

$$\sum_{j=0}^{k} \alpha_j = 0$$

This has been treated in the previous paragraph as exercise 3.B.

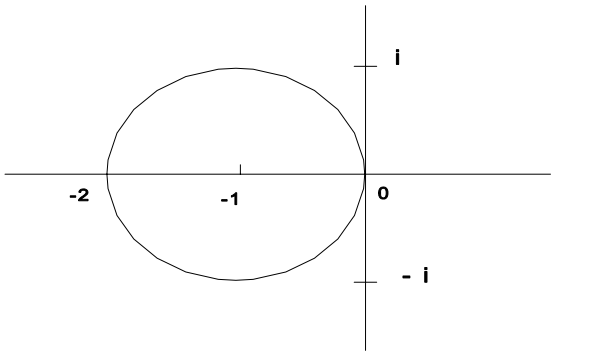


Figure 3.2. Region of absolute stability for the explicit Euler rule in the complex h-plane, $h=\lambda\Delta t$

We now define r_1 as the *principal root* of the characteristic equation.

All other roots are called *spurious roots*. Spurious roots are often strongly influencing the stability of the method. Therefore control of spurious roots is important. One step methods do not have spurious roots.

Example

Consider the fourth order *Simpson rule* given by:

$$\frac{c_{n+2} - c_n}{2\Delta t} = \frac{1}{6} f_{n+2} + \frac{4}{6} f_{n+1} + \frac{1}{6} f_n$$

The first characteristic equation is given by:

$$r^2 - 1 = 0$$

The roots $r_{1,2}$ are +1 and -1, i.e. the method is zero-stable.

The stability equation is given by:

$$(1 - \frac{1}{3}h)r^2 - \frac{4}{3}hr - (1 + \frac{1}{3}h) = 0$$

where $h=\Delta t\lambda$.

The location of the roots of P(r) is a classical problem. To have a first impression one can use the computer and compute the maximum value of $|r_1|$ and $|r_2|$ for a sector in the left part of the complex h plane. If one does that it will turn out that near the origin of the complex h plane all moduli are greater then one. This indicates that the Simpson rule has an empty region of absolute stability[4].

This example shows that despite the order of accuracy a numerical method can be rather useless for practical application. The Simpson rule is an example.

The *trapezoidal rule* has only one characteristic root, $r_1=(1+h/2)/(1-h/2)$. In this case the region of absolute stability covers the whole left-half plane. If a LM method is absolutely stable for each h where Re $h \leq 0$ we call such a method *A-stable*. A-stability is a desirable property of a numerical integration method since, at least for linear problems, there are no stability limits for the stepsize. However only a limited amount of methods have this property as indicated by the following theorem:

---

[4]A more sophisticated way to prove the instability of Simpsons rule can be obtained by the use of so-called *Schur polynomials*. For this consider a general k^{th} order polynomial with complex coefficients:

$$\phi(r) = \gamma_k r^k + \gamma_{k-1} r^{k-1} + \ldots + \gamma_k r + \gamma_0 = \sum_{j=0}^{k} \gamma_j r^j$$

$\phi(r)$ is said to be a Shur polynomial if its roots r_j satisfy $|r_j|<1$, j=1,...,k. Now we define the polynomials:

$$\hat{\phi}(r) = \sum_{j=0}^{k} \overline{\gamma}_{k-j} r^j$$

$$\phi_1(r) = \frac{1}{r}[\hat{\phi}(0)\phi(r) - \phi(0)\hat{\phi}(r)]$$

Here $\overline{\gamma}_j$ denotes the complex conjugate of γ_j. Clearly $\phi_1(r)$ has degree at most k-1. Therefore $\phi_1(r)$ is called the reduced polynomial. By a theorem of Schur, see Miller (1971), $\phi(r)$ is a Schur polynomial if and only if $|\hat{\phi}(0)| > |\Phi(0)|$ and $\phi_1(r)$ is a Schur polynomial. For our example $\phi_1(r)$ is a first order polynomial of which the zero is easy to find. In general recurrent application of the Schur theorem leads to simple criteria.

(i) An explicit linear multistep method cannot be A-stable. (ii) The order of an A-stable implicit multistep method cannot exceed two. (iii) The second order A-stable implicit linear multistep method with the smallest truncation error is the Trapezoidal rule.

Note however that the trapezoidal rule is known to have bad stability properties for non-linear problems and for problems where stiffness plays a role.

3.6. Systems of equations, the problem of stiffness.

In practice we are faced with initial value problems that involve not just a single first-order differential equation but a system of m simultaneous first order equations.

In this section we consider the following problem:

$$\frac{d\vec{c}}{dt} = \vec{f}(\vec{c},t), \quad \vec{c}(t_0) = \vec{C}^0 \tag{3.23}$$

where we have the following vector valued functions:

$$\vec{c} = \begin{bmatrix} c_1 \\ c_2 \\ . \\ . \\ . \\ c_m \end{bmatrix}, \ \vec{f} = \begin{bmatrix} f_1 \\ f_2 \\ . \\ . \\ . \\ f_m \end{bmatrix}, \ \vec{C}^0 = \begin{bmatrix} C_1^0 \\ C_2^0 \\ . \\ . \\ . \\ C_m^0 \end{bmatrix}$$

A general linear multistep method applied to these equations is denoted as follows:

$$\sum_{j=0}^{k} \frac{\alpha_j}{\Delta t} c^{n+j} = \sum_{j=0}^{k} \beta_j \ f^{n+j} \tag{3.24}$$

We now use superscripts instead of subscripts to indicate that we deal with vectors instead of scalars.

All definitions that we have used so far can easily be extended to vectors. For example the local truncation error becomes a vector, the difference operator $D_{\Delta t}$ associated with (3.24) becomes a vector etc. Absolute values will have to be changed into norms. Convergence now becomes the requirement that:

$$\lim_{\substack{\Delta t \to 0 \\ n=\frac{t-t_0}{\Delta t}}} \left\| c^n - \vec{c}(t_n) \right\| = 0$$

holds for all $t \in (t0, T]$ and for all solutions $\{c^n\}$ of the vector difference equation (3.24). Here we have assumed some suitable starting procedure such as the application of an LM method with stepnumber one. $\|...\|$ denotes some suitable vector norm for example given by:

$$\left\| \vec{c} \right\| = \sqrt{\sum_{j=1}^{m} (c_j)^2}$$

The definition of zero-stability now requires boundedness of the norm $\|e^n\|$.

Under the assumption of appropriate starting procedures also in the case of systems zero-stability and consistency are sufficient and necessary for convergence.

For absolute stability we assume the vector valued function f(c,t) of (3.24) to be a constant m by m matrix denoted by J.

The test equation then becomes:

$$\frac{d\vec{c}}{dt} = J\vec{c}$$

Application of (3.24) to this equation gives:

$$\sum_{j=0}^{k}(\alpha_j I - \Delta t \beta_j J)c^{n+j} = 0 \tag{3.25}$$

where I is the m by m unit matrix. We now assume further that the eigenvalues λ_i, i=1,...,m are distinct. Then there exists a non-singular matrix H such that:

$$H^{-1} J H = \Lambda = \begin{bmatrix} \lambda_1 0 \ldots 0 \\ 0 \lambda_2 \ldots 0 \\ \vdots \\ 0 \ldots 0 \lambda_m \end{bmatrix}$$

Pre-multiplying (3.25) by H^{-1} and defining d^n by

$$c^n = Hd^n$$

gives:

$$\sum_{j=0}^{k} H^{-1}(\alpha_j I - \Delta t \beta_j J) H d^{n+j} = 0$$

Because of the definition of H this equation is equal to:

$$\sum_{j=0}^{k}(\alpha_j - \Delta t \beta_j \Lambda) d^{n+j} = 0$$

Since *I* and *Λ* are diagonal matrices, the components of this equation are uncoupled; that is it be may be written in the form:

$$\sum_{j=0}^{k}(\alpha_j - \Delta t \beta_j \lambda_j) d_i^{n+j} = 0, \quad i = 1,2,\ldots,m$$

Each of the m equations denoted here is independent of the others. Each equation is now exactly of the form (3.21). It now becomes clear that for this test equation also complex numbers are to be taken into account for λ.

Absolute stability has to be checked for each of the *m* equations.

At this point we consider the following differential equation:

$$\frac{dc}{dt} = \begin{bmatrix} -500.5 & 499.5 \\ 499.5 & -500.5 \end{bmatrix} \cdot c, \quad c(0) = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad c = \begin{bmatrix} c_1(t) \\ c_2(t) \end{bmatrix} \tag{3.26}$$

The solution of this system is given by:

$$c_1(t) = 1.5e^{-t} + 0.5e^{-1000t}$$
$$c_2(t) = 1.5e^{-t} - 0.5e^{-1000t}$$

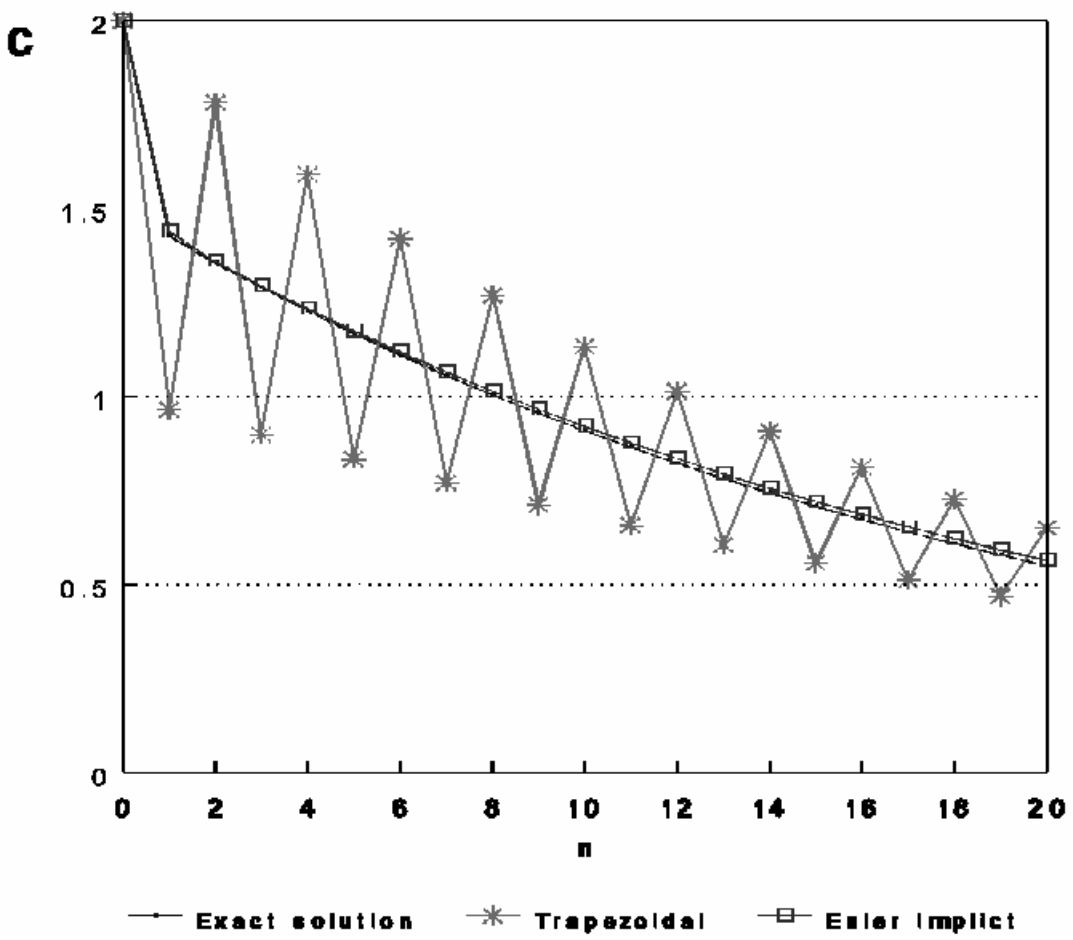


figure 3.3

Both parts of the solution contain parts, or solution modes, that decay very rapidly if t increases. The eigenvalues of this system are $\lambda_1=-1$ and $\lambda_2=-1000$. If (3.26) is approximated by the explicit Euler rule we get the stability conditions $|\Delta t|\#2$ and $|-1000\Delta t|\#2$. This means that λ_2 determines the stability condition. This eigenvalue however belongs to a solution mode that decays very rapidly. After a short period of time this mode is already less than the smallest number, not equal to zero, that is representable by a digital computer. This is a typical example of what numerical analysts call the problem of *stiffness*. The ratio of the eigenvalue with the largest modulus and the eigenvalue with the smallest modulus is called the *stiffness ratio*. A solution mode which is negligible for the overall solution is the limiting factor for stability or the maximum stepsize. This leads to very inefficient numerical integration.

In stead of using the first order Euler rule one could have used the trapezoidal rule. Now the timestep is unrestricted. If one chooses a timestep of 0.05 seconds one will find out that the numerical results are inaccurate and highly oscillating, see figure 3.3. Again a much smaller timestep has to be taken in order to get accurate results.

Finally we consider the implicit Euler rule. Now we observe accurate results already for $\Delta t=0.05$ seconds, despite of the fact that the order of consistency of this method is only one while the trapezoidal rule has an order of consistency of two.

To explain these observations one must observe the solution modes of the numerical solution c^n.

First we consider the explicit Euler rule. For our problem the explicit Euler rule becomes:

$$\frac{c^{n+1}-c^n}{\Delta t}=\begin{bmatrix}-500.5 & 499.5\\ 499.5 & -500.5\end{bmatrix}c^n$$

Taking into account the initial conditions of (3.26) then the solution of this difference equation can be denoted as:

$$c^n = 1.5\begin{bmatrix}1.0\\1.0\end{bmatrix}p_1^n + 0.5\begin{bmatrix}0.5\\-0.5\end{bmatrix}p_2^n$$

where p_1=1-Δt and p_2=1-1000Δt. These are called the *numerical solution modes*. Since we consider a one step method the number of numerical modes equals the number of modes in the analytical solution.

Clearly p_2 determines the stability and also oscillatory solutions. For $\Delta t \leq 0.002$ the solution is stable while for Δt≤0.001 the solution will show no sign of numerical oscillations.

In the same way we can analyze the trapezoidal rule for this case. We will now find that the numerical solution can be denoted as:

$$c^n = 1.5\begin{bmatrix}1.0\\1.0\end{bmatrix}q_1^n + 0.5\begin{bmatrix}0.5\\-0.5\end{bmatrix}q_2^n$$

where q_1=(1-0.5Δt)/(1+0.5Δt) and q_2=(1-500Δt)/(1+500Δt).

Although both modes are stable inaccurate solutions were obtained for Δt=0.05. In this case q_2=-0.92. This mode causes oscillating, slowly decaying solutions. In the limiting case that Δt→∞ this mode becomes -1, i.e. an oscillating and undamped mode.

Finally we analyze the Euler implicit method. The numerical solution in this case is denoted by:

$$c^n = 1.5\begin{bmatrix}1.0\\1.0\end{bmatrix}r_1^n + 0.5\begin{bmatrix}0.5\\-0.5\end{bmatrix}r_2^n$$

where r_1=1.0/(1+Δt) and r_2=1.0/(1+1000Δt).

In this case for Δt=0.05 r_2=0.02. This leads to rapid decay, without oscillations, of the second solution mode. Note that in the limiting case Δt→∞ $r_{1,2}$=0. Observations of these kind has lead to the construction of methods particularly for the approximation of stiff systems of equations, see Gear (1971). Very attractive properties for a method for the approximation of stiff problems are the following:

(i) A-stability, (ii) when applied to the testproblem ∂c/∂t=λc, c_n→0 if Δt→∞.

If these conditions are fulfilled a method is sometimes called *stiffly A-stable*. These requirements however are very severe. Therefore other concepts are defined as well such as A(α) stability and stiff stability, for these notions however the reader is referred to Lambert (1990). Since Gear many methods for stiff problems were constructed, for an overview see Gupta et al.(1985).

Among the methods that are considered to be suitable for stiff systems we find the backward differentiation methods. The implicit Euler rule and the second order backward differentiation method, that were treated in this chapter belong to this class. For problems with a mild stiffness ratio the θ method, with 0.5<θ<1, is a good compromise. Note however that there are much better methods available, including automatic error control, than the methods described here. One can find these methods in subroutine libraries such as NAG and IMSL. The only purpose of our introduction is to demonstrate roughly the problem of stiffness, i.e. the problem of approximating systems of differential equations of which the Jacobian (J=∂f/∂c) is characterized by having widely separated eigenvalues. As we will see later on this situation is typical when partial differential equations are approximated by the method of lines. Also in ecological, chemical or biological modelling the problem of stiffness occurs. An example of a class of ecological models are the so-called predator-prey relations. These equations, of which we have given an example in the beginning of this chapter, are generally non-linear ordinary differential equations and sometimes they are stiff.

Different timescales that lead to the application of stiffly stable methods, are not only the result of the timescales in the so-called "homogeneous" part of the solution. To illustrate this we consider a simple mass spring system as given by figure 3.4.

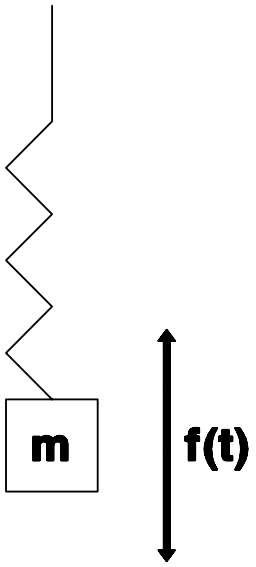


figure 3.4

An external force $f(t)=f_0 cos Wt$ acts on m. This yields the following, inhomogeneous, system of equations:

$$\frac{dv}{dt}+\frac{A}{m}v+\frac{K}{m}x=\frac{f_0}{m}\cos\omega t$$

$$\frac{dx}{dt}-v=0$$

where m is the mass, K denotes the stiffness and A is the friction coefficient. The general solution x(t) of this equation can be represented by:

$$x(t)=x_H(t)+x_I(t)$$

where $x_H(t)$ denoted the general solution of the homogeneous and $x_I(t)$ is a solution of the inhomogeneous equations. The homogeneous solution is given by:

$$x_H(t)=\alpha e^{-(A/2m)t}cos(\omega_1 t+\beta)$$

where α and β are constants that are determined by initial conditions and $W_1=(K/m-A^2/4m^2)^{1/2}$. It is assumed that $K/m > A^2/4m^2$ (oscillatory solution). The inhomogeneous or forced solution $x_I(t)$ is given by:

$$x_I(t)=\frac{f_0}{\sqrt{m^2(\omega_0^2-\omega^2)^2+A^2\omega^2}}\cos(\omega t-\varphi)$$

$$where\ \cos\varphi=\frac{m(\omega_0^2-\omega^2)}{\sqrt{m^2(\omega_0^2-\omega^2)^2+A^2\omega^2}}$$

$$\omega_0=\sqrt{\frac{K}{m}}$$

The homogeneous solution, or transient, will tend to zero when time proceeds. The inhomogeneous solution, or forced oscillation, or steady state will remain. If the timescales of the homogeneous solution 2m/A and W_1 are very different from the timescale of the steady state solution W then stiffly stable methods might be atrractive as well for this case. Only in this case numerical damping should be working also along the imaginary axis of the h plane. This is especially the case if the initial condition is in fact non-physical and if only the steady state is of interest. For many tidal simulations there is a similar situation.

3.7. Summary, concluding remarks.

In the last paragraph of this chapter we briefly summarize the concepts that were treated in this chapter. The main issue has been the mapping of ordinary differential equations onto recurrence or difference equations that can be solved with digital computers. For this the following steps were considered:

Step 1:

Translation of a real-life problem into a model that is an *initial value problem for a first order differential equation* given by (3.23). Such an equation must be such that a unique solution exists.

Step 2:

Mapping of (3.23) onto a set of *difference equations*. This process is called *discretization*. The class of methods that were treated in this chapter were the so-called *linear multistep methods* given by (3.24). In order to obtain useful approximations difference equations must fulfil the following conditions:

(I) consistency
(II) convergence
(III) zero-stability
(IV) absolute-stability

Note that (I) + (III) = (II).

Sometimes additional properties of difference equations are useful such as:

(V) unconditional stability
(VI) stiff A-stability

Step 3:

Finally the difference equations are to be solved. If a method is *explicit* then the solution is trivial. *Implicit* difference equations are sometimes difficult to solve. Iterative methods for non-linear equations, such as Newton iteration or matrix inversion methods are to be used in certain instances.

Exercise 3.D:

(1) Give a definition of all the notions written in italics given in this paragraph,
(2) describe the notion of spurious roots and
(3) describe the notion of local truncation error.

The following linear multistep methods, that were given in this chapter, are useful fr practical applications:

Explicit Euler rule
Implicit Euler rule
Trapezoidal rule
θ method
Midpoint rule
Second order backward differentiation method

Exercise 3.E:

Give the stepnumber and the coefficients α_k and β_k that define the methods mentioned above in italics. Determine the local truncation error.

Finally it is to be noted that we have given only a very superficial introduction to the numerical solution of ordinary differential. We have been treating only linear multistep methods[5]. Other well-known classes are for example *Runge Kutta methods* and *extrapolation methods*. See e.g. Lambert (1990) or Press et al.(1988). The latter not only treats theory but gives also FORTRAN and PASCAL listings of many algorithms. To avoid typing a floppy disk containing all the listings is available as well. Today's methods are all equipped to control errors either locally or globally. The numerical libraries *IMSL* and *NAG* contain many good methods for the integration of ordinary differential equations.

Finally we would like to remark that this chapter was primarily written as an introduction to the numerical solution of differential equations in general. In other words we have used these equations as the simplest possible vehicle to introduce some important notions that have a general validity. The following chapter, on initial value problems of partial differential equations, will use these notions, it will also show however where there are important deviations.

[5] Of these methods we treated only up to 2 step methods while many more possibilities exist within this framework see e.g. Shampine and Gordon[]. We did not treat for example *predictor corrector* methods although they belong also to the class of linear multistep methods. A well-known example of a predictor corrector method is given by:

$$predictor: \frac{c^{[1]} - c^n}{\Delta t} = f^n$$

$$corrector: \frac{c^{n+1} - c^n}{\Delta t} = \tfrac{1}{2} f^{[1]} + \tfrac{1}{2} f^n$$

For a treatment the reader is referred to the already mentioned literature.

Chapter 4 Time dependent partial differential equations, basic principles

4.1. Introduction

This chapter deals with the numerical solution of time dependent partial differential equations (PDE's). If simple box models are no longer sufficient then also spatial variations are to be taken into account. This will lead to mathematical description that not only have the time t, but also the space x,y,z as independent variables. The resulting equations are PDE's. These equations are to be completed with initial conditions and/or boundary conditions. The resulting equations must be such that the following conditions are satisfied:

(i) the solution exists
(ii) the solution is unique
(iii) the solution depends continuously on the boundary conditions and/or initial conditions

If these conditions are satisfied we say that the problem is *well-posed*. A problem which is not well-posed cannot be solved numerically.

As mentioned in section 2.6, PDE's are generally divided into three categories:

(I) parabolic equations
(II) hyperbolic equations
(III) elliptic equations

We will restrict ourselves to initial value problems, i.e. the equations of categories I and II.

Consider the convection-diffusion equation:

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} - K\frac{\partial^2 c}{\partial x^2} = \lambda c \tag{4.1}$$

This is the simplified linear transport equation. The numerical approximation of this equation will be the first subject of this chapter.

This equation is parabolic but for sufficiently small values of K it can be considered as hyperbolic. Elliptic problems will hardly be dealt with in these lecture notes.

This chapter deals with three cases, in section 4.2 the case v=0, λ=0 will be treated, in section 4.3 the case K=0, λ=0 and in paragraph 4.4 the full equation will be dealt with. The case v=0, K=0 has been the subject of chapter 3.

An example of a hyperbolic set of equations is given by the shallow water equation. This equation is the subject of section 4.5.

4.2. The consistent discretization of the simplest diffusion equation

Consider the diffusion equation given by:

$$\frac{\partial c}{\partial t} - K\frac{\partial^2 c}{\partial x^2} = 0 \tag{4.6}$$

This equation has to be completed with two boundary conditions and with initial conditions. One boundary condition is given on each side of the spatial domain, R={0,X}. The time interval is given by (0,T]. Initial conditions are given at t=0. For this equation it is allowed to prescribe each of the three types of boundary conditions. In most cases a boundary condition is prescribed of type (A), the Dirichlet type, or of type (B), the Von Neumann type. Type (A) describes the solution itself at the boundary. This boundary equation allows transport of matter through the boundary. Type (B) describes the gradient at the boundary. A very common boundary condition of this type is that

the gradient is prescribed to be equal to zero. This means that gradient transport of matter or other substances such as heat can not take place at the boundary.

As for the ODE's in the first chapter (4.6) has to be translated in a *consistent* way into a *stable* set of recurrent relations. The solution of this set of recurrent relations has to be *convergent* to the solution of the set of PDE's that were approximated.

Before dealing again with the concepts of consistency, stability and convergence we first describe the discretization process that we use mostly within the frame work of these lecture notes. This process of discretization is called the *method of lines*. It consists of two steps:

(i) Translation of the PDE (4.6) into a discrete system of ODE's. This step is called the *semi-discretization* of (4.6)

(ii) Translation of the semi-discrete system of equations into a set of fully discrete equations by an approximation method for ODE's such as the linear multistep methods that were dealt with in the previous chapter.

Before we can take the first step we have to define a *spatial grid*. We consider a set of points given by $\{x_0,x_1,...,x_i,...,x_K\}$ where $x_K=X$,

$x_{i-1}<x_i<x_{i+1}$, $i=1,...,K-1$ and $\Delta x_i=x_{i+1}-x_i$. We will consider a equidistant grid i.e. $\Delta x_i=\Delta x=(X-x_0)/K$.

There are various ways to obtain a set of semi-discrete equations. The simplest way of semi-discretization is provided by replacing the spatial derivatives by equivalent finite difference expressions. An example is given by:

$$\frac{dc_i}{dt} - K\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2} = 0 \qquad (4.7)$$

where $i=1,...,K-1$. We assume boundary conditions of type (A) prescribed at x_0 and x_K.

The approximation of the second derivative can be arrived at if one remembers that

$$\frac{\partial^2 c}{\partial x^2} = \frac{\partial}{\partial x}\frac{\partial c}{\partial x}$$

which is approximated by

$$\frac{(\frac{\partial c}{\partial x})_{i+1/2} - (\frac{\partial c}{\partial x})_{i-1/2}}{\Delta x} = \frac{\frac{c_{i+1} - c_i}{\Delta x} - \frac{c_i - c_{i-1}}{\Delta x}}{\Delta x}$$

In the case of a variable diffusion coefficient the term with the second derivative in (4.7) should read (see section 2.2):

$$\frac{\partial}{\partial x} K \frac{\partial c}{\partial x}$$

which can be approximated by

$$\frac{(K\frac{\partial c}{\partial x})_{i+1/2} - (K\frac{\partial c}{\partial x})_{i-1/2}}{\Delta x} = \frac{\frac{K_{i+1} + K_i}{2}\frac{c_{i+1} - c_i}{\Delta x} - \frac{K_i + K_{i-1}}{2}\frac{c_i - c_{i-1}}{\Delta x}}{\Delta x}$$

The consistency of equation (4.7) with the partial differential equation (4.6) is verified by substitution of $c(x_i,t)$ into a semi-discrete difference operator given by:

$$D_{\Delta x,dt} = \frac{dc_i}{dt} - L_{\Delta x}c_i \text{, where } L_{\Delta x}c_i = K\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2}$$

Consistency now requires that:

$$\lim_{\Delta x \to 0} D_{\Delta x,dt} c(x_i,t) = 0$$

This means that we require that:

$$\lim_{\Delta x \to 0} \frac{\partial c(x,t)}{\partial t} - K \frac{c(x - \Delta x,t) - 2c(x,t) + c(x + \Delta x,t)}{\Delta x^2} = 0$$

As in chapter 3 we can verify this requirement by Taylor series expansion.

In this way we can also find the order of accuracy of this semi-discretization. This is accomplished by the determination of the local truncation error E_i. We will find that:

$$E_i = O(\Delta x^2)$$

Also the following notation is used:

$$\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2} = \frac{\partial^2 c}{\partial x^2} + O(\Delta x^2)$$

which follows from the following Taylor series expansion:

$$\frac{c - \Delta x c_x + \frac{\Delta x^2}{2} c_{xx} - \frac{\Delta x^3}{3!} c_{xxx} + \frac{\Delta x^4}{4!} c_{xxxx} - \ldots - 2c + c + \Delta x c_x + \frac{\Delta x^2}{2} c_{xx} + \frac{\Delta x^3}{3!} c_{xxx} + \frac{\Delta x^4}{4!} c_{xxxx} + \ldots}{\Delta x^2} =$$

$$= \frac{\partial^2 c}{\partial x^2} + \frac{\Delta x^2}{12} \frac{\partial c^4}{\partial x^4} + \ldots$$

where:

$$c_x = \frac{\partial c}{\partial x}, \quad c_{xx} = \frac{\partial^2 c}{\partial x^2} \quad \text{etc.}$$

We say that the spatial difference operator that replaces the second order derivative is of second order of accuracy. Note that this does not necessarily imply that the local truncation error is of the same order, although this is the case for (4.7). To illustrate this we consider the following spatial discretization of (4.6):

$$\frac{1}{12} \frac{dc_{i-1}}{dt} + \frac{5}{6} \frac{dc_i}{dt} + \frac{1}{12} \frac{dc_{i+1}}{dt} - K \frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2} = 0$$

The local truncation error is in this case of $O(\Delta x^4)$.

Exercise 4.A:

Verify the observation mentioned above.

We now return to (4.7). This is not yet a complete discretization. To complete the discretization process we choose, as an example, the explicit Euler rule for the discretization in space. This yields:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} - K \frac{c_{i-1}^n - 2c_i^n + c_{i+1}^n}{\Delta x^2} = 0 \qquad (4.8)$$

where i=1,...,K-1. We assume that boundary conditions of type (A) are prescribed.

Consistency of (4.8) requires the following

$$\lim_{\substack{\Delta x \to 0 \\ \Delta t \to 0}} D_{\Delta x,\Delta t} c(x_i,t_n) = 0$$

where $D_{\Delta x,\Delta t}$ is now derived from (4.8) in a similar way as $D_{\Delta x,dt}$.

Since the explicit Euler rule is a first order method and since the spatial discretization method is of second order accuracy it is not surprising that the overall local truncation error E^{n+1}_i is $O(\Delta x^2,\Delta t)$ which means that the total discretization is second order in space and first order in time. To verify this one must make use of Taylor series expansions in two independent variables given by:

$$c(x+\Delta x,t+\Delta t) = c(x,t) + \left(\Delta x\frac{\partial}{\delta x} + \Delta t\frac{\partial}{\partial t}\right)c(x,t) + \frac{1}{2!}\left(\Delta x\frac{\partial}{\partial x} + \Delta t\frac{\partial}{\partial t}\right)^2 c(x,t) + \frac{1}{3!}\left(\Delta x\frac{\partial}{\partial x} + \Delta t\frac{\partial}{\partial t}\right)^3 c(x,t) + \ldots$$

or:

$$c(x+\Delta x,t+\Delta t) = c(x,t) + \Delta x c_x + \Delta t c_t + \frac{1}{2!}\left(\Delta x^2 c_{xx} + 2\Delta x\Delta t C_{xt}\Delta t^2 c_{tt}\right) +$$

$$+\frac{1}{3!}\left(\Delta x^3 c_{xxx} + 3\Delta x^2\Delta t c_{xxt} + 3\Delta x\Delta t^2 c_{xtt} + \Delta t^3 c_{ttt}\right) + \ldots$$

Exercise 4.B:

Derive the local truncation error of (4.8)

It is important that when determining the truncation error **all** Taylor expansions are made with respect to one and the same point of reference; which point is chosen is a matter of convenience.

4.2.1. Convergence and stability

Also for PDE's we have to consider the notions of *convergence*, *absolute stability* and *zero stability*.

As for ODE's convergence implies that the global error, or the difference between the numerical solution and the analytical solution of the PDE, tend to zero if the grid parameters Δx and Δt tend to zero.

Absolute stability implies the boundedness of the numerical solution c^n_i for $n\to\infty$ while Δx and Δt are fixed. Zero-stability implies the boundedness of c^n_i for $i\to\infty$, $n\to\infty$ while $\Delta x\to 0$ and $\Delta t\to 0$ while x_i and t_n are fixed.

The principal difference in absolute stability of initial value problems for ODE's and PDE's lies in the fact that now there is also some Δx that tends to zero. This causes the size of the system of difference equations, that is solved at each timestep, to tend to infinity as well. A discussion on this effect of a growing size of the system of equations is given by Richtmyer and Morton (1967). We restrict ourselves to the remark that while for ODE's absolute stability imposes more severe restrictions on the time step than zero-stability this is generally not the case for PDE's. Stability in both ways is necessary for practical applications but the analysis to find the stability conditions might be difficult and in some, non-linear, cases even impossible. For PDE's we restrict ourselves to zero stability, although A-stable methods will probably be absolutely stable as well when applied to systems of ODE's resulting from semi-discretization of ODE's. Moreover in some cases, mostly purely symmetric, absolute stability and zero-stability impose the same stability conditions. Within the context of PDE's by stability we will mean zero-stability unless specified otherwise.

The relation between convergence and stability is described by *Lax's equivalence theorem*:

> **Given a properly posed initial-value problem and a finite difference approximation to it that satisfies the consistency condition, zero-stability is the necessary and sufficient condition for convergence.**

Stability can be separated into the influence on the stability of the internal scheme and of the boundary conditions including the boundary schemes. For a discussion on these topics see e.g. Godunov and Ryabenkii (1964), Richtmyer and Morton (1967) or Hirsch (1991). We restrict ourselves to the stability of the internal scheme, i.e the finite difference scheme that is applied as much as possible in the grid domain of the problem. In this way we obtain only necessary conditions for stability since our analysis is incomplete. This implies that in practice we should not be surprised of schemes becoming unstable despite of some stability analysis that we had performed.

To study only the internal scheme it is sufficient to disregard boundary conditions and to consider a spatial domain given by (-∞,∞). We will also assume that the initial conditions are periodic with some periodic length L. This allows us to denote this initial condition as a Fourier series given by:

$$c(x,0) = \sum_{j=-\infty}^{\infty} \hat{c}_j \; e^{i\frac{2\pi}{L_j}x}$$

where L_j=L/j.

To study the stability of (4.8) we substitute this expression into (4.8) and we verify if each fourier mode is bounded if n→∞. For this it is sufficient to substitute only one mode into (4.8). After dropping the index j we obtain:

$$\hat{c}^{n+1} e^{i\frac{2\pi}{L}x} - \hat{c}^n e^{i\frac{2\pi}{L}x} - q\left(\hat{c}^n e^{i\frac{2\pi}{L}(x-\Delta x)} - 2\hat{c}^n e^{i\frac{2\pi}{L}x} + \hat{c}^n e^{i\frac{2\pi}{L}(x+\Delta x)}\right) = 0$$

where $q = K\dfrac{\Delta t}{\Delta x^2}$

After some manipulation, including cancelling common factors, we obtain:

$$\hat{c}^{n+1} = [1 - 2q(1 - \cos\frac{2\pi\Delta x}{L})]\hat{c}^n \qquad (4.9)$$

L/Δx is the number of points per wave length, its possible minimum value is 2 while there is no maximum. From this it follows that 0≤ξ≤π, where ξ=2πΔx/L. We rewrite (4.9) as:

$$\hat{c}^{n+1} = r\hat{c}^n$$

where:

$$r = 1 - 2q(1 - \cos\xi)$$

Here r is the *amplification factor* of the numerical method. For stability we must have that $|r| \le 1$, this gives:

$$K\frac{\Delta t}{\Delta x^2} \le \frac{1}{2}$$

This kind of stability analysis is usually referred to as the verification of the *Von Neumann condition*.

The discretization in time of (4.7) can also be done by the Trapezoidal rule, yielding:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} - \frac{1}{2}K\frac{c_{i-1}^{n+1} - 2c_i^{n+1} + c_{i+1}^{n+1}}{\Delta x^2} - \frac{1}{2}K\frac{c_{i-1}^{n} - 2c_i^{n} + c_{i+1}^{n}}{\Delta x^2} = 0 \qquad (4.10)$$

This finite difference method is known as the *Crank Nicolson scheme*.

The solution of this set of finite difference equations implies the solution of a tridiagonal set of equations. This solution procedure is described in section 4.2.2.

Exercise 4.C:

(i) *Compute the local truncation error of the Crank Nicolson method and*
(ii) *show by means of the Von Neumann condition its unconditional stability.*

4.2.2. Solving finite difference equations, the *Thomas Algorithm*

After spatial-discretization and discretization in time we have obtained a system of finite-difference equations with which we can compute numerical approximations. To obtain these approximations we have to solve the finite difference equations. For explicit methods this is straightforward. We can rewrite (4.8) as:

$$c_i^{n+1} = c_i^n + \frac{K\Delta t}{\Delta x^2}(c_{i-1}^n - 2c_i^n + c_{i+1}^n),\ j = 1,2,...,K-1$$

At the boundaries we simply prescribe:

$$c_0^n = C_0(n\Delta t), \quad c_K^n = C_k(n\Delta t)$$

where $C_0(t)$ and $C_K(t)$ are functions prescribed as Dirichlet type of boundary conditions. At t=0 all values for c_n are given as initial conditions.

For an implicit set of finite difference equations such as the Crank Nicolson scheme given by (4.10) the solution of the set of equations is less straightforward.

In fact (4.10) constitutes a matrix equation given by:

$$\begin{bmatrix} 1+q & -\frac{q}{2} & & & \\ -\frac{q}{2} & 1+q & -\frac{q}{2} & & \\ & \ddots & \ddots & \ddots & \\ & & -\frac{q}{2} & 1+q & -\frac{q}{2} \\ & & & -\frac{q}{2} & 1+q \end{bmatrix} \cdot \begin{bmatrix} c_1^{n+1} \\ c_2^{n+1} \\ \vdots \\ c_{K-2}^{n+1} \\ c_{K-1}^{n+1} \end{bmatrix} = \begin{bmatrix} c_1^n + \frac{q}{2}\left(C_0(n\Delta t) - 2c_1^n + c_2^n\right) \\ c_2^n + \frac{q}{2}\left(c_1^n - 2c_2^n + c_3^n\right) \\ \vdots \\ c_{K-2} + \frac{q}{2}\left(c_{K-3} - 2c_{K-2}^n + c_{K-1}^n\right) \\ c_{K-1} + \frac{q}{2}\left(c_{K-2} - 2cK - 1 + C_K(n\Delta t)\right) \end{bmatrix}$$

The matrix of this system is *tridiagonal*, this means that all elements $a_{i,j}$ of a MxM matrix A are zero whenever $|i-j|>1$.

For tridiagonal systems of this kind, which we denote as:

$$\begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{M-1} & b_{M-1} & c_{M-1} \\ & & & a_M & b_M \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_{M-1} \\ X_M \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{M-1} \\ d_M \end{bmatrix} \qquad (4.11)$$

there exists an efficient algorithm to solve the equations. This algorithm is known as the *Thomas algorithm* or *double sweep method*. It consists of two steps. First (4.11) is manipulated into the form:

$$\begin{bmatrix} 1 & c_1' & & & \\ & 1 & c_2' & & \\ & \ddots & \ddots & \ddots & \\ & & & 1 & c_{M-1}' \\ & & & & 1 \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_{M-1} \\ X_M \end{bmatrix} = \begin{bmatrix} d_1' \\ d_2' \\ \vdots \\ d_{M-1}' \\ d_M' \end{bmatrix}$$

where:

$$c'_1 = \frac{c_1}{b_1} \quad , \quad d'_1 = \frac{d_1}{b_1}$$

$$c'_i = \frac{c_i}{b_i - a_i c'_{i-1}} \quad , \quad d'_i = \frac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}} \quad , \quad i = 2,\ldots,M$$

This step is called the forward sweep. The second step, the backward sweep, consists of a back-substitution:

$$X_M = d'_M$$
$$X_i = d'_i - X_{i+1} c'_i \, , \quad i = M-1, M-2, \ldots, 1$$

This algorithm is economical, especially on PC and workstation type of computers. It requires only 5M-4 operations (multiplications and divisions). To prevent growth of round-off errors, see Golub and Van Loan (1983), a sufficient condition is $|b_i| > |a_i| + |c_i|$.

The algorithm is highly recursive. This implies that this algorithm is not economical on vector- and/or parallel computers. For this type of computers other algorithms, such as *cyclic reduction*, see e.g. Golub and Van Loan (1983) or Van der Vorst (1988), are more appropriate.

4.2.3. Accuracy, relaxation factor for the diffusion equation

The local truncation error gives insight in the accuracy of the approximation only in a qualitative way. It is possible to show that the global error $e_n=c(t_n)-c_n$ is of the same order as the local truncation error. A proof of this is considered beyond the scope of these lecture notes, see e.g. Gustafson (1975). This however gives only insight into the asymptotic behaviour, $\Delta t \to 0$ and $\Delta x \to 0$, of the global error. For a second order method it means for example that if Δx and Δt are divided by 2 then the global error will change by a factor of 4. In other words this gives no information on the actual global error for given values of the numerical parameters Δx and Δt. To get some feeling for this aspect we treat the global error of a simple test problem. This simple test problem is the same as the test equation which is used for the Von Neumann stability analysis. This test problem, that consists of a pure initial value problem, is given by:

$$\frac{\partial c}{\partial t} - K \frac{\partial^2 c}{\partial x^2} = 0, \quad -\infty < x < \infty \tag{4.11'}$$

The initial conditions of (4.11') are given by:

$$c(x,0) = e^{i\frac{2\pi}{L}x}$$

If we substitute $c(x,t) = \hat{c}(t) e^{i\frac{2\pi}{L}x}$ into (4.11') we obtain the following ODE:

$$\frac{d\hat{c}}{dt} + K\left(\frac{2\pi}{L}\right)^2 \hat{c} = 0$$

The initial condition is given by $\hat{c}(0) = 1$. The solution of this equation is:

$$\hat{c}(t) = e^{-K\left(\frac{2\pi}{L}\right)^2 t}$$

We can now define the relaxation time T by:

$$T = K^{-1}\left(\frac{2\pi}{L}\right)^{-2}$$

T is the time in which the initial amplitude is decreased by an order of magnitude given by e^{-1}.

Subsequently we can consider a semi-discrete system of ODE's given by:

$$\frac{dc_j}{dt} - K\frac{c_{j-1} - 2c_j + c_{j+1}}{\Delta x^2} = 0 \tag{4.12}$$

with initial conditions given by:

$$c_j = e^{i\frac{2\pi}{L}j\Delta x}$$

If we substitute $c_j(t) = \tilde{c}(t)e^{i\frac{2\pi}{L}j\Delta x}$ into (4.12) then we obtain after some manipulation:

$$\frac{d\tilde{c}(t)}{dt} + K\frac{2 - 2\cos\xi}{\Delta x^2}\tilde{c} = 0$$

where $\xi=2\pi/n$, $n=L/\Delta x$ which is the number of points per wavelength.

This equation is characterized by a relaxation time $T' = \frac{\Delta x^2}{K(2 - 2\cos\xi)}$.

At this point we express the global error in terms of the *relaxation factor*, a factor that we define as: $R'=T/T'$. This means that if R'>1 then the dissipation rate of the spatial discretization is larger than the true dissipation rate while if R'<1 then the dissipation rate of the spatial discretization is less than the exact one. R' can be denoted as:

$$R' = \frac{K\frac{2 - 2\cos\xi}{\Delta x^2}}{K\frac{4\pi^2}{L^2}} = \frac{2 - 2\cos\xi}{\xi^2}$$

In figure 4.1 this factor is visualized. It is to be noted that by Taylor's series expansions a good estimate of this factor can be obtained. For this case this yields:

$$R' \approx 1 - \frac{\xi^2}{12}$$

By this simple formula rough estimates for the maximum allowable grid size can be made.

In practice the semi-discrete equations are never solved exactly. The discretization in time will also influence the results. This can be taken into account by the assumption that full discretizations of our test problem, for example by (4.8) or (4.10) can be denoted as:

$$\tilde{c}^{n+1} = r(q,\xi)\tilde{c}^n$$

where $q=K\Delta t/(\Delta x^2)$.

For example the explicit Euler rule for the discretization in time of our test problem yields:

$$r(q,\xi)=1-2q(1-\cos\xi)$$

Exercise 4.D:

Verify the equation given above.

We assume that the numerical relaxation time T' is given by:

$$\frac{\Delta t}{T'} = -\log\left(r(q,\xi)\right)$$

The relative relaxation factor can now be defined as:

$$\frac{T}{T'} = \frac{\Delta t/T'}{\Delta t/T} = \frac{-\log\left(r(q,\xi)\right)}{K\Delta t\dfrac{4\pi^2}{L^2}} = -\frac{\log\left(r(q,\xi)\right)}{q\xi^2}$$

In figure 4.1 this factor is plotted for various values of q and for various discretizations in time. For this figure q=0 means the semi-discretization as such.

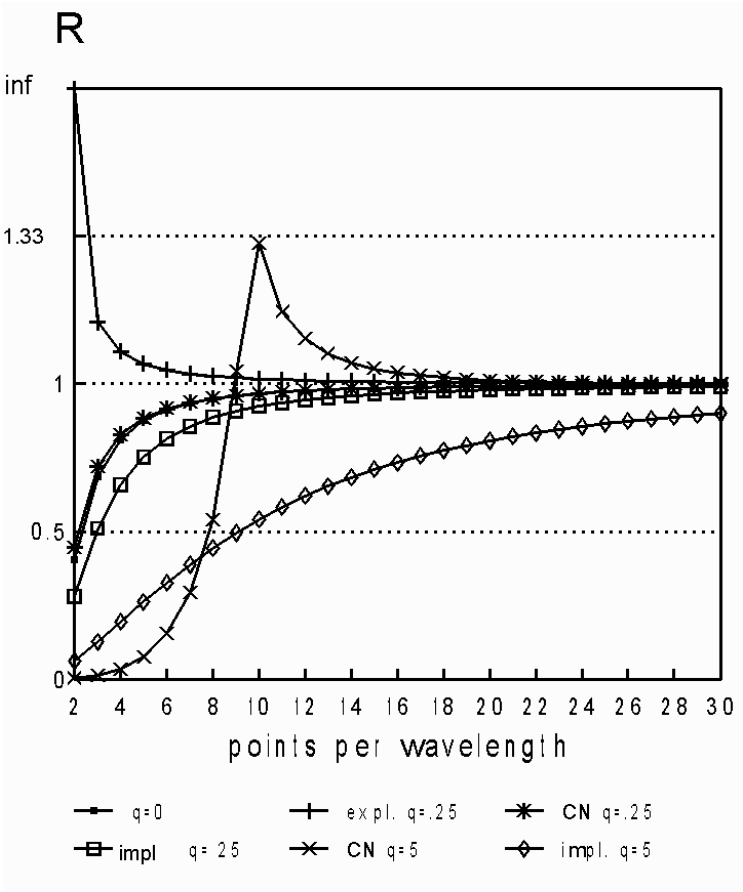


Figure 4.1. Relaxation factor as function of the number of points per wavelength for various numerical schemes. "CN" is the Crank-Nicholson scheme.
Note that the scale is nonlinear for values of R'>1; the transformation is according to Y=2-2/R'; it is seen that for one case R' goes to infinity.

4.2.4. Summary of numerical approximations for the simplest diffusion equation

In this section an overview is given of combinations of the linear multistep methods that we dealt with in chapter 3 and the semi-discretization given by:

$$\frac{dc_i}{dt} - K\frac{c_{i-1} - 2c_i + c_{i+1}}{\Delta x^2} = 0$$

Before we give the overview we firstly explore the *stiffness* of this set of equations.

If we denote our equation as dc/dt=Ac then A denotes a (M-1)x(M-1) tridiagonal matrix given by:

$$A = \frac{K}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{bmatrix}$$

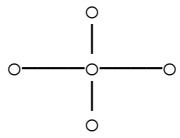
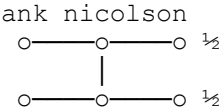
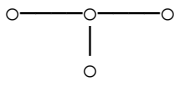
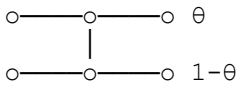
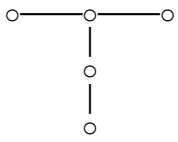
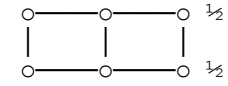
The eigenvalues of this matrix are:

$$\lambda_j = \frac{K}{\Delta x^2}\left(-2 + 2\cos\frac{j\pi}{M} \right),\ j = 1,2,...,M-1$$

This implies that the eigenvalues lie within the interval $(-4K/\Delta x^2, 0)$ of the real line. For relative small values of Δx this implies a large distance between the eigenvalues, in other words the problem is sometimes very stiff, which should lead to application of a stiffly stable method.

The overview is given by table 4.1.

Table 4.1

Explicit euler 	$\frac{c^{n+1}-c^n}{\Delta t}=Ac^n$ E=O(Δt)+O(Δx^2) Stable if q≤ 1/2, maximum accuracy if q=1/6
Midpoint 	$\frac{c^{n+1}-c^{n-1}}{2\Delta t}=Ac^n$ Always unstable, useless
Crank nicolson  ½ ½	$\frac{c^{n+1}-c^n}{\Delta t}=\tfrac{1}{2}Ac^{n+1}+\tfrac{1}{2}Ac^n$ E=O(Δt^2)+O(Δx^2) Always stable, but not stiffly stable for large values of q.
Implicit euler 	$\frac{c^{n+1}-c^n}{\Delta t}=Ac^{n+1}$ E=O(Δt)+O(Δx^2) Always stable and also stiffly stable
Θ method  θ 1-θ	$\frac{c^{n+1}-c^n}{\Delta t}=\theta Ac^{n+1}+(1-\theta)Ac^n$ E=O(Δt)+O(Δx^2) Implicit, unconditionally stable if θ>½, optimal values for θ are possible for large values of q.
Backward differentiation 	$\frac{3c^{n+1}-4c^n+c^{n-1}}{\Delta t}=Ac^{n+1}$ E=O(Δt^2)+O(Δx^2) Implicit, always stable and also stiffly stable, preferable for large values of q
M matrix method  ½ ½ 1/12 5/6 1/12	$M\frac{c^{n+1}-c^n}{\Delta t}=\tfrac{1}{2}Ac^{n+1}+\tfrac{1}{2}Ac^n, Mc_j=\frac{1}{12}c_{j-1}+\frac{5}{6}c_j+\frac{1}{12}c_{j+1}$ E=O(Δt^2)+O(Δx^4) Implicit, very accurate, like Finite Element Method, but solutions are not always free from spurious oscillations.

Exercise 4.E:

Verify (a) the truncation errors, (b) the stability conditions and (c) the local truncation error for q=1/6 for the explicit Euler rule, all for the numerical schemes in the table given above.

4.3. The discretization of the simplest convection equation

Transport of matter in estuaries is not only a result of transport that is modelled by diffusion but also a result of transport that is modelled by *convection*. The simplest convection equation is given by:

$$\frac{\partial c}{\partial t}+v\frac{\partial c}{\partial x}=0 \tag{4.13}$$

where v is the known propagation velocity and c is a passive scalar, e.g. salt concentration. Equation (4.13) is hyperbolic in character.

This equation has to be completed with initial conditions and with one boundary condition at the inflow boundary of the domain [0,X]. If we assume that v>o then this condition has to be prescribed at x=0.

If v is constant then a general solution of (4.13) can be written as:

$$c(x,t) = \begin{cases} C^0(x - vt), & x \geq vt \\ \\ C_0(t - x/v), & x < vt \end{cases} \tag{4.14}$$

where the boundary condition and the initial condition are given by:

$$c(x,0)=C^0(x),\ x\geq 0$$

$$c(0,t)=C_0(t),\ t>0$$

The solution is constant along lines in the x,t space with a slope 1/v. These lines are called the *characteristics* for (4.13).

This equation is characterized by absence of dissipation and by travelling of waves or disturbances at a speed v.

4.3.1. Examples of discretizations, the Courant condition

Again, following the method of lines, we can consider firstly a spatial discretization given by:

$$\frac{dc_j}{dt} = -v\frac{c_{j+1} - c_{j-1}}{2\Delta x} \tag{4.15}$$

By Taylor series expansions we can verify that $E=O(\Delta x^2)$ for this spatial approximation. This spatial discretization is called *second order central differencing*.

If we combine (4.15) with the explicit Euler rule we obtain:

$$\frac{c_j^{n+1} - c_j^n}{\Delta t} = -v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x}$$

If we calculate the amplification factor r, to verify the Von Neumann condition for stability, then we obtain:

$$r(\sigma,\xi) = 1 - i\sigma\sin\xi,\ \sigma = v\frac{\Delta t}{\Delta x},\ \xi = \frac{2\pi\Delta x}{L}$$

Note that σ is generally referred to as the *Courant number*.

Exercise 4.F:

Verify the calculation of r(σ,ξ).

For stability we must have that $|r| \leq 1$. It is easy to see however that $|r| = 1 + O(\Delta t)$, from which it follows that this scheme is never stable.

If we combine (4.15) with the Midpoint rule then we obtain:

$$\frac{c_j^{n+1} - c_j^{n-1}}{2\Delta t} = -v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x} \tag{4.16}$$

Verification of the Von Neumann condition yields:

$$r_{1,2}(\sigma,\xi) = -i\sigma\sin\xi \pm \sqrt{1 - \sigma^2\sin^2\xi}$$

For stability we must have that $|\sigma| \leq 1$.

Exercise 4.G:

Verify this condition, explain that the amplification factor consists of two characteristic roots r_1 and r_2. Show that both roots have an absolute value of 1 for all values of ξ, as long as $|\sigma| \leq 1$.

In this case this condition coincides with the *Courant Friedrichs Lewy condition or CFL condition*, after Courant, Friedrichs and Lewy (1928) (CFL). The Courant condition states that a necessary condition for a numerical scheme to be convergent with (4.13) is that the characteristic of (4.13) lies within the so called *numerical domain of dependence*. This is the region enclosed by the grid points that are part of the finite difference approximation, see figure 4.2.

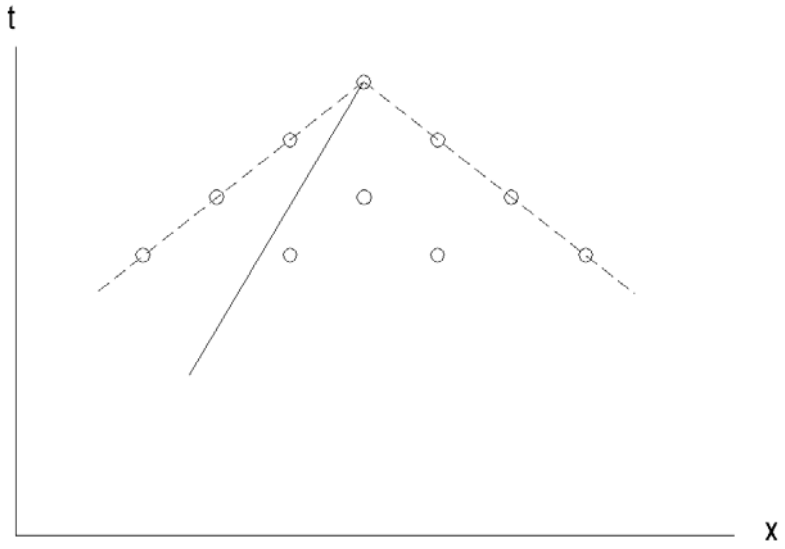


Figure 4.2. Numerical domain of dependence for the leap-frog scheme (bounded by dashed lines) and a characteristic (drawn line).

Note that (4.16) constitutes two sets of independent equations. One set is defined at the points (2j,2n) and (2j+1,2n+1) while the other set consists of the (2j+1,2n) and (2j,2n+1). By cancelling one of the two sets we gain efficiency by a factor of two without losing accuracy. The resulting scheme is the *Leap-frog* method.

Of course (4.15) can be combined also with implicit discretizations in time. And, as can be expected, the stability is unconditional.

Exercise 4.H:

Verify this for the backward differentiation method. Explain the unconditional stability also in terms of the CFL condition.

Another spatial discretization which is often used is called *first order upwinding*.

It is given by:

$$\frac{dc_j}{dt} = \begin{cases} -v\dfrac{c_j - c_{j-1}}{\Delta x}, & \text{if } v \geq 0 \\ -v\dfrac{c_{j+1} - c_j}{\Delta x}, & \text{if } v < 0 \end{cases} \tag{4.17}$$

(4.17) is often combined with the explicit Euler rule. The reason why this combination is often applied will be explained later on.

The discretizations (4.15) and (4.17) are by far not the only spatial discretization that are applied. We summarize some of them in the following table:

Table 4.2

Second order central differencing	$\frac{dc_j}{dt} = - v \frac{c_{j+1} - c_{j-1}}{2\Delta x}$
Box method (second order)	$\frac{1}{2}\frac{dc_{j+1}}{dt} + \frac{1}{2}\frac{dc_j}{dt} = - v \frac{c_{j+1} - c_j}{\Delta x}$
Fourth order M(ass) matrix method	$\frac{1}{6}\frac{dc_{j-1}}{dt} + \frac{4}{6}\frac{dc_j}{dt} + \frac{1}{6}\frac{dc_{j+1}}{dt} = - v \frac{c_{j+1} - c_{j-1}}{2\Delta x}$
First order upwind differencing	$\frac{dc_j}{dt} = \begin{cases} -v\frac{c_j - c_{j-1}}{\Delta x}, & \text{if } v \geq 0 \\ -v\frac{c_{j+1} - c_j}{\Delta x}, & \text{if } v < 0 \end{cases}$
Second order upwind differencing	$\frac{dc_j}{dt} = \begin{cases} -v\frac{3c_j - 4c_{j-1} + c_{j-2}}{2\Delta x}, & \text{if } v \geq 0 \\ -v\frac{-3c_j + 4c_{j+1} - c_{j+2}}{2\Delta x}, & \text{if } v < 0 \end{cases}$
Third order upwind differencing	$\frac{dc_j}{dt} = \begin{cases} -v\frac{2c_{j+1} + 3c_j - 6c_{j-1} + c_{j-2}}{6\Delta x}, & \text{if } v \geq 0 \\ -v\frac{-2c_{j-1} - 3c_j + 6c_{j+1} - c_{j+2}}{6\Delta x}, & \text{if } v < 0 \end{cases}$

Exercise 4.I:

Calculate the truncation errors of the methods given above.

Note that semi-discretizations based upon central differences can not be used at the boundary x=X of the spatial domain because such a discretization needs than a value for c that is outside this domain. To circumvent this problem at such a boundary location one has to use an upwind method. The box scheme does not have this problem. At x=0 this problem does not exist because here one has to prescribe a boundary condition.

The second and first order upwind schemes can be combined with implicit integration methods in time. In that case the systems of equations can not be solved with the Thomas algorithm. A similar algorithm, which is called the *generalized Thomas algorithm*, see Fletcher (1988) has to be applied in that case.

For the discretization in time one can use the linear multi step methods described in chapter 3 that were used already also for the diffusion equation. At this point it is to be noted that the method of lines is certainly not the only way by which the transport equation can be discretized. The method of lines is using only a minimum amount of information of the equations. Accurate methods can be constructed by taking into account the mathematical and the underlying physical properties of the transport equation, see e.g. Hirsch (1991), Fletcher (1988) or Van Stijn et al.(1987).

A method for the integration in time that exploits only a little bit more information from the equation that is to be approximated is the *Lax Wendroff* method. To describe this method we again consider the combination of Euler's explicit rule and second order central differencing. A combination that is never stable. The local truncation error of Euler's rule is given by:

$$E^n = -\frac{1}{2}\Delta t \frac{\partial^2 c}{\partial t^2}$$

From (4.13) one can derive the following identity:

$$\frac{\partial^2 c}{\partial t^2} = v^2 \frac{\partial^2 c}{\partial x^2}$$

Now we use the centered difference expression for $\partial c/\partial x^2$ to get the following finite difference approximation of (4.13):

$$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x} = \frac{1}{2}\Delta t .v^2 \frac{c_{j-1}^n - 2c_j^n + c_{j+1}^n}{\Delta x^2} \qquad (4.18)$$

Exercise 4.J:

(i) Show that the local truncation error of the Lax-Wendroff scheme is $O(\Delta x^2, \Delta t^2)$,
(ii) show that the Von Neumann condition is fulfilled if $|\sigma| \leq 1$. Hint: rewrite 4.18 as:

$$c_j^{n+1} = c_j^n - \frac{\sigma}{2}\left(c_{j+1}^n - c_{j-1}^n\right) + \frac{\sigma^2}{2}\left(c_{j-1}^n - 2c_j^n + c_{j+1}^n\right)$$

Similar to the Lax Wendroff scheme one can construct the *QUICKEST* scheme, see e.g Abbott (1989) and Fletcher (1988). This scheme is based upon a combination of third order upwind differencing and the explicit Euler rule. To obtain also third order accuracy in time the first order and second order term of the local truncation error of the explicit Euler rule are corrected where now also the following identity is used:

$$\frac{\partial^3 c}{\partial t^3} = -v^3 \frac{\partial^3 c}{\partial x^3}$$

Under the assumption that v≥0 this leads to the following approximation:

$$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{2c_{j+1}^n + 3c_j^n - 6c_{j-1}^n + c_{j-2}^n}{6\Delta x} = \frac{1}{2}\Delta t v^2 \frac{c_{j-1}^n - 2c_j^n + c_{j+1}^n}{\Delta x^2} - \frac{1}{6}\Delta t^2 v^3 \frac{(c_{j-1}^n - 2c_j^n + c_{j+1}^n) - (c_{j-2}^n - 2c_{j-1}^n + c_j^n)}{\Delta x^3}$$

Exercise 4.K:

(i) Show the third order of accuracy of this QUICKEST scheme,
(ii) give the formula of this scheme for the case v<0.

4.3.2. Propagation properties

As for the diffusion methods we analyze the accuracy of the numerical methods for a simple test problem. This test-problem is similar to the one we used for the diffusion equation. This means that this test-problem is the same as the purely initial value problem that is used for the Von Neumann stability analysis. This test problem, that consists of a purely initial value problem, is given by:

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} = 0, \quad -\infty < x < \infty \qquad (4.19)$$

The initial conditions of (4.19) are given by:

$$c(x,o) = e^{i\frac{2\pi}{L}x}$$

If we substitute $c(x,t) = \hat{c}(t)e^{i\frac{2\pi}{L}x}$ into (4.19) we obtain the following ODE:

$$\frac{d\hat{c}}{dt} + v\hat{D}\hat{c} = 0$$

where

$$\hat{D} = i\frac{2\pi}{L}$$

The initial condition is given by $\hat{c}(0) = 1$. The solution of this equation is:

$$\hat{c}(t) = e^{-iv\frac{2\pi}{L}t}$$

The solution of (4.19) is now given by:

$$c(x,t) = e^{i\frac{2\pi}{L}(x-v\,t)}$$

This solution is compliant with (4.14). It can be considered as a travelling sine wave. It travels at a speed v.

Next we consider a system of ODE's resulting from semi-discretization of (4.19) by second-order central differences given by:

$$\frac{dc_j}{dt} + v\frac{c_{j+1} - c_{j-1}}{2\Delta x} = 0 \tag{4.20}$$

and with initial conditions given by:

$$c_j(0) = e^{i\frac{2\pi}{L}j\Delta x}$$

If we substitute $c_j(t) = \tilde{c}(t)e^{i\frac{2\pi}{L}j\Delta x}$ into this equation we obtain the following ODE:

$$\frac{d\tilde{c}}{dt} + v\tilde{D}\tilde{c} = 0$$

where:

$$\tilde{D} = i\frac{\sin(\xi)}{\Delta x},\ \xi = \frac{2\pi\Delta x}{L}$$

The solution of this equation is given by:

$$\tilde{c}(t) = e^{-iv\frac{\sin(\xi)}{\Delta x}t}$$

The "analytical solution" of (4.20) now becomes:

$$c_j(t) = e^{i\frac{2\pi}{L}\left(j\Delta x - v\frac{\sin(\xi)}{\xi}t\right)}$$

This solution can be considered as a sine wave travelling at a speed:

$$\tilde{v} = v\frac{\sin(\xi)}{\xi}$$

At this point we can define the *relative wave speed* as the ratio $\tilde{v}/v$ which is equal to:

$$\frac{\sin(\xi)}{\xi}$$

It follows from this expression that second order central differencing causes the numerical wave to be lagging when compared to the analytical wave. In other words central differences are inducing a *lagging phase error*.

It is to be noted that for the ratio that defines the relative wave speed the following relation holds:

$$\frac{\tilde{v}}{v} = \frac{Im\,\tilde{D}}{Im\,\hat{D}}$$

The latter expression is the one that we use for the computation of the relative wave speed in general. Figure 4.3 shows relative wave speeds for various spatial discretizations.

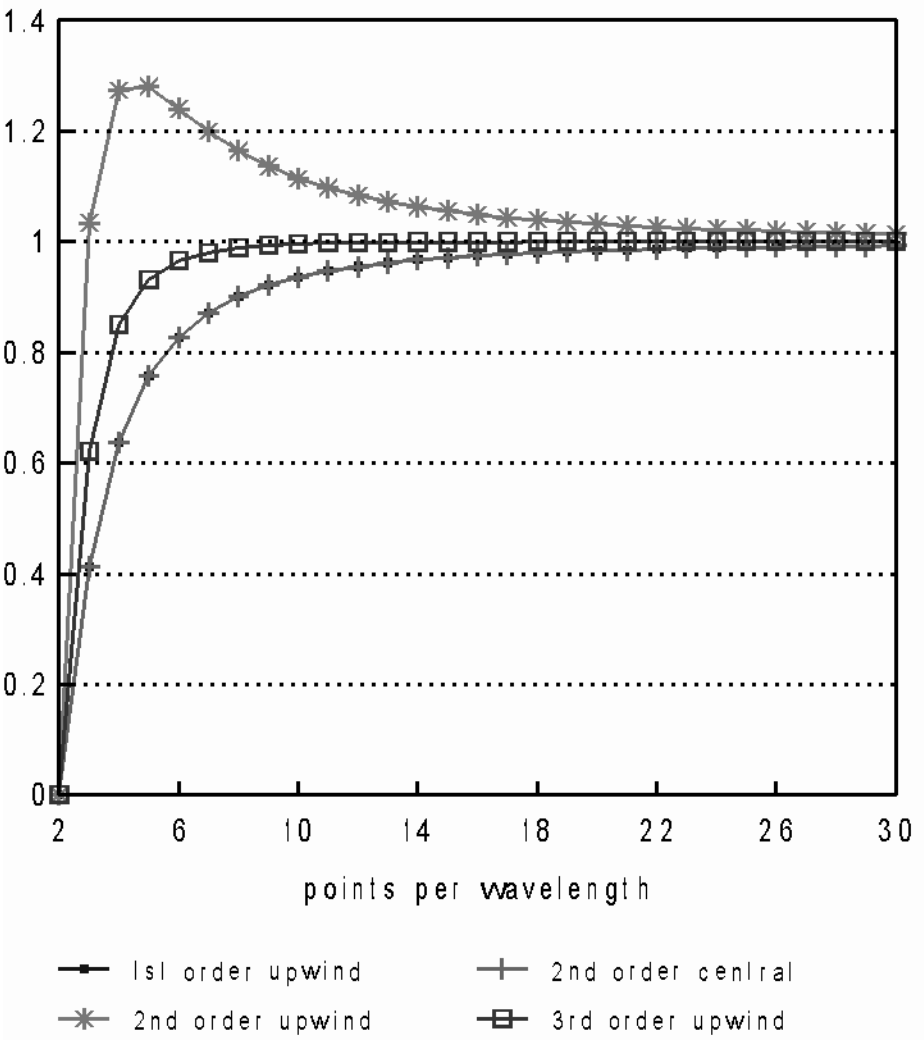


Figure 4.3. Relative wave propagation velocity as function of number of points per wavelength for various semi-discretizations

The quantity $\tilde{D}$ is not necessarily purely imaginary. If we compute for example this quantity for first order upwind differencing then we obtain:

$$\tilde{D} = \frac{1 - \cos\xi + i\sin\xi}{\Delta x}$$

For this case the numerical solution becomes:

$$c_j(t) = e^{v\frac{-1+\cos(\xi)}{\Delta x}t} e^{i\frac{2\pi}{L}\left(j\Delta x - v\frac{\sin(\xi)}{\xi}t\right)}$$

This solution still represents a travelling wave, but the amplitude of this wave is decreasing as time proceeds. Numerical approximations that show this behaviour are called *dissipative*. The *amplitude factor* is defined as the relative decay of the amplitude of a wave after travelling for a period T=L/v. For our example this factor becomes:

$$e^{n_p(-1+\cos\xi)}$$

where n_p denotes the number of points per wave length; $n_p=2\pi/\xi$.

In figure 4.4 we show the amplitude factors for various spatial discretizations.

So far we did not take into account the influence of the discretization in time. To do this we consider integration in time by a one step method that, after Fourier transform as in the semi-discrete case, allows the following notation:

$$\tilde{c}^n = [r(\sigma,\xi)]^n \tilde{c}^0$$

The complex quantity $\tilde{D}$, that we need for the computation of the relative wave speed is now determined by the following relation:

$$\Delta tv\tilde{D} = -\log\left(r(\sigma,\xi)\right)$$

Again it is possible to obtain a relation, entirely with dimensionless parameters, for the relative wave speed. The relative wave speed is now given by:

$$\frac{Im\,\tilde{D}}{Im\,\hat{D}} = \frac{Im\,\Delta tv\tilde{D}}{\Delta tv Im\,\hat{D}} = \frac{Im\left[-\ln\left(r(\sigma,\xi)\right)\right]}{\Delta tv\frac{2\pi}{L}} = \frac{Im\left[-\ln\left(r(\sigma,\xi)\right)\right]}{\sigma\xi}$$

For various combinations of spatial discretizations and discretizations in time the relative wave speed is given in figure 4.5.

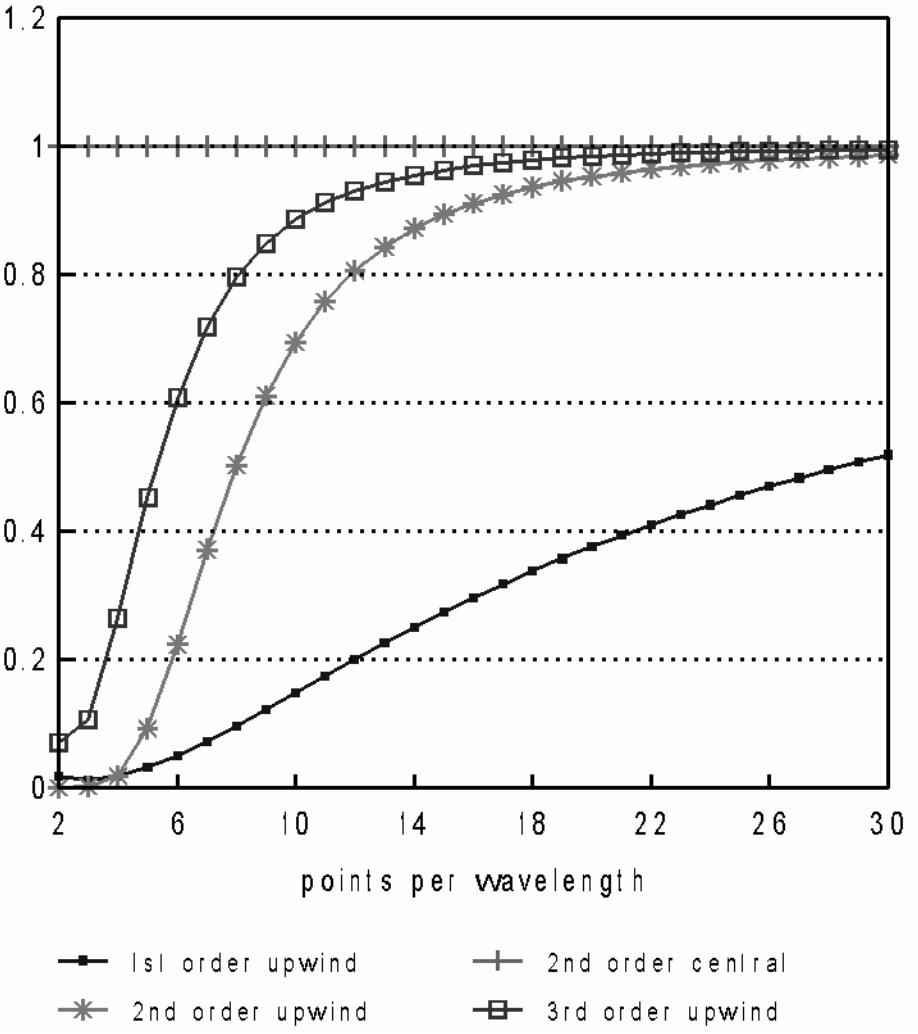


Figure 4.4. Amplitude factor per wave period as function of points per wavelength for various spatial discretizations

The amplitude factor is equal to the amplitude of r^n, where n is chosen such that the total time that is simulated is equal to L/v. This implies $n=L/(v\Delta t)$. Or $n=n_p/\sigma$. For various combinations of spatial discretizations and discretizations in time the amplitude factor is given in figure 4.6.

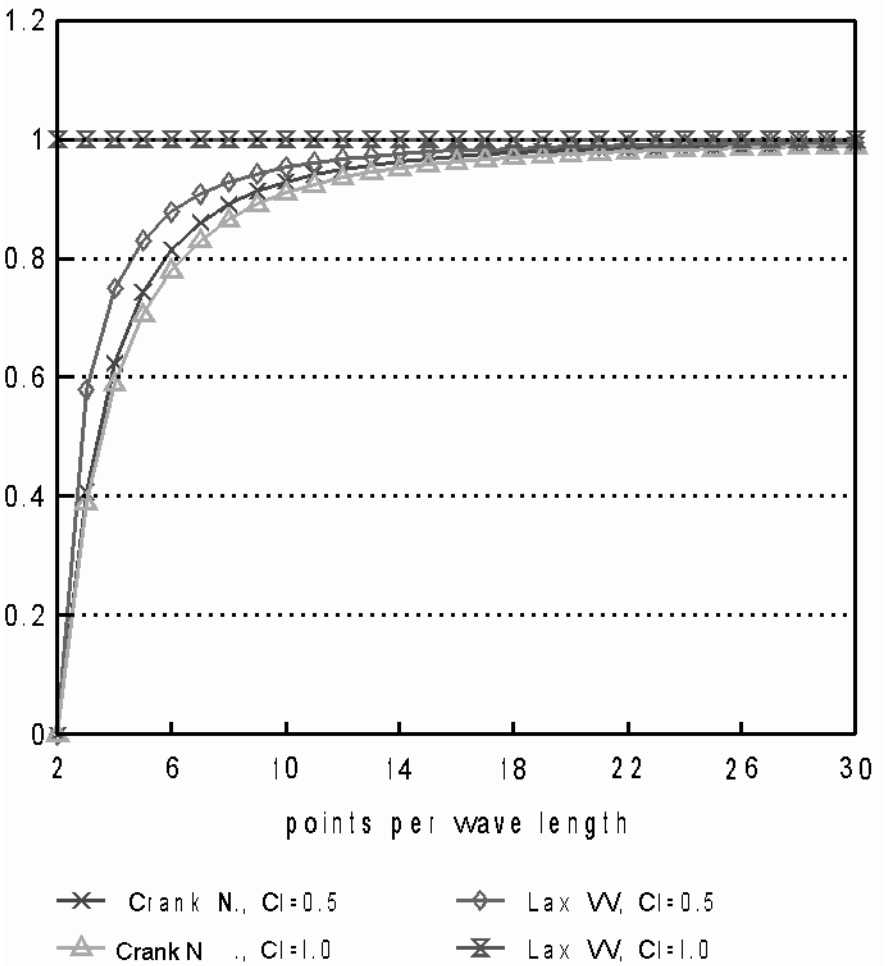


Figure 4.5. Relative wave speed as function of points per wavelength for various combinations of spatial discretizations and discretizations in time. Crank-Nicolson and Lax-Wendroff schemes are shown, each for two values of the Courant number (indicated as Cf in the figure).

It is to be noted that some schemes have no errors at all for the case $\sigma=1$. In this case these methods integrate exactly along the characteristic. This reduces these schemes to the relation $c_j^{n+1}=c_{j-1}^{n}$. This propery is called *point to point transfer*, see e.g. Roache (1976). This property only holds for the simple linear test case.

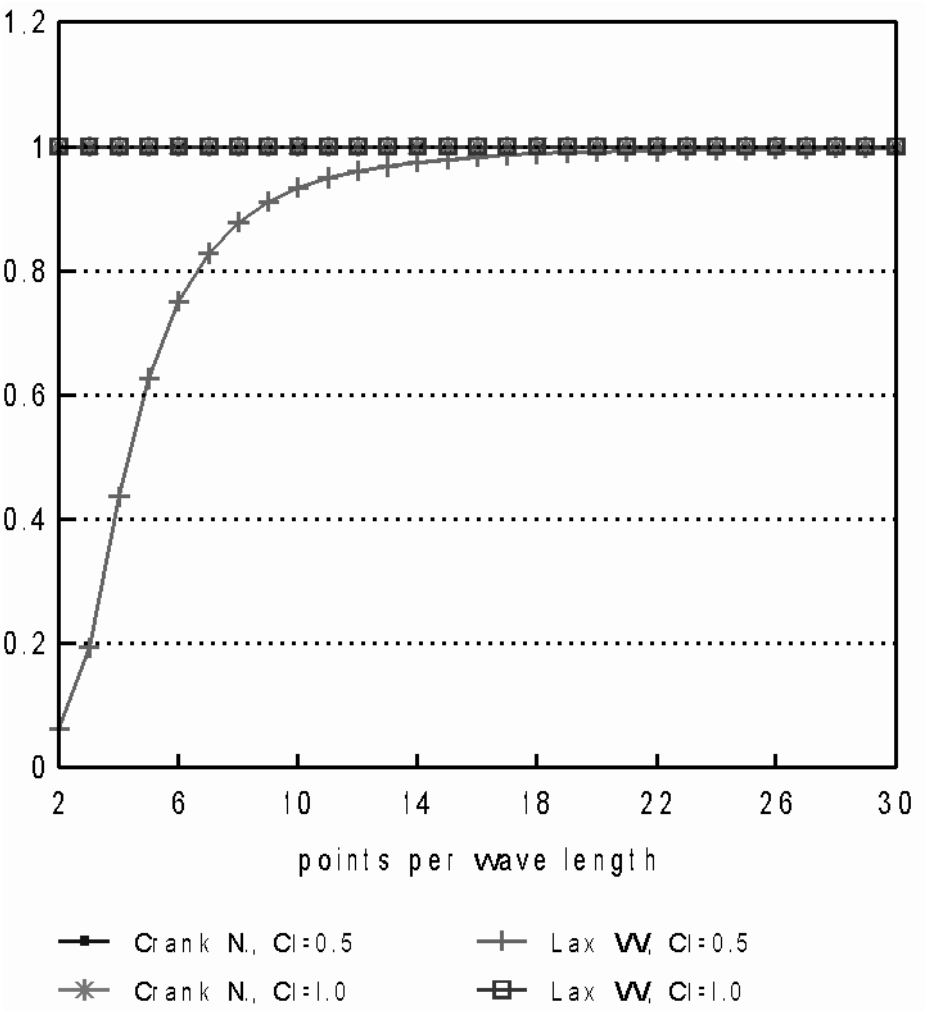


Figure 4.6. The amplitude factor as function of points per wavelength for various combinations of spatial discretizations and discretizations in time. Crank-Nicolson and Lax-Wendroff schemes are shown, each for two values of the Courant number (indicated as Cf in the figure).

4.3.3. Modified equation approach

Numerical dissipation and phase errors depending on the wavenumber or, as it sometimes called, numerical dispersion can be associated with the local truncation error. For this goal the truncation error must be interpreted in a special way.

It is assumed that D(c)=0 represents the differential equation and $D_{\Delta x,\Delta t}(c_j^n)=0$ represents the finite difference equation. The local truncation error E_j^n is given by the expression $D_{\Delta x,\Delta t}[c(j\Delta x,n\Delta t)]$. This expression is obtained by Taylors series expansions. We have that:

$$D(c) = D_{\Delta x,\Delta t}(c) - E_j^n(c) = 0$$

At this point we define the following equation:

$$D_{\Delta x,\Delta t}(\bar{c}) = D(\bar{c}) + E(\bar{c}) = 0$$

This is the so-called *modified equation*, an expression introduced by Warming and Hyett (1974). We assume that this equation is completed with a sufficient number of initial and boundary conditions. The solution $\bar{c}(x,t)$ of this equation is such that $\bar{c}(j\Delta x,n\Delta t) = c_j^n$. In other words the modified equation can be considered as the equation that

is actually solved by the finite difference method. Due to the Taylor series expansion this equation has an infinite number of terms. In the modified equation odd-order derivatives are associated with dipersion and even ordered derivatives are associated with dissipation.

As an example we will derive the modified equation associated with the first order upwind method that is given by:

$$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{c_j^n - c_{j-1}^n}{\Delta x} = 0$$

where we have assumed that v>0.

The local truncation error is given by:

$$E_j^n = \frac{1}{2}\Delta t\frac{\partial^2 c}{\partial t^2} - \frac{1}{2}\Delta x v\frac{\partial^2 c}{\partial x^2} + H.O.T.$$

where H.O.T. means higher ordered terms.

If we use the following identity

$$\frac{\partial^2 c}{\partial t^2} = v^2\frac{\partial^2 c}{\partial x^2}$$

then we can rewrite the local truncation errror as:

$$E_j^n = v\frac{1}{2}(\Delta t v - \Delta x)\frac{\partial^2 c}{\partial x^2} + H.O.T$$

The modified equation is now given by:

$$\frac{\partial \bar{c}}{\partial t} + v\frac{\partial \bar{c}}{\partial x} + v\frac{1}{2}(v\Delta t - \Delta x)\frac{\partial^2 \bar{c}}{\partial x^2} + H.O.T. = 0$$

Instead of a pure convection equation this modified equation is a convection diffusion equation with a diffusion coefficient given by:

$$K = -v\frac{1}{2}(v\Delta t - \Delta x)$$

This diffusion coefficient is entirely due to the numerical approximation that is used and it is therefore called *numerical diffusion*.

This modified equation has solutions that are different from the convection equation. For first order upwind differencing these differences are sometimes such that the numerical solution is not sufficiently accurate for practical applications. Near boundaries however this scheme might be quite useful.

In order to have stable solutions the diffusion coefficient must be positive, this implies that:

$$-v\frac{1}{2}(v\Delta t - \Delta x) \geq 0 \rightarrow$$

$$v\Delta t - \Delta x \leq 0 \rightarrow$$

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

In other words in this way we have again obtained the CFL condition. This type of stability analysis based upon the modified equation approach is sometimes referred to as *heuristic stability theory*, see Hirt (1968). It is called heuristic since there is no theoretical basis to proof the correctness of the results obtained in this way.

4.3.4. Conservation, non-oscillating solutions

In this section we treat some well-known characteristics of numerical methods for the approximation of transport equations. In practice it is not sufficient to satisfy only the necessary requirements for consistency, convergence and stability. In some cases it might be necessary that a numerical methods has some additional physical properties such as conservation and non-oscillating solutions.

First we deal with the property of conservation. The simple convection equation that we used as a starting point for this section is obtained by application if a balance principle which is based upon the following:

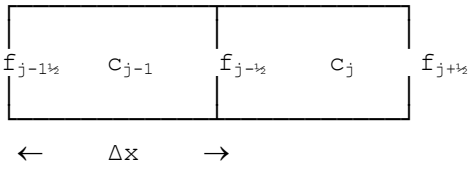
storage+output-input=0

This rule is a simplified description of what is generally called a conservation law. At the discrete level we still want to recognize the balance principle on which the equations, that we want to approximate, are based. To this aim we reformulate our equations in the so-called flux-formulation that is given by:

$$\frac{\partial c}{\partial t} + \frac{\partial \left(F(c) \right)}{\partial x} = 0$$

For our simple convection equation the flux F(c)=vc, where v is a constant.

The numerical grid that we use is considered as a chain of cells where for each cell we apply the balance principle. The fluxes are considered to be located at the mutual cell boundaries:



Fluxes that are leaving one cell are supposed to be entering the adjacent cell. Now for a conservative method it should be possible to formulate it in the following form:

$$\frac{dc_j}{dt} + \frac{F_{j+½} - F_{j-½}}{\Delta x} = 0 \qquad (4.21)$$

In this case the balance principle does not only hold for one cell but also for a chain of cells, as can be easily verified.

For our linear test equation each semi-discretization can be reformulated in a conservative form. We give a few examples:

Second order central differencing: $$F_{j+½} = v\left(\frac{1}{2} c_{j+1} + \frac{1}{2} c_j \right)$$

First order upwind: $$F_{j+½} = \begin{cases} vc_j, & v \geq 0 \\ vc_{j+1}, & v < 0 \end{cases}$$

Second order upwind: $$F_{j+½} = \begin{cases} v\left(\frac{3}{2} c_j - \frac{1}{2} c_{j-1} \right), & v \geq 0 \\ v\left(\frac{3}{2} c_{j+1} - \frac{1}{2} c_{j+2} \right), & v < 0 \end{cases}$$

Note that these formulations are based upon interpolation at the cell boundaries. After substitution into (4.21) it follows that the flux formulations are equivalent to the "normal" formulations given in table 4.2. The advantage of

this flux formulation lies in the easy extension to nonlinear cases or cases with variable coefficients. In this non-linear case conservation is not a trivial property that each approximation has, see e.g. Hirsch (1991).

Exercise 4.L:

Find the flux formulation for third order differencing.

The second property that we want to deal with is the property of is the property non-oscillating solutions. If we consider the physics of a dissolved substance then it is obvious that a concentration will never become negative. Also in the absence of sinks and sources it is not possible that solution shows extrema that were not already in the boundary conditions or in the initial conditions. In other words by pure convection it is impossible that the solution gets new extrema. Numerical approximations however are showing this very often, especially in the neighbourhood of sharp gradients. In the literature test problems with sharp gradients are dealt with very extensively, see e.g. Abbott and Basco (1989). Near these gradients numerical solutions tend to oscillate, showing unrealistic local extrema and unrealistic negative solutions.

Exercise 4.M:

Make a computation for the convection equation using e.g. the Crank-Nicolson scheme. The initial value should be a step function. Use the PROPSC program to carry out this computation (see appendix B). Observe the distortion of the front during the numerical propagation.

All numerical schemes show these negative solutions, with one exception: First order upwind differencing. The fact that positive solutions are guaranteed, is easy to show, consider the following upwind scheme:

$$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{c_j^n - c_{j-1}^n}{\Delta x} = 0$$

Now we rewrite this equation as:

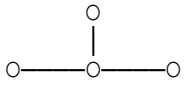
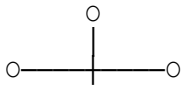
$$c_j^{n+1} = (1-\sigma)c_j^n + \sigma c_{j-1}^n$$

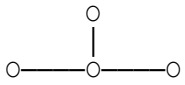
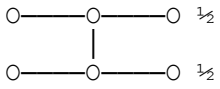
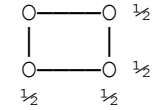
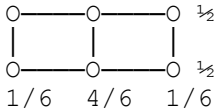
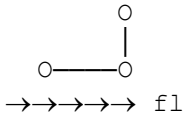
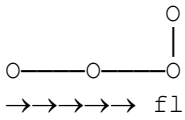
If $0 \le \sigma \le 1$ then the right side of this equation only contains positive coefficients. This means that if all values c^n are positive then all values for c^{n+1} must be positive as well. In other words if the initial condition and the boundary condition are positive then the numerical solution is positive.

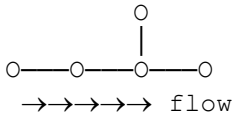
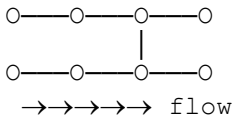
The disadvantage of this scheme however is the large amount of numerical diffusion that is introduced by this approximation that can lead to highly inaccurate results. At this point we come to the true reason why there are so many numerical methods for the convection equation. By using high order upwind schemes for example spurious oscillations will be less than in the case of central differences while the numerical dissipation is not that large as in the case of first order upwind differencing. In this way reasonable solutions can be obtained. Nevertheless the property of monototic solutions, i.e. solutions that are positive and do not contain unrealistic extrema, (for a more precise definition of this concept of monotonicity see e.g Hirsch, 1991), is only fully guaranteed by methods based upon first order upwind differencing. In order to obtain this property of monotonicity, within the framework of accurate numerical approximations of an order higher than one, non-linear schemes are constructed. This however is considered to be beyond the scope of these lecture notes. Extensive references to the relevant literature on this topic are given by Hirsch (1991) or Van Stijn et al.(1987).

4.3.5. Summary of convection methods:

Table 4.3

Discretization in space	Discretization in time	Resulting approximation and stability condition	Name of resulting scheme, layout
Second order central differencing $O(\Delta x^2)$	Euler's Explicit rule $O(\Delta t)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} = -v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x}$ always unstable	
	Midpoint method $O(\Delta t^2)$	$\frac{c_j^{n+1} - c_j^{n-1}$ $}{2\Delta t} = -v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x}$	Leap frog 

Discretization in space	Discretization in time	Resulting approximation and stability condition	Name of resulting scheme, layout
		σ≤1, (point to point)	
	Lax Wendroff $O(\Delta t^2)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x} = \frac{1}{2}\Delta t\, v^2 \frac{c_{j-1}^n - 2c_j^n + c_{j+1}^n}{\Delta x^2}$ σ≤1, (point to point)	Lax Wendroff method 
	Trapezoidal rule $O(\Delta t^2)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} = -\frac{1}{2}v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x} - \frac{1}{2}v\frac{c_{j+1}^{n+1} - c_{j-1}^{n+1}}{2\Delta x}$ always stable	Crank Nicolson 
Box method $O(\Delta x^2)$	Trapezoidal rule $O(\Delta t^2)$	$\frac{c_{j+½}^{n+1} - c_{j+½}^n}{\Delta t} = -\frac{1}{2}v\frac{c_{j+1}^n - c_j^n}{\Delta x} - \frac{1}{2}v\frac{c_{j+1}^{n+1} - c_j^{n+1}}{\Delta x}$ $c_{j+½}=(c_{j+1}+c_j)/2$ always stable, (point to point)	Keller box or Preissmann scheme 
Fourth order M(ass) matrix method $O(\Delta x^4)$	Trapezoidal rule $O(\Delta t^2)$	$M\frac{c_j^{n+1} - c_j^n}{\Delta t} = -\frac{1}{2}v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x} - \frac{1}{2}v\frac{c_{j+1}^{n+1} - c_{j-1}^{n+1}}{2\Delta x}$ $Mc_j=(c_{j-1}+4c_j+c_{j+1})/6$ always stable	Finite element method or Stone and Brian scheme 
First order upwind differencing $O(\Delta x)$	Eulers explicit rule $O(\Delta t)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{c_j^n - c_{j-1}^n}{\Delta x} = 0$ σ≤1, (point to point)	
Second order upwind differencing $O(\Delta x^2)$	Lax Wendroff type $O(\Delta x^2)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{3c_j^n - 4c_{j-1}^n + c_{j-2}^n}{2\Delta x} = \frac{1}{2}\Delta t\, v^2 \frac{c_j^n - 2c_{j-1}^n + c_{j-2}^n}{\Delta x^2}$ σ≤2, (point to point)	

Discretization in space	Discretization in time	Resulting approximation and stability condition	Name of resulting scheme, layout
Third order upwind differencing $O(\Delta x^3)$	Lax Wendroff type $O(\Delta t^3)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{2c_{j+1}^n + 3c_j^n - 6c_{j-1}^n + c_{j-2}^n}{6\Delta x} =$ $\frac{1}{2}\Delta t v^2 \frac{c_{j-1}^n - 2c_j^n + c_{j+1}^n}{\Delta x^2} -$ $\frac{\Delta t^2 v^3}{6}\frac{(-c_{j-2}^n + 3c_{j-1}^n - 3c_j^n + c_{j+1})}{\Delta x^3}$ $\sigma \leq 1+$, (point to point)	QUICKEST (Leonard) 
	Trapezoidal rule $O(\Delta t^2)$	$\frac{c_j^{n+1} - c_j^n}{\Delta t} + \frac{1}{2}v\frac{2c_{j+1}^n + 3c_j^n - 6c_{j-1}^n + c_{j-2}^n}{6\Delta x}$ $+\frac{1}{2}v\frac{2c_{j+1}^{n+1} + 3c_j^{n+1} - 6c_{j-1}^{n+1} + c_{j-2}^{n+1}}{6\Delta x} = 0$ always stable	

Note that if the schemes in this table have the point to point property for $\sigma=1$ then this is indicated in the table.

Exercise 4.N:

List the schemes in this table that are dissipative, establish the order of dissipativity, derive the modified equation.

4.4. Convection diffusion equation

In this paragraph we return to our startingpoint, the transport equation given by:

$$\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} - K\frac{\partial^2 c}{\partial x^2} = \lambda c \tag{4.1}$$

The discretization of this complete equation is the subject of this paragraph.

4.4.1. Examples of discretizations

Also for this equation we follow the method of lines, i.e. first we replace the spatial derivatives by algebraic expression which yields a set of semi discrete equations. Various examples were given in the previous section for the approximation of the diffusive part and of the convective part. If we use the the simplest second order approximations then we obtain:

$$\frac{dc_j}{dt} + v\frac{c_{j+1} - c_{j-1}}{2\Delta x} - K\frac{c_{j-1} - 2c_j + c_{j+1}}{\Delta x^2} = \lambda c_j \tag{4.23}$$

After this step we can replace the derivative in time by some algebraic expression based upon a linear multistep method. Again we can use either inplicit methods or explicit methods. We treated many more methods for the convective part than we did for the diffusive part. This is due to the fact that advection is more difficult to deal with, at least from the numerical point of view. For advection problems one is always facing the dilemma that without numerical dissipation numerical solutions show various non-physical phenomena such as oscillations and negative solutions while if numerical dissipation is added then there might be too much dissipation leading to inaccurate solutions. This leads to the use of high ordered upstream schemes, their dissipation is relatively small except for the short waves that are represented inaccurately anyway such that disspation of waves with these wavenumbers is acceptable. If this linear approach is not sufficient then filter techniques are used, we have given references for that subject.

If there is enough physical disspation these special precautions are not necessary. In that case the semi discretization as given above, is generally speaking, sufficiently accurate. (Especially if one also uses a mass matrix, as arising in the case of finite element methods and of which examples are given in our tables.) The question arises now when is the physical disspation enough in order to be able to use the discretization given by (4.23). To estimate this one studies a steady state situation described by (see also exercise 2.J):

$$v\frac{\partial c}{\partial x} - K\frac{\partial^2 c}{\partial x^2} = 0$$

The domain of this equation is (0,X). The boundary conditions are given by $c(0)=C^0$ and $c(X)=0$.

An example of the physics that is described by this equation is for example the salt concentration pattern in an estuary, see Vreugdenhil (1989). For this example (4.23) becomes:

$$v\frac{c_{j+1} - c_{j-1}}{2\Delta x} - K\frac{c_{j-1} - 2c_j + c_{j+1}}{\Delta x^2} = 0$$

We rewrite this equation as:

$$(\tfrac{1}{2}R - 1)c_{j+1} + 2c_j - (\tfrac{1}{2}R + 1)c_{j-1} = 0$$

where $R=v\Delta x/K$, R is called the *cell Reynolds number* or *cell Péclet number*.

As already treated in section 3.1 the general solution of this equation can be denoted as:

$$c_j = d_1 r_1^j + d_2 r_2^j$$

r_1 and r_2 are the roots of the characteristic equation given by:

$$(\tfrac{1}{2}R - 1)r^2 + 2r - (\tfrac{1}{2}R + 1) = 0$$

d_1 and d_2 are constants that are determined by the boundary conditions and the numerical approximations near the boundary.

The roots of this characteristic equation are:

$$r_1 = 1, \quad r_2 = \frac{(2+R)}{(2-R)}$$

To prevent c_j from oscillating one must have that $r_2 \geq 0$ which means that $R<2$. In other words if $\Delta x<2K/v$ then oscillations will not occur. In this case (4.23) is , again generally speaking, a satisfactory approximation of (4.1). It can be shown that this condition is also necessary to guarantee positive solutions in the time dependent case where (4.23) is combined with Euler's explicit rule for the discretization in time. In that case, while we assume that $\lambda=0$, (4.23) becomes:

$$\frac{c_j^{n+1} - c_j^n}{\Delta t} + v\frac{c_{j+1}^n - c_{j-1}^n}{2\Delta x} - K\frac{c_{j-1}^n - 2c_j^n + c_{j-1}^n}{\Delta x^2} = 0$$

This equation is rewritten as:

$$c_j^{n+1} = (q-\sigma)c_{j-1}^n + (1-2q)c_j^n + (q+\sigma)c_{j+1}^n$$

where $q=K\Delta t/\Delta x^2$ and $\sigma=v\Delta t/\Delta x$, the latter is the Courant number.

To guarantee positive solutions we must have that all coefficients of this equation are positive. This is the case if $q \leq \frac{1}{2}$ and $\frac{1}{2}|c| \leq q$. The latter relation is equivalent with $R \leq 2$ which is equal to the cell Peclet condition that we already obtained. It is to be noted that both conditions are also sufficient conditions to fulfil the Von Neumann stability condition, see Hirsch (1991) pp 403-406 where also the necessary and sufficient conditons to fulfil Von Neumann stability are given.

Ofcourse we can integrate (4.23) also by other linear multistep methods such as the trapezoidal rule or the θ-method.

Exercise 4.O:

(i) Give the finite difference equations when the θ method is used and
(ii) show the unconditional stability for θ≥½.

A special class of one step methods for the integration in time of (4.23) or an equivalent equation with different approximations for the spatial derivatives is the so called *method of fractional steps*. In this method the original differential operator is factorized in a sequence of equations. For (4.21) we may obtain:

Decay (Physical, chemical or biological) $\frac{dc}{dt} = \lambda c$

Diffusion $\frac{\partial c}{\partial t} - K\frac{\partial^2 c}{\partial x^2} = 0$

Advection $\frac{\partial c}{\partial t} + v\frac{\partial c}{\partial x} = 0$

Each equation is approximated by a one step method in a sequential way as follows:

$$c^* = A_r c^n$$
$$c^{**} = A_d c^*$$
$$c^{n+1} = A_a c^{**}$$

For each stem a method can be used that is specific for that type of equation. The overall scheme becomes:

$$c^{n+1} = A_r A_d A_a c^n$$

This shows that if at each stage a stable scheme is used then the overall scheme must be stable as well. The order of accuracy of this method is O(Δt), only in special cases it is possible to construct each step such that a higher order of accuracy can be obtained. An example could be the following sequence:

Decay step, Euler implicit: $c_j^* = \frac{1}{1 - \Delta t\lambda} c_j^n$

Diffusion step, Euler explicit: $c_j^{**} = qc_{j-1}^* + (1 - 2q)c_j^* + qc_{j+1}^*$

Advection step, Lax Wendroff: $c_j^{n+1} = \frac{1}{2}\sigma(1+\sigma)c_{j-1}^{**} + (1 - \sigma^2)c_j^{**} + \frac{1}{2}\sigma(1 - \sigma)c_{j+1}^{**}$

For more detailed information on this method one is referred to Hirsch (1991).

4.5. Shallow water equations, long waves

The last set of equations that we consider are the shallow water equations. As mentioned already in exercise 2.O a simplified version of these equations is given by:

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} = 0$$
$$\frac{\partial h}{\partial t} + d\frac{\partial u}{\partial x} = 0 \qquad (2.23')$$

As we have done continuously we discretize these equations by the method of lines. Paragraph 4.5.1 describes both the discretization in space and the discretization in time. Paragraph 4.5.2 shows that for the analysis of stability or of accuracy in terms of propagation properties it is sufficient to consider a simple convection equation.

4.5.1. Discretizations of the shallow water equations

As done for all previous equations we replace first the spatial derivatives by algebraic expressions. If we use second order central differences then we obtain the following set of ODE's:

$$\begin{aligned} \frac{du_j}{dt} + g\frac{h_{j+1} - h_{j-1}}{2\Delta x} &= 0 \\ \frac{dh_j}{dt} + d\frac{u_{j+1} - u_{j-1}}{2\Delta x} &= 0 \end{aligned} \qquad (4.24)$$

Again for the discretization in time the linear multistep methods of chapter 3 can be applied. If we consider (4.24) we see that (4.24) consists of two independent sets of equations, one for $\{u_{2j}, h_{2j+1}\}$ and one for $\{u_{2j+1}, h_{2j}\}$. This means that there is no loss af accuracy if one these two sets is cancelled. If we cancel the first then this means that we compute approximations for h at even number points and approximations for u at odd numbered points. The grid now looks as:

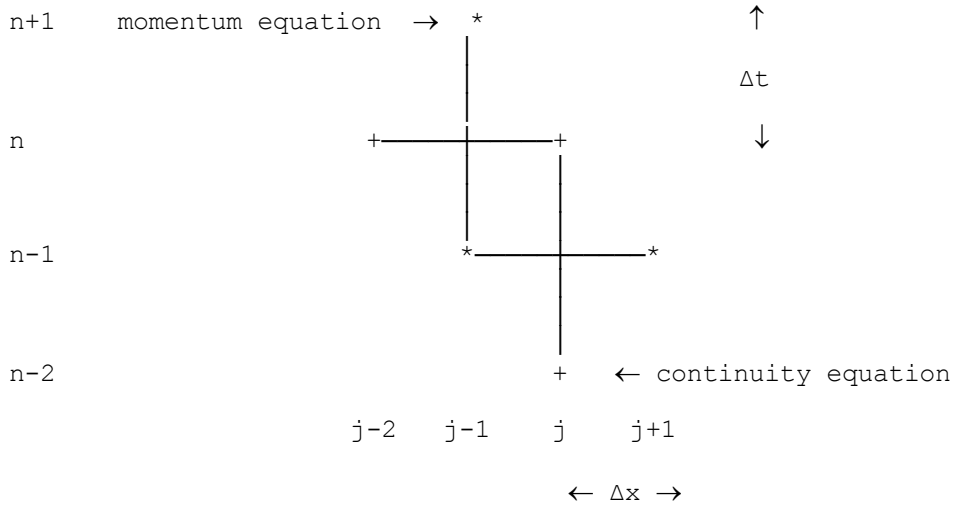
$$\begin{array}{c} \cdots \; h_{j-1} \quad u_j \quad h_{j+1} \quad u_{j+2} \quad h_{j+3} \; \cdots \\ \leftarrow \Delta x \rightarrow \end{array}$$

Such a grid is called a *staggered grid* in space. Of the explicit methods the mid-point rule is a possibility. This yields:

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} + g\frac{h_{j+1}^n - h_{j-1}^n}{2\Delta x} = 0$$

$$\frac{h_j^{n+1} - h_j^{n-1}}{2\Delta t} + d\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x} = 0$$

For this finite difference scheme staggering can take place in time as well. This yields the following grid in both time and space:



"*" is a u-point while "+" is a h-point, i.e. points where either values for u or for h are to be computed.

The method based upon this combination of the staggered grid in time and space, second order central differencing and the mid-point rule is called the *Leap-frog scheme*.

In many cases the integration in time will be based upon the θ method, where θ>½. This yields the following method:

$$\frac{u_{j+1}^{n+1} - u_{j+1}^{n}}{\Delta t} + (1-\theta) g \frac{h_{j+2}^{n} - h_{j}^{n}}{2\Delta x} + \theta g \frac{h_{j+2}^{n+1} - h_{j}^{n+1}}{2\Delta x} = 0$$

$$\frac{h_{j}^{n+1} - h_{j}^{n}}{\Delta t} + (1-\theta) d \frac{u_{j+1}^{n} - u_{j-1}^{n}}{2\Delta x} + \theta d \frac{u_{j+1}^{n+1} - u_{j-1}^{n+1}}{2\Delta x} = 0$$

Another scheme that is often used for commercial systems for the simulation of flow in channel networks is the so-called *Box scheme or Preissmann scheme*. This scheme is based upon the following set of semi-discrete equations:

$$\begin{aligned} \frac{du_{j+½}}{dt} + g \frac{h_{j+1} - h_{j}}{\Delta x} &= 0 \\ \frac{dh_{j+½}}{dt} + d \frac{u_{j+1} - u_{j}}{\Delta x} &= 0 \end{aligned} \qquad (4.25)$$

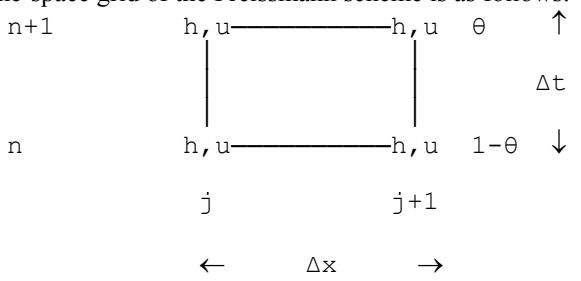
where $u_{j+½} = \frac{u_j + u_{j+1}}{2}$ and $h_{j+½} = \frac{h_j + h_{j+1}}{2}$.

It is to be noted that this method is the same as the box method that we treated for the simple convection equation. This scheme is only combined with implicit methods, in most case the θ method. This gives:

$$\frac{u_{j+½}^{n+1} - u_{j+½}^{n}}{\Delta t} + (1-\theta) g \frac{h_{j+1}^{n} - h_{j}^{n}}{\Delta x} + \theta g \frac{h_{j+1}^{n+1} - h_{j}^{n+1}}{\Delta x} = 0$$

$$\frac{h_{j+½}^{n+1} - h_{j+½}^{n}}{\Delta t} + (1-\theta) d \frac{u_{j+1}^{n} - u_{j}^{n}}{\Delta x} + \theta d \frac{u_{j+1}^{n+1} - u_{j}^{n+1}}{\Delta x} = 0$$

The time-space grid of the Preissmann scheme is as follows:



Exercise 4.P:

Compute the local truncation error and assess the consistency of the schemes given above.

The compact structure of this method, i.e. the fact that the complete scheme is defined on only two grid points in space makes this method very suitable for space varying grid sizes as is often the case in practical applications.

In fact all the methods, based upon central differencing, that we treated for the simple advection equation can be used also for the shallow water equation. In practice however the ones that we have given here are most widely used.

4.5.2. Stability and propagation properties

As usual we verify stability by the Von Neumann condition. This means that we consider an initial value problem for a discrete set of equations given by:

$$\sum_{m=0}^{k} \frac{\alpha_m}{\Delta t} M y^{n+m} = \sum_{m=0}^{k} \beta_m A y^{n+m}$$

Here y denotes a vector with values ζ_j, u_j $-\infty \le j \le \infty$, M denotes some averaging operator occuring for example in case of the box scheme, A denotes the matrix resulting from the semi discretization of the shallow water equation and the coefficients α and β result from application of a general linear multistep method, given by (3.24), to a system of ODE's.

The initial conditions are given by:

$$h_j^0 = e^{ij\xi},\ u_j^0 = e^{ij\xi},\ \xi = \frac{2\pi\Delta x}{L}$$

Substitution of this initial condition yields:

$$\sum_{m=0}^{k} \frac{\alpha_m}{\Delta t} \hat{M} \begin{bmatrix} \hat{u}^{n+m} \\ \hat{h}^{n+m} \end{bmatrix} = \sum_{m=0}^{k} \beta_m \begin{bmatrix} 0 & g\hat{L} \\ d\hat{L} & 0 \end{bmatrix} \begin{bmatrix} \hat{u}^{n+m} \\ \hat{h}^{n+m} \end{bmatrix}$$

This equation can be rewritten as:

$$\sum_{m=0}^{k} \frac{\alpha_m}{\Delta t} \begin{bmatrix} \hat{u}^{n+m} \\ \hat{h}^{n+m} \end{bmatrix} = \sum_{m=0}^{k} \beta_m \begin{bmatrix} 0 & g\hat{L}/\hat{M} \\ d\hat{L}/\hat{M} & 0 \end{bmatrix} \begin{bmatrix} \hat{u}^{n+m} \\ \hat{h}^{n+m} \end{bmatrix}$$

From chapter 3 we know that it is sufficient to study only:

$$\sum_{m=0}^{k} \frac{\alpha_m}{\Delta t} \hat{c}_{1,2}^{n+m} = \sum_{m=0}^{k} \beta_m \hat{\lambda}_{1,2} \hat{c}_{1,2}^{n+m}$$

where $\hat{\lambda}_{1,2}$ denote the eigenvalues of $\begin{bmatrix} 0 & g\hat{L}/\hat{M} \\ d\hat{L}/\hat{M} & 0 \end{bmatrix}$, which are given by $\pm(gd)^{1/2}\,\hat{L}/\hat{M}$.

In other words it is sufficient to study only the simple convection equation given by:

$$\frac{\partial c}{\partial t} + U\frac{\partial c}{\partial x} = 0 \qquad (4.13)$$

but in this case $v = \pm(gd)^{1/2}$.

Also for the propagation properties it is sufficient to study only this simple test equation.

Example:

We pose following problem:

An engineer wants to model a river with a uniform depth of ±10 m. For this purpose he will be using a program package that is based upon central differencing for the spatial discretization. At one location an open boundary is located where tidal elevations are to be prescribed as boundary conditions. He requires that tidal waves with a period of ±3 hours, say 10^4 seconds, are represented with a relative phase error of less than 0.1 %. What is the maximum spatial gridsize that he can use to fulfil that requirement?

Answer:

From the section on the convection equation we know that the relative phase speed for this method is given by $\sin(\xi)/\xi$. For a relative phase error of less than 10^{-3} we must have that $\sin(\xi)/\xi > 1\text{-}10^{-3}$. By Taylor series expansion we estimate:

$$\frac{\sin\xi}{\xi} \approx \frac{\xi - \frac{1}{6}\xi^3}{\xi} = 1 - \frac{1}{6}\xi^2$$

In other words we must have that $\xi^2<10^{-3}$ or $\xi<0.032$. This implies that $L/\Delta x >200$, in other words the number of point per wave length must be larger than 200. For a uniform depth of 10 m the wave speed is ± 10 m/s. For a period of 10^4 seconds this implies a wavelength of 10^5 m. A number of grid points per wavelength larger than 200 now means a grid size less than 500 m.

4.6. Summary

The discretization process described in this paragraph describes the discretization of an equation given by:

$$\frac{\partial \vec{c}}{dt} = \mathsf{L}\vec{c}$$

This process of discretization in essence consists of two parts, first the construction part and second the analysis part.

I The construction step:

The construction part is based upon the *method of lines*. This means that first the operator L is discretized, i.e. replaced by algebraic expressions denoted by $\mathsf{L}_{\Delta x}$. For example $\partial c/\partial x$ is replaced by $(c_{j+1}-c_{j-1})/2\Delta x$. The result of this step is a set of ODE's denoted by:

$$\frac{d\vec{c}}{dt} = \mathsf{L}_{\Delta x}\vec{c}$$

To complete the discretization process we apply a linear multistep methods to this equation, then we obtain:

$$\sum_{m=0}^{k}\frac{\alpha_m}{\Delta t}\vec{c}^{\,n+m} = \sum_{m=0}^{k}\beta_m \mathsf{L}_{\Delta x}\vec{c}^{\,n+m}$$

Now the discretization is completed. The resulting set of equation can be solved by a computer, either explicit or implicit. Alternatives for linear multistep methods are Runge Kutta methods, e.g. Lambert (1976), Lax Wendroff methods in case of the convection equation or fractional step methods.

II The analysis step:

The resulting set of finite difference equations is analysed for the following aspects:

Consistency	By means of Taylor series. The finite differnce equations are to be consistent with the differential equations.
Stability	By means of the Von Neumann condition. The solution of the finite difference scheme must be stable, it should be bounded and not sensitive to small disturbances.
Convergence	The solution of the finite difference scheme must be convergent with the solution of the differential equation. This implies that the difference between the two solutions becomes arbitrary small if Δx and $\Delta t \rightarrow 0$. A consistent and stable scheme is assumed to be convergent. This is based upon the Lax equivlalence theorem.

These three aspects are a necessity for finite difference schemes the following aspects are usefull to analyse but not strictly necessary:

Local truncation error	By means of Taylor series expansion.
Modified equation	By means of Taylor series expansion, to analyse the order of numerical dissipation.
Relative relaxation time	By means of Fourier analysis to analyse the accuracy of the disspation rate.
Propagation properties	By means of Fourier analysis, to analyse *amplitude errors* and *phase errors* for convection dominated problems.
Stiff stability	To analyse whether diffusion approximations maintain their disspative nature, despite of the size of the timestep.

Monotonicity	To check whether non-physical oscillations or non-physical negative values for concentrations of dissolved or suspended matters will not occur in the numerical solution.
Cell Péclet number	Oscillation condition for convection-diffusion approximations.
Conservation	Conservation principle fulfilled also by numerical method.

Chapter 5 The structure of a computer model: DUFLOW

For the solution of engineering problems numerical approximations are available in the form of standard computer programs. Examples of such programs are continuous systems modelling packages such as DYNAMO (Richardson and Pugh, 1981), Stella and TUTSIM, or programs for flow and transport in networks such as DUFLOW.

In the continuous systems modelling packages a user can formulate a set of (first order) ordinary differential equations with time as the independent coordinate. The range of applications is very wide, for instance social end economic systems (Forrester, 1961; Meadows et al., 1974), process engineering, water quality modelling in a lake etc. The number of (interacting) parameters can be very large, even hundreds. The user specifies the expressions for the time derivatives of all the parameters, their initial values, the integration time step and (sometimes) the desired method of integration, and the output he wants. The package carries out the integration of the differential equations, and it shows the results in the form of graphs, tables etc. It is noted that in many simulation packages the method of integration is more sophisticated than the ones treated in chapter 3 of this book.

In this chapter we will discuss the model DUFLOW (Spaans et al.,1989) in more detail, show how the numerical approximation is used in this program, and describe data the user must enter when he wants to make an actual simulation. The program DUFLOW is more specific than the continuous systems modelling packages, because the partial differential equations are built into the computer program already, so that the range of applications is narrower. DUFLOW contains the shallow water equations for one spatial dimension, and the corresponding equation for transport of a pollutant.

Programs such as DUFLOW do not only contain a straightforward discretization of the shallow water equation as discussed in the previous chapter; a number of features have been added to enhance the usability. As a result the program can be used not only for single channels, but also for networks of canals and rivers. There is a variety of boundary conditions, and there is the possibility to include flow control structures such as weirs and pumping stations in a model.

The next section presents the generalisation of the shallow water equation to a network of channels; section 5.3 discusses the input data.

5.1. Network instead of single channel

If a computer program must be able to handle a network, there is the necessity for the user to tell the program what the structure of the network is, in other words how the various channels are connected. This is done as follows: a network is supposed to consist of channel sections, sections for short, and nodes; each section connects two nodes. Naturally we find the discharges in the sections, and because we assume that the water level is continuous over a node, the water levels are defined in the nodes. In defining the network a number is assigned (by the user) to each node and each section, and the program keeps track of the structure by maintaining a list where it finds which nodes are connected by each section; if M is a section number, $K_1(M)$ and $K_2(M)$ are the nodes at the ends. In DUFLOW the user enters the values of $K_1(M)$ and $K_2(M)$ along with the data on cross-sections etc. for section M. The order of the nodes K_1 and K_2 determines the sign of the flow in the channel section. In other words, if the user tells the program that section 11 connects nodes 7 and 14 (in that order) the program will take flow from 7 to 14 as positive and from 14 to 7 as negative.

DUFLOW is based on the box method. Many programs for flow in networks however are based on an alternating scheme. In an alternating scheme a discharge point is defined halfway each section (see figure 5.1) and a water level is defined in each node. As discussed in chapter 4 the box method has discharges and water levels in the same points; in DUFLOW this is solved by having the water levels in the nodes, as before, and having two discharges per channel section, one at each end of the section (see figure 5.2).

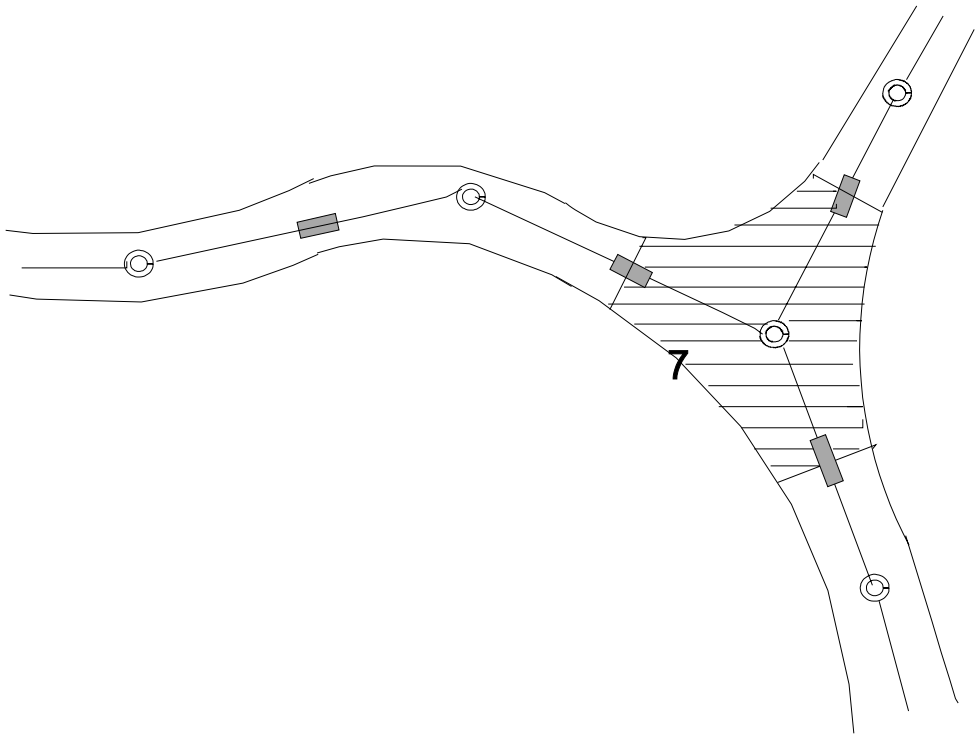


Figure 5.1. Network schemetization with alternating numerical scheme. The hatched area is the storage surface of node 7.

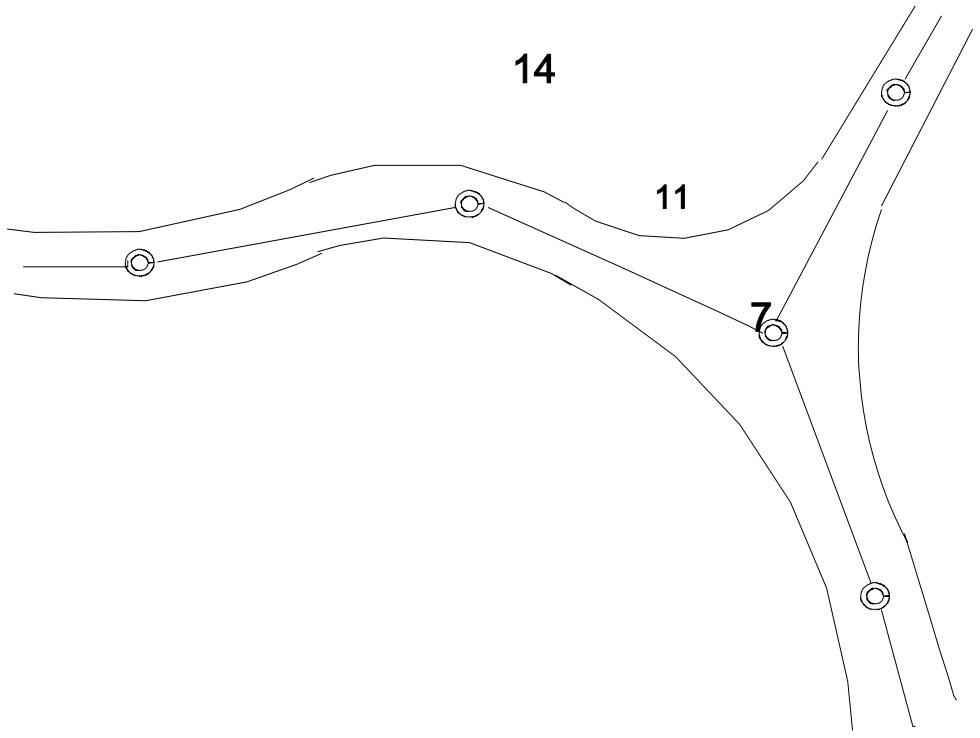


Figure 5.2. Network schemetization with box scheme

When considering the differential equations the branching points can be considered as internal boundaries. In these points the following (internal) boundary conditions hold: the water level is continuous over the node, and the sum of the discharges to the node is zero.

In a computer program the branching points are not boundary points in the sense that the user must prescribe the above conditions as boundary conditions; once the program knows the structure of the network these conditions will

be taken into account automatically. In fact in the program there is no distinction between true branching points and other nodes. We will illustrate this procedure first for an explicit computation procedure based on the alternating scheme in space, and then with the procedure employed in DUFLOW which uses the box scheme.

The explicit method is one which was often used in the first generation of network models, a popular scheme was the leap-frog scheme (see section 4.5.1). It uses a staggered grid which means that discharges (or velocities) and water levels are computed at different places and in the case of the leap-frog method also at different times; values of Q are computed at times $(n-1)\Delta t$, $n\Delta t$ etc., values of H at $(n+1/2)\Delta t$ etc. (see figure 5.3).

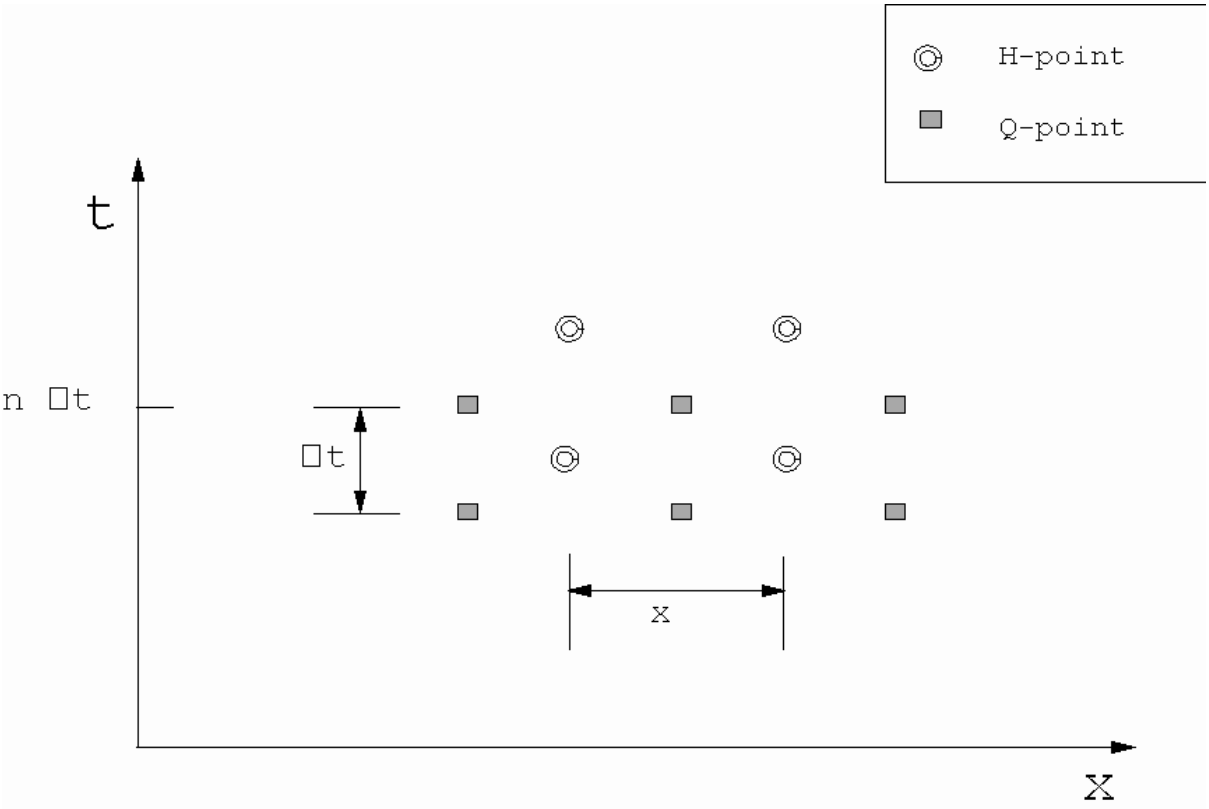


Figure 5.3. The leap-frog scheme in x-t-plane

In computing the discharges one uses the equation of motion; the value at $n\Delta t$ is unknown, the one at $(n-1)\Delta t$ is known from previous computations, or from the initial condition. dQ/dt is approximated by the difference between the two values of Q, divided by Δt. The water level gradient is found from the values of H at the two adjacent nodes; which these nodes are is known to the program. The values of H are also known, they are taken at $(n+½)\Delta t$. The advective term is usually neglected in programs using the leap-frog method, because in order to compute $\partial Q U/\partial x$ one has to use values of Q in adjacent sections; in a network environment it is difficult however to decide which these sections are. The bottom friction term on the other hand is expressed in the values of Q in the section itself, so this term is incorporated without difficulty. Q at the "new" time is now the only unknown in the equation so it is easily calculated.

After all the values of Q at time $n\Delta t$ are calculated, the values of H are found using the continuity equation. In a network environment we assign to a node a surface of storage which is equal to the sum of half of the storage surfaces of all adjacent sections; the surface for one node is hatched in figure 5.1. The amount of water is conserved, so we make $B_{ss}dH/dt$ (where B_{ss} is the storage surface of the node) equal to the sum of all the Q's flowing to the node. It is not necessary to determine for each node which channel sections come to the node, and what is their orientation. Both problems are solved by introducing a quantity (SumQ in the pseudocode program below) which represents the sum of the Q values for one node. The pseudocode program reads:

Repeat until end of computational period:

Increase time by $\Delta t/2$

```
For all channel sections in the network do:
        K1 = from-node, K2 = to-node
        Compute water level slope from H(K1)-H(K2)
        Compute new value for Q in the section using
        the equation of motion
        Make SumQ(K1) = SumQ(K1) - Q
        Make SumQ(K2) = SumQ(K2) + Q
-------------------------------------------------------------------
Increase time by Δt/2
For all nodes in the network do
        {K is the node number}
        {Bss(K) is the storage surface of node K}
        Add SumQ(K)* Δt/Bss(K) to H(K)
        Make SumQ(K) = 0
-------------------------------------------------------------------------------
```

Obviously this program is incomplete; it should have provisions for nodes where boundary conditions are defined, and for sections which are flow control structures instead of open channel sections.

Exercise 5.A:

Write a computer program based on the leap-frog method for the shallow water equation. The program should read the network structure and the channel dimensions from a file. Let node 1 be a boundary node with given periodically varying water level (if you find this too simple, feel free to make it more complicated).

Note that the leap-frog method is conditionally stable. As shown in figure 4.2 the characteristics may not go outside the numerical domain of dependence; the propagation velocity is now $v = \pm\sqrt{gA/b}$, and the stability condition thus is

$$\sqrt{gA/b}\,\frac{\Delta t}{\Delta x} \le 1$$

Implicit method: DUFLOW

DUFLOW is based on the box scheme, so in each section we have two equations at our disposal, one is a discretized version of the continuity equation, the other is a discretized version of the equation of motion. In these two equations four unknowns appear, viz. values of Q and H at both ends of the section. The H's are defined at the nodes, so two of the unknowns are $H(K_1)$ and $H(K_2)$, the Q's are defined at both ends of the section M itself, so the other unknowns are $Q_1(M)$ and $Q_2(M)$. Using the two equations in the section we can express the Q's by:

$$\begin{aligned} Q_1^{n+1}(M) &= N_{11}H^{n+1}(K_1) + N_{12}H^{n+1}(K_2) + N_{13} \\ Q_2^{n+1}(M) &= N_{21}H^{n+1}(K_1) + N_{22}H^{n+1}(K_2) + N_{23} \end{aligned} \qquad (5.5)$$

In addition we have the relations at the nodes, viz. the requirement that the sum of the discharges to the node is zero. Substitution of the equations (5.5) transforms these into equations involving values of H at the nodes. How this is done is shown in the pseudocode program below. In this program A is the matrix in which the relations for H are stored. The equations for H read:

$$\Sigma_k\, A_{ik} H_k + R_i = 0 \qquad (5.6)$$

The pseudocode program for the computation of H and Q in the network reads:

```
-------------------------------------------------------------------------------
Repeat until end of computational period:
        Increase time by Δt
        Make all elements in matrix A and in R =0
        ---------------------------------------------------------------
        For all channel sections in the network do:
                K1 = from-node, K2 = to-node
```

Compute coefficients N_{11}, N_{12}, N_{13}, N_{21},
N_{22}, N_{23} from continuity equation and
equation of motion
Subtract from $A_{k1,k1}$: N_{11}
Subtract from $A_{k1,k2}$: N_{12}
Subtract from R_{k1}: N_{13}
Add to $A_{k2,k1}$: N_{21}
Add to $A_{k2,k2}$: N_{22}
Add to R_{k2}: N_{23}

Solve values of H from linear system with
matrix A and right hand side R

For all channel sections in the network do:
{ M is the section number}
K_1 = from-node, K_2 = to-node
Make $Q_1(M) = N_{11}*H(K_1) + N_{12}*H(K_2) + N_{13}$
Make $Q_2(M) = N_{21}*H(K_1) + N_{22}*H(K_2) + N_{23}$

After the matrix A is built, the values of H can be calculated. To do so various standard procedures are available. The simple Thomas algorithm described in section 4.2.2 can be used only for a simple channel without bifurcations; so we need a more complicated procedure here. DUFLOW has two options for solving the linear system of equations, one is a direct method (Gaussian elimination), the other is an iterative method.

The after the values of H have been calculated the discharges can be found using the same equations that were used to eliminate them.

Note that as a consequence of the procedure to handle a branching point a dead end automatically has a boundary condition Q = 0; the sum of the discharges to such a node consists of only one contribution! As a consequence DUFLOW, like most network programs, does not need an explicitly given boundary condition at a dead end.

Exercise 5.B:

If inflows occur at various nodes of the network, the pseudocode program needs a small extension. Extend the program such that given arbitrary inflows at all nodes can be taken into account.

5.2. Input of boundary conditions etc.

Input to the DUFLOW program can be separated into the following categories:

- general data,
- geometry of the network,
- boundary conditions,
- output requests.

General data are for instance the time step, the value of time at the start of the computation, the length of the computational period, the choice of friction formula (there are two options), the choice of the method of solution of the linear system of equations etc.

In the input of the network geometry the following data have to be given for each open channel section: number of begin node, number of end node, length of the section, cross-section data at begin and end of the section (bottom level, width of flow as function of water level, width of storage as function of water level, friction coefficient as function of water level).

In many real-life networks flow control structures such as weirs, culverts and pumping station occur. Although such structures are sometimes interpreted as boundary conditions for the shallow-water equation they are treated in DUFLOW and most other programs just as channel sections. The only difference is that coefficients such as N_{11} etc. are computed with different formulas. The input of structures is also much the same as that of channel sections.

Boundary conditions proper are: given value of H in a node, inflow into a node. DUFLOW will ask for which nodes you want to prescribe a boundary condition, what quantity has to be prescribed and how this quantity varies as function of time.

After a computation has been made, the user can ask DUFLOW to make graphs or tables of water level, discharge or velocity. Graphs can be presented giving these quantities as function of x (along a certain path in the network) or as function of time.

5.3. Flow and transport computation

In the computation of concentrations we need flow data as is shown by equation (2.9) and (2.10). So a flow computation must be carried out before a transport computation is done. There are two set-up of computer programs to do this. In one the computer programs for flow and for transport are separate; first the flow program is run which writes flow data for the whole computational region and the whole computational period to disk memory; then the transport program is run which reads the flow data from disk and carries out the transport computation. In the second set-up the flow computation and transport computation are integrated in one program. Now the program first computes flows for each time interval from t to $t+\Delta t$ and then carries out the transport computation for the same interval.

Chapter 6 Usage of numerical models

A.1. Overview

Let us consider a very general engineering problem. Engineering is concerned with designs to bring about changes in the existing real world. Usually there are a number of alternative designs available. Numerical models (among others) are used to predict the consequences of the change of the physical reality as designed. In such a case there are at least two sets of computations needed:

- one set of computations to simulate the existing situation; the results of these computations are used together with measurements in the existing situation to verify and calibrate the model,
- the second set of computations to simulate the designed changed situation; these are used to check whether the design is sound.

Design and modeling are processes in which certain stages can be distinguished. There are strong parallels between the two, and also between these and the research process. For each process we give a simple scheme below:

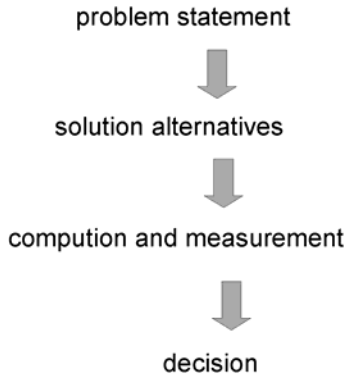


Fig. 6.1. scheme of the design process

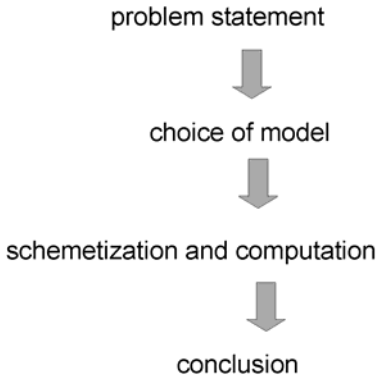


Fig. 6.2. scheme of the modeling process

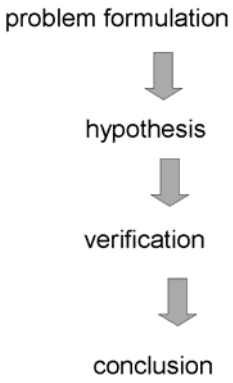


Fig. 6.3. scheme of the research process

In reality these processes are iterative. In the case of a design process a design alternative may be modified after a series of computations or measurements, and sometimes the problem statement has to be modified. The modeling process is a sub-process of the design process. Consequently there may be iteration within iteration.

What these schemes make clear is, that a problem statement is essential. In the case of a numerical simulation the following things must be clearly stated: what change with respect to the present state must be simulated, and which quantity in which region or place is wanted as outcome of the computation, and with what accuracy.

In preparing the computation itself the following steps are taken:

1. decide upon the equation which is to be used; for instance: can we do the job with a one-dimensional model or do we have to go to a 2-D or even 3-D model; can we assume the situation to be stationary or not? This determines the type of computer program to be used.
2. determine the computational region; here we need to know the modification (change with respect to the present situation) and the quantity of interest.
3. decide upon the initial and boundary conditions to be used in the computation; this is strongly related to the previous point.
4. decide upon the size of the time steps and spatial steps. Here we need the statement on the required accuracy from the problem statement.

Whether or not we can use a 1-D model depends on a number of arguments: the geometry of the physical space, the presence of stratification, the nature of the quantity of interest.

- In the geometry we must be able to recognize one-dimensional channels; the presence of storage areas which are not so clearly one-dimensional is usually not harmful.
- if there are density differences over the vertical in the region, the velocity distribution will be different from the one which was assumed in the derivation of the 1-D equations. The presence of density differences in axial direction (along the axis of the 1-D channel) is not a problem, as long as the 1-D computer program can handle such density difference (few of them can).
- Whether or not a thing like stratification prevents us from using the 1-D approximation also depends on the nature of the quantity of interest. If we are interested in water levels some stratification usually is not much of a problem, but if we want to compute transport of pollutant stratification is something to take seriously.

There are many sources of inaccuracy in a numerical simulation:

Schemetization errors: errors due to the choice of the equation, errors due to the neglect of terms in the equation, errors due to inaccuracy in the coefficients used in the model, errors due to the finiteness of the computational region.

Discretization errors: errors due to the choice of the numerical scheme, and due to the finiteness of the space and time steps.

How to control the schemetization errors and how to take into account measurements is discussed in section 6.3. The discretization errors are treated in section 6.4. Sometimes the results are not just inaccurate but grossly in error; section 6.5 gives suggestions concerning possible causes and remedies.

6.1. Choice of computational region and boundary conditions

A computational region must be limited both in time and in space if only because of the limited capacity of the computer. Obviously the computational region must not be larger than necessary. What is necessary depends on the physical situation, not on the numerical approximations.

When choosing the size of the computational region we must remember that we want to simulate a modified situation while we are able to gather information only on the present situation. For instance we can carry out measurements to establish a boundary condition but we must make sure that this boundary condition is still valid in the modified situation.

An initial or boundary condition will be incorrect if (*a*) it is influenced by the modification, and (*b*) it in turn influences the quantity of interest.

For instance: why do people choose a water level as boundary condition for a flow model at a place where a river enters the sea? This is not because a water level is easily measured; the true reason is that the sea is so large compared to the river that whatever we modify in the river, we may safely assume that the water level at sea will hardly be influenced.

A correct boundary must be chosen such that either

the boundary condition is not influenced by the modification of the physical system

or

the boundary condition does not influence the quantity of interest in the region of interest.

The above example is a boundary condition fulfilling the first condition. An example of a boundary condition fulfilling the second condition is encountered when we carry out a flow computation in an upstream stretch of the river. Making a model stretching all the way down to the sea is uneconomical. We know from exercise 2.P that a downstream boundary condition in a river has only a limited region of influence the length of which is 2 or 3 times d/I_b. Assuming that at this downstream boundary there is no structure which we can use as boundary, we must assume that the river flow is undisturbed and that a good approximation is to take equation (2.24) as boundary condition. This condition states that the flow is uniform; although this is not entirely true the errors it introduces are minor and if the place of the boundary is a few times d/I_b downstream of the region of interest, the flow condition in the region of interest is not disturbed.

In choosing the time when we start the computation we have a problem similar to the choice of the boundary. Very often our information on the initial state of the system is little. We usually start the computation so far back that the influence of the initial condition on the state in the region of interest is negligible. For instance in a river we often want to start the flow computation with a steady state, or in an estuary we want to start with a periodical state (periodical with tidal period). We let the program run for a time long enough to make the model stationary, or periodical resp.. The length of this computational period is often related to the time it takes for the long wave to travel over the entire model.

Consider the following example: In a river a flood prevention structure is to be designed. One of the alternatives is to build a weir in the river together with a basin connected to the river (see figure 6.4). The idea is that the basin stores part of the flood thereby decreasing the maximum water levels downstream of the structure.

At a point M (also in figure 6.4) the water level has been measured for a large number of years. The effectiveness of the structure must be demonstrated by simulations of some historical floods; one of these is shown in figure 6.5.

In applying the 1-D model we first choose the upstream boundary. It would seem attractive to choose M as the upstream boundary and the measured water level as boundary condition. This incorrect however because the water level at M will be influenced by the building of the weir; it is therefore unsuitable as boundary condition for the design situation.

The boundary must be chosen further upstream where the flow is not influenced by the weir. Computations with the present situation must show whether the boundary condition applied there is correct in the sense that the measurements at M are correctly reproduced.

In choosing the downstream boundary we consider the region of interest. We must show the effect of the design on the water level at points downstream of the weir, at say 10 and 20 km from the weir. As described above, the

downstream boundary must be 2 or 3 times d/I_b further downstream, and equation (2.24) is used as boundary condition.

Exercise 6.A:

Make a 1-D model of the above situation with DUFLOW or another network program. Assume a cross-section of the river as shown in figure 6.6; take for the friction coefficient C_{fr} 0.004. Estimate the upstream boundary condition and run the model for the present situation. Then introduce the flood prevention structure and run again for the modified situation.

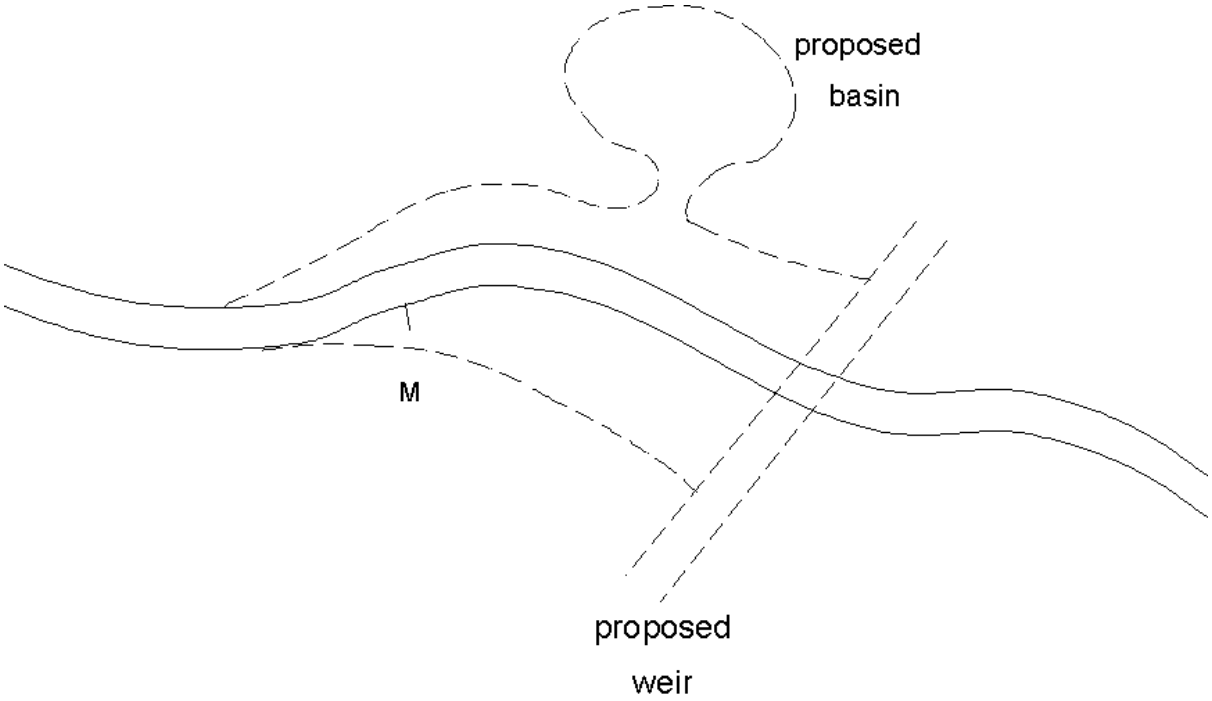


Figure 6.4. plan of river with proposed flood prevention structure

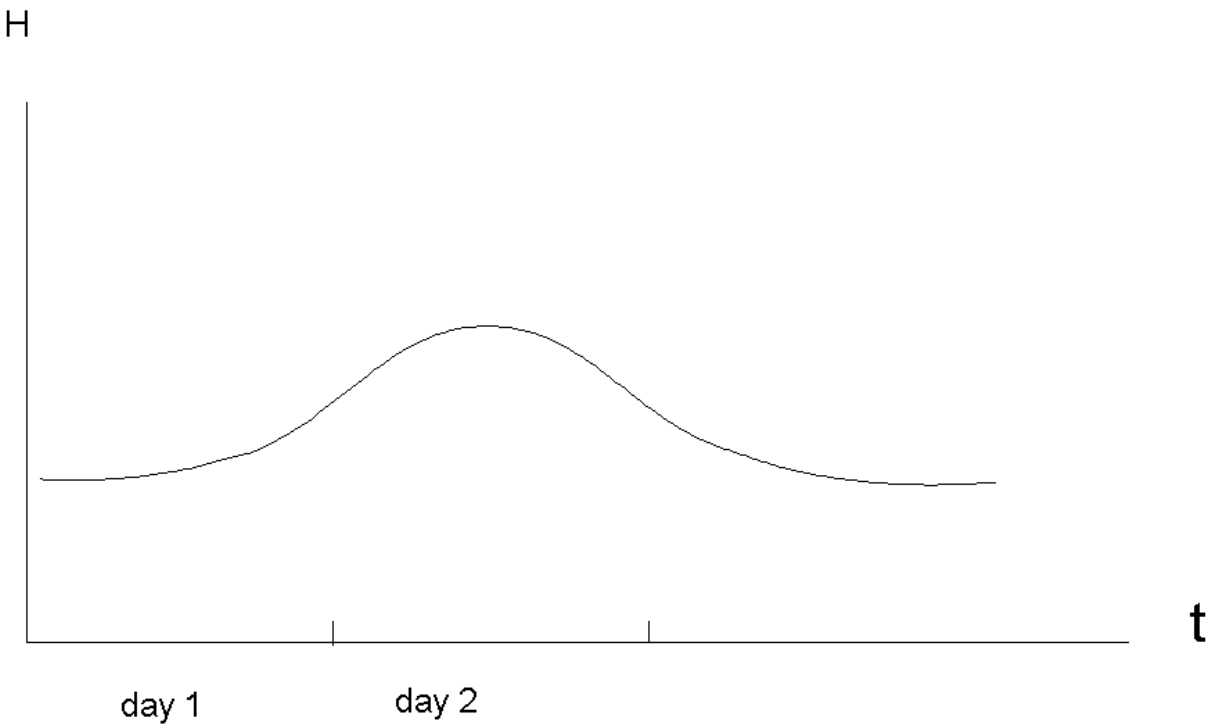


Figure 6.5. measurement of a flood taken at M (see figure 6.4)

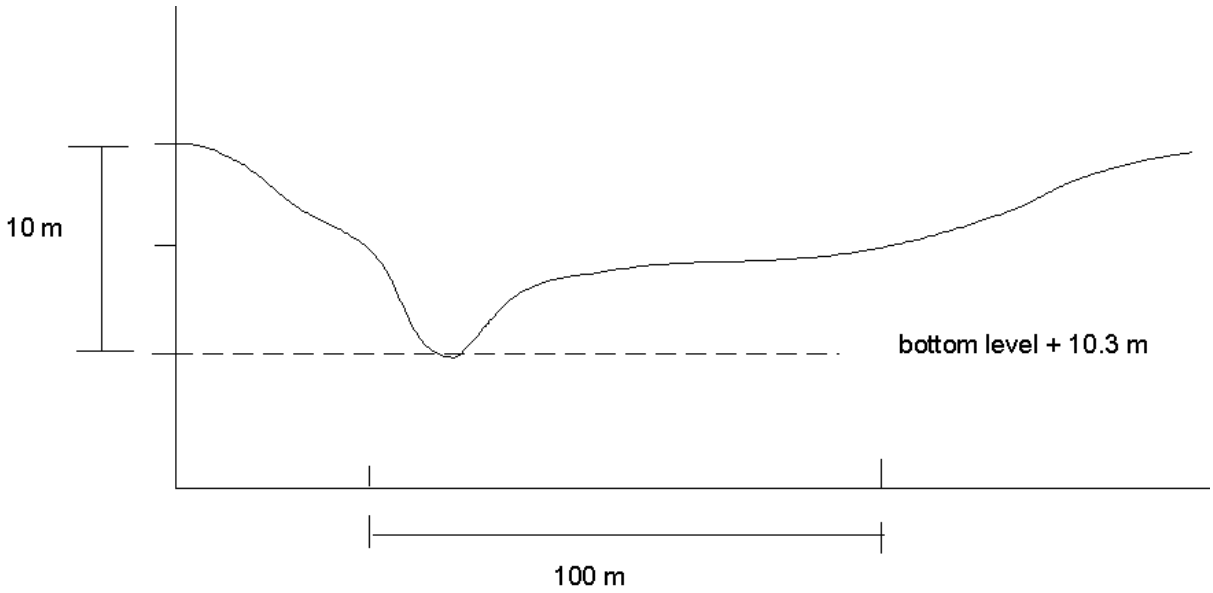


Figure 6.6. cross-section of the river

6.2. Validation, calibration and verification

The various steps in the modeling process need verification. Nowadays "quality assurance" is a popular term meaning that the result of a piece of work must conform in a verifiable manner to specified or generally accepted quality standards. Some commissioning agencies will specify accuracy requirements for numerical simulations but more often than not this is not done or only in very vague terms. This does not mean that the model engineer can do what he likes; the profession or society in general has made up regulations in view of the engineer's great responsibility. If there is a legal conflict it is important that the engineer can show that he has worked with state-of-the-art models and standards.

The first step in the modeling process is the choice of the type of model; a complete proof that the right choice has been made cannot be given. What can be done is: make a simulation computation without any tuning of model coefficients, showing whether the model is able to predict the phenomena which one is interested in. We call this a validation run.

This simulation also gives an indication of the predictive value of the model. The predictive value is low if a lot of tuning is necessary in order to get acceptable results. Moreover, if a lot of tuning has to be done it is doubtful whether the model will be able to give a good prediction of the design situation. With the present state of knowledge for instance ecological models have a poor predictive value because too little is known about the interaction processes between ecological components.

After successful validation the more tedious work of calibrating the model starts. Calibration is usually necessary because the validation runs do not yet provide results with the prescribed accuracy, and because almost any model has some coefficients whose values are not fixed but dependent on the circumstances. In the shallow water equation the friction coefficient is a well-known example, in the transport equation the diffusion coefficient.

Traditionally calibration is a tedious trial-and-error process whereby coefficients are varied and model runs made until the results of the runs are in sufficiently good accordance with the measurements. The process can be automated to a certain extent (Booij and Holthuijsen, 1988).

It is good procedure to use only part of the available measurements for calibration. The other part is used later on for verification. Verification is the process to prove that the model indeed predicts with the prescribed accuracy. Obviously in the verification we should not use the same measurements that were used for calibration. The part of the measurements that is used for calibration should be selected **at random** from the total set of measurements.

If measurements have to be planned our numerical model can be useful too. Two things have to be established:

A. Which of the model coefficients need to be determined more accurately; there are two sub-questions to be considered:
    1. How large (roughly) is the range of uncertainty of each coefficient?
    2. How large is the influence of each coefficient on the quantity of interest? in other words: how sensitive is the quantity of interest to the value of each coefficient?

B. What quantities can be measured best to determine the coefficients selected under A? Here also there are two sub-questions:
    1. What quantities can be measured with available facilities, i.e. financial means, available equipment and personnel etc.? Note that for instance the measurement of a discharge is much more costly than measurement of a water level.
    2. Which of the quantities are influenced by the coefficients that we want to determine? Note that if a coefficient does not have any influence on a measured quantity we cannot expect to get information on this coefficient from the measurement.

Twice in this procedure we need a **sensitivity analysis**. There are many other situations where sensitivity analysis is useful, for instance if we want to find out whether a boundary condition influences the quantity of interest. Sensitivity analysis consists of systematical variation of model coefficients and comparison of the results of the computations.

It is assumed that for each coefficient we know its range of uncertainty, i.e. the interval within which the true value of the coefficient must be found. Another way to characterize the coefficient is by its reference value, a value somewhere in the middle of the interval (the most likely or most common value), and the deviation, say half the length of the interval.

The most simple procedure to carry out sensitivity analysis is to make one model run with reference values for all the coefficients, and one model run in which each time one coefficient is varied with a fraction of its deviation. This provides us with two kinds of information: how much the total deviation of a dependent variable is likely to be, and how much each of the coefficients contributes to this deviation. This procedure has been automated by Booij and Holthuijsen (1988).

Exercise 6.B:

Consider the system described in exercise 6.A. Try to calibrate the upstream boundary condition such that the measurements in point M are reproduced with acceptable error (say a few cm).

Exercise 6.C:

Consider a tidal river (width and bottom level are shown in figure 6.7). At place A along the channel a discharge of effluent is planned and for two places B and C one wants to know the maximum concentrations. To get a feeling for the values of these maximums a sensitivity analysis with the program DUFLOW (Spaans et al., 1989) is to be carried out.
The tidal range at the seaward boundary S, the river inflow at the upstream boundary R where the tidal influence is negligible, and the friction coefficient are given (see table 6.1).

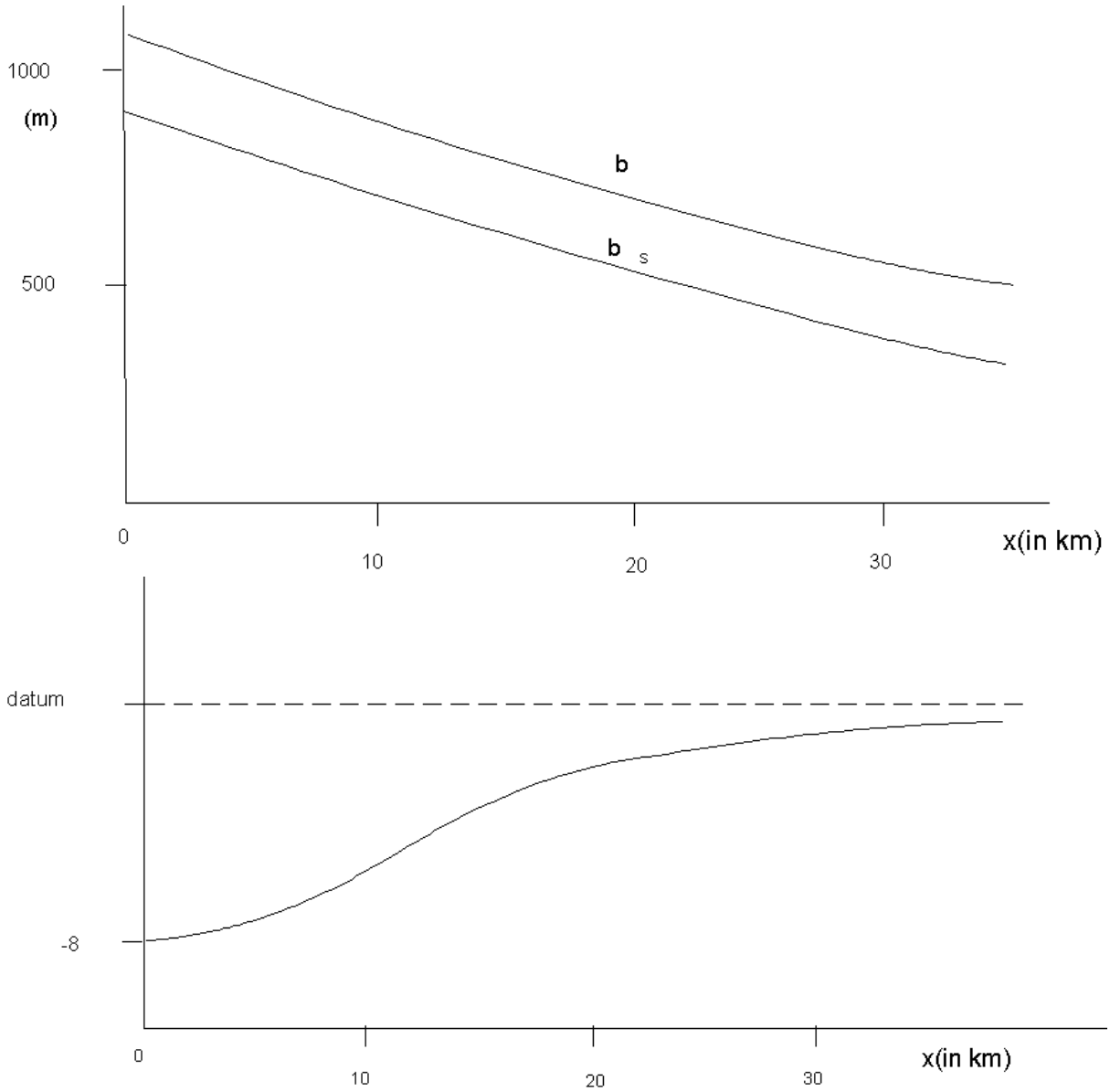


Figure 6.7 Reference values of cross-sectional parameters (upper panel: b=width of storage, b_s=width of flow, lower panel: z_b=bottom level) as function of axial coordinate x.

Table 6.1. Reference value and deviation of a number of model coefficients.

Amplitude of water level H at S (m)	*1.2*	*0.2*
Average of H at S (m)	*0.0*	*0.1*
Upstream discharge at R (m^3/s)	*2000*	*700*
Friction coefficient c_{fr} (-)	*0.004*	*0.001*
Diffusion coefficient D (m^2/s)	*200*	*100*

The deviation of the bottom level is 0.2 m, the deviation of the width of flow is 5% of the reference value, and the deviation of the width of storage is 10% of the reference value.

It is perfectly all right to attempt the analysis with different values.

6.3. Discretization errors

Discretization errors are caused by the finiteness of the the steps Δx and Δt. The parameters relevant for the discretization error are dimensionless ratios of Δx and the various length measures, for instance wavelength, and Δt and the time measures occurring in the problem.

It is stressed that there is no need to calculate Δx and Δt accurately. It is sufficient to know them within a factor of 2.

The general rule is that a step size must be sufficiently small to give a good description of the phenomenon. In the case of the shallow water equation Δx must be small compared with the wavelength but also small enough to follow the changes in cross-section of the channel. In practice the spatial step size is determined first (based on accuracy considerations), and then the time step is chosen (based both on stability and accuracy requirements).

If the wavelength is the dominant factor a first guess of Δx could be 0.01 to 0.02 times the wavelength; similarly for Δt a first guess could be 0.01 to 0.02 T. If damping, characterized by negative exponential behaviour, is dominant a first guess could 0.05 to 0.01 times the relaxation time.

Furthermore the stability conditions if applicable must be observed. They usually take a form such as

$$C\frac{\Delta t^m}{\Delta x^n} < p$$

where C is some physical constant (propagation velocity, diffusion coefficient or combinations) and p is a real constant of order 1.

Knowledge of the physics is essential because the physics determines things like wavelength, wave period, damping and propagation velocity.

After Δx and Δt have been estimated the accuracy of the computation can be determined experimentally using sensitivity analysis with Δx and Δt. Here it is important to know the truncation error of the numerical scheme. Let us assume that the truncation error is of the order Δt^n. Then, assuming that the computation is stable, the error in the results will be $K.\Delta t^n$. The factor K is not known; to determine it we need to compare two computations with different time step; for instance we make one computation with Δt and one with $\Delta t/2$. The errors will be $K.\Delta t^n$ and $K.\Delta t^n/2^n$; the difference provides a good estimate of the accuracy of the computation.

If the error is influenced by both Δx and Δt, it must first be established which of the two is dominant. If the error is of lower order in Δt than in Δx, it is Δt which must be varied. If the order is the same in both step sizes, both must be varied.

6.3.1. Errors in propagation problems

In section 4.3.2 we have seen that for the pure propagation problem the semi-discretized model using the mid-point rule gives the following relative propagation velocity:

$$v_r = \frac{\sin\xi}{\xi}$$

Exercise 6.D:

In the same way derive that the relative propagation velocity for the trapezium rule is:

$$v_r = \frac{tg(\xi/2)}{\xi/2}$$

Hint: if you are not able to solve this problem, look first at the next paragraph.

Exercise 6.E:

In the same way derive that the relative propagation velocity for the first order upstream scheme is:

$$v_r = \frac{\sin\xi}{\xi}$$

It is possible also to derive the relative propagation velocity for a semi-discretized model in which the discretization with respect to time has already taken place, but the discretization with respect to space has not. We will show how this works out for the θ-rule.

The function $c^n(x)$ represents $c(x,t)$.

Such a semi-discretized model for the simple wave equation would read:

$$\frac{c^{n+1} - c^n}{\Delta t} + v(\theta \frac{dc^{n+1}}{dx} + (1-\theta)\frac{dc^n}{dx}) = 0$$

Again we consider the initial value problem in which

$$c^o = c_0 e^{ikx}$$

Thus the complex amplification factor r must be found from:

$$\frac{r-1}{\Delta t} + ikv(\theta r + 1 - \theta) = 0$$

or:

$$r = \frac{1 - i(1-\theta)\mu}{1 + i\theta\mu},$$

$$\text{where: } \mu = kv\Delta t = \frac{v\Delta t}{\Delta x}.k\Delta x = \sigma\xi$$

Then the complete time-dependent solution is:

$$c_0 r^n e^{ikx} = c_0 r^{t/\Delta t} e^{ikx} =$$

$$c_0 e^{\ln(r).t/\Delta t + ikx} = c_0 e^{\ln|r|.t/\Delta t + iarg(r).t/\Delta t + ikx}$$

From this it is seen that the amplitude decreases as:

$$e^{\ln|r|.t/\Delta t}$$

The phase of the complete solution is

$$iarg(r).t/\Delta t + ikx$$

So in the numerical model the propagation velocity is

$$\frac{-arg(r)}{k.\Delta t}$$

Thus the relative propagation velocity is equal to

$$\frac{-arg(r)}{\mu}$$

so that we find for the θ-rule:

$$v_r = \frac{arctg((1-\theta)\mu) + arctg(\theta\mu)}{\mu}.$$

As should be expected we see that the propagation error depends on the number of time steps per wave period, which is represented here by μ.

Exercise 6.F:

In the same way derive that the relative propagation velocity for the mid-point rule (in t) is:

$$v_r = \frac{\arcsin \mu}{\mu}$$

The amplitude error, represented by the numerical damping over one wave period can also derived from r, it is equal to

$$d = |r|^{T/\Delta t}.$$

For the θ-rule this amounts to

$$d = \left(\frac{|1 - i(1-\theta)\mu|}{|1 + i\theta\mu|} \right)^{2\pi/\mu}$$

For the mid-point rule and for the trapezium rule we find that the amplitude error of the semi-discretization with respect to time is zero.

Most of the popular numerical schemes we can derive using combinations of the above semi-discretizations in x and t; for such schemes we can calculate the error (approximately) by adding the errors incurred by the two semi-discretizations. This holds for the damping error as well as for the propagation error. Remember that the (relative) error in the propagation velocity is $v_r - 1$. For example for the Crank-Nicholson scheme (mid-point in x, trapezium rule in t) we find the following relative propagation velocity:

$$v_r = \frac{\sin \xi}{\xi} + \frac{2arctg(\mu/2)}{\mu} - 1$$

The above result is only an approximation; it is valid if each of the errors that we have added are small.

Exercise 6.G:

Make a table stating for each of the following numerical schemes from which semi-discretizations in x and t it can be thought to be constructed: Box scheme, Leap-frog scheme, First order upwind scheme.

Exercise 6.H:

In the same way as above derive the amplitude error for the Crank-Nicholson scheme.

Using the methods described above we can approximately calculate the propagation error and the amplitude error once we know the wavelength and the wave period and the space and time steps, if the problem we are dealing with is primarily a propagation problem. This is true for the shallow water equation (both in the tidal regime and in the upper river regime) and for the transport equation, also if a diffusion term is present; usually the effect of the diffusion term on the numerical error is small compared with the influence of the advection term.

The following section will present an application of the method.

6.3.2. Determination of wavelength and period (shallow water equation)

When studying non-stationary flow the wavelength or the period is determined by (a) the boundary conditions, (b) manipulation of flow control structures, (c) forcing or (d) the geometry. In flow problems the wave period is more or less the same in the whole computational region.

In tidal problems and in flood waves in rivers the wave period T is set by the boundary condition; the wavelength must be derived from it. If forcing for instance by the wind, is dominant it is also the wave period which is determined by the time-variation of the forcing.

Sometimes the period is not determined by natural phenomena but by human interference; examples are the switching on of pumping stations, hydro-power plants, opening of sluices etc.

In all these cases the wavelength $L = v.T$ where v is the propagation velocity for that wavelength. v also depends on the flow conditions, primarily the depth. The depth can vary strongly for instance in tidal basins and in sewer systems. Estimating v is difficult in that case, so it is advisable to stay on the safe side when estimating step sizes.

In cases of oscillations, for instance in harbours and lakes, it is the wavelength which is determined. The wavelength is related to the dimension of the system. In case of harbour oscillations it is 4 times the legnth of the harbour, in case of a lake 2 times the length of the lake. If the system is not a single channel the wavelength cannot be calculated so easily but it is still related to the size of the system; a rough estimate is good enough to determine the step sizes.

In cases of oscillations the depth does not vary strongly so v can be found with little uncertainty.

When we consider the stability of the shallow-water computation we must take into account all possible frequencies, i.e. from 0 up to $2\Delta x$. Therefore in the stability condition we must use the largest propagation velocity which is $|U| + \sqrt{gA/b}$.

This is in contrast with the accuracy consideration where v is the velocity of the dominant wave.

6.3.3. determination of the wavelength in transport computations

In case of long waves the wave period can be taken to be roughly the same in the whole system. This is different in the transport equation. Due to the presence of the diffusion term a "cloud" of pollutant will spread in time. The standard solution in this case is given by equation (2.12). Since this solution is not a sinus we cannot exactly assign a wavelength; however by fitting a sinusoidal function to expression (2.12) we find that the wavelength is approximately $4\sqrt{Kt}$. Although an arbitrary "cloud" of pollutant will not have a shape according to (2.12) it will after some time develop to a shape more resembling (2.12).

example

Assume that at a point P on a river a conservative toxic substance has been released during 1 hour. The transport velocity in the river is 0.6 m/s and the diffusion coefficient is 100 m^2/s.
A transport computation is to be made to compute the concentration of the pollutant at a place Q, 60 km downstream of P. Since the pollutant will be spread when reaching Q the relevant wavelength or period will be larger than at P.

The procedure is as follows: first we estimate the initial wavelength and corresponding time t; then we add the travel time from P to Q; from the resulting time we calculate the wavelength at Q and the corresponding wave period.

*Since the material was released over 1 hour the "cloud" length will be 3600*0.6=2160 m. The corresponding wavelength (see figure 6.8) is twice this value, i.e. 4320 m. If this is $4\sqrt{Dt}$ t must be $t=(4320/4)^2/100=11664$ s. The travel time between P and Q is 60000/0.6=100000 s, so that the wavelength at Q is $4\{100*(11664+100000)\}^{1/2}$ = 13366 m, and the wave period at Q is obtained by dividing the wavelength by the propagation velocity, i.e. 13366/0.6 = 22277 s, or appr. 6 hr.*

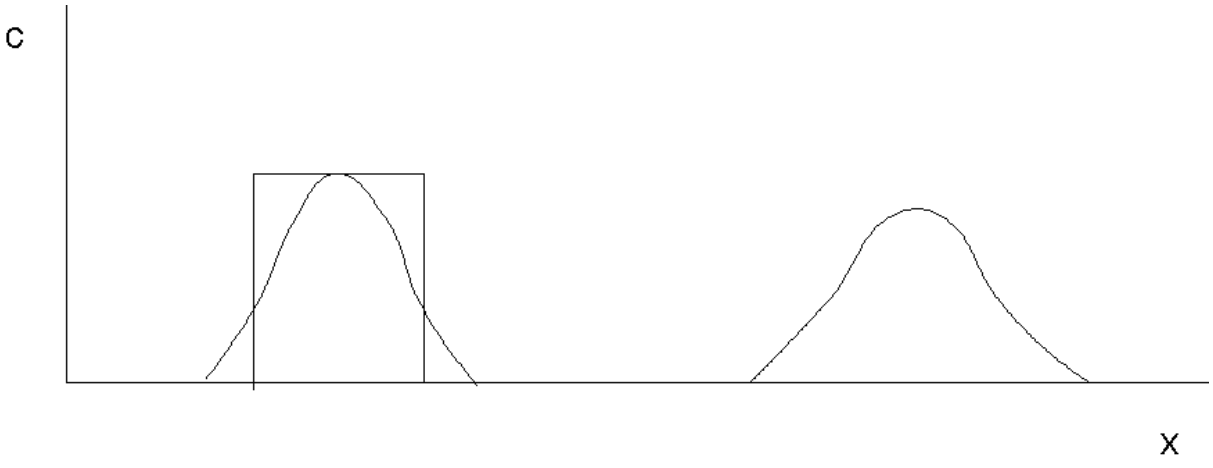


Figure 6.8. Initial distribution of a material and distribution after a certain propagation time.

6.3.4. Estimating space and time step

The step sizes Δx and Δt can be determined more accurately than in section 6.4.1 using graphs or tables of propagation factors. In chapter 4 some of such graphs are presented. There are graphs for the relative propagation velocity v_r, i.e. the propagation velocity in the numerical model divided by the propagation velocity in the analytical model, and for the damping factor d, i.e. the amount of damping of a wave which has propagated in the numerical model over one wave period. These graphs have been derived for the linear simple wave equation (2.13). In this analytical model the propagation velocity is constant and the wave damping is zero.

When using the graphs for the shallow water equation we neglect the nonlinearity and the non-uniformity of the propagation velocity. In using the graphs for the transport computation we neglect the effect of the diffusion term. In both cases we still may expect a reasonable estimate for the step sizes.

Both v_r and d are given in the graphs as function of the number of points per wavelength/period, or of their inverses $\Delta x/L$ or $\Delta t/T$, resp. There are a number of curves in the graph each for a certain Courant number.

Using the graph we are able to find v_r and d for a combination of Δx and Δt. In practice however accuracy requirements are not formulated in terms of v_r and d, but in terms of a percentage of a physical quantity or an absolute error in a quantity such as Q, U, H or C.

The procedure to be followed is: translate the accuracy requirements into conditions for v_r and d; then choose a combination of Δx and Δt fulfilling the conditions. There is not a single solution but a set of solutions, so a reasonable selection must be made for instance one which minimizes the computational effort. Often the space step is constrained by the wish to get a good representation of the geometry, and the time step is then constrained by accuracy and/or stability requirements.

example:

A city R located on a tidal river 80 km from the sea wishes to compute water levels due to tides and storm surges with an accuracy of 5 cm.

Data: amplitude of the tide during spring-tide at the seaward boundary: h_m=5 m, tidal period T=45000 s, width ($b=b_s$) of the river 500 m, average depth 10 m, c_{fr}=0.004.

Select step sizes fulfilling the requirements if the program to be used is based on the box scheme.

Solution: the error can be due to error in propagation velocity or to numerical damping. Since the two errors do not attain their maximum at the same time, it seems safe to choose both errors to be at most 3 cm.

The error due propagation velocity error (illustrated by figure 6.9) can be estimated at:

$$h_m \frac{2\pi t}{T} |1 - v_r|$$

where t is the time the wave needs to travel from the sea to R. For a tidal wave the propagation velocity is not much less than $\sqrt{gA/b} = 10m/s$; so t = 80000/10 = 8000 s. Thus:

$$|1 - v_r| < \frac{0.03}{5} \frac{45000}{2\pi.8000} = 0.005$$

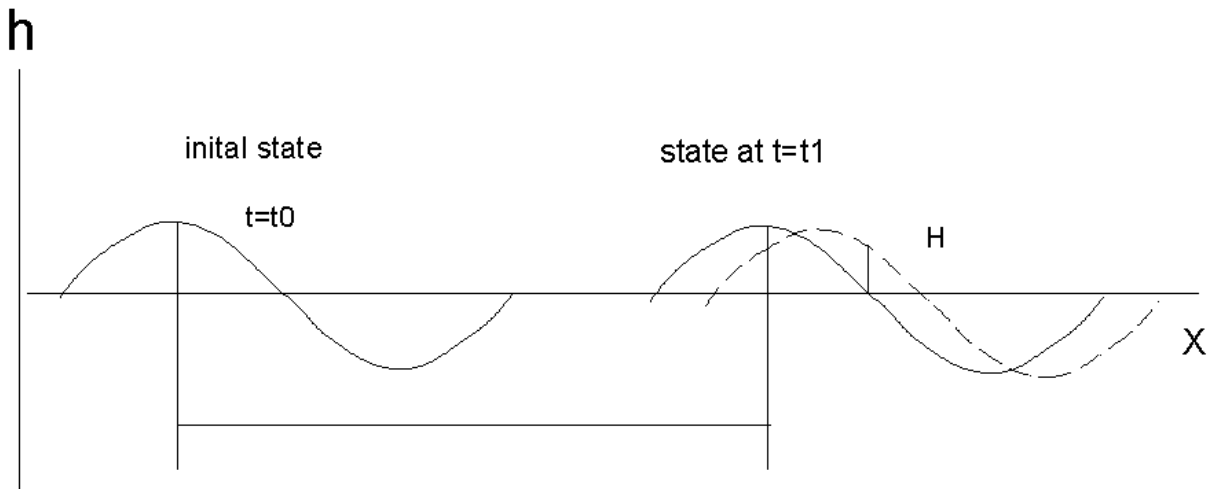


Figure 6.9. Error in computed water level due to error in propagation velocity. The continuous line represents the exact solution, the dashed line the numerical solution, both after the same propagation time. The distance travelled during propagation is indicated in the figure, it is equal to $c(t_1-t_0)$..

The error due to damping also depends on the travel time of the wave; after time t the amplitude of the wave (in the numerical model) has decreased to $d^{t/T}$; this is in addition to the physical damping (due to bottom friction). In reality the amplitude at R will be smaller, say ah_m, and in the numerical model $ad^{t/T}h_m$ (see figure 6.10). The error is

$$a\,|\,1-d^{t/T}\,|\,h_m$$

this must be smaller than 0.03 m. We can try to estimate a from an analytical solution of the shallow water equation; it is simpler to assume $a = 1$ which is on the safe side. Thus:

$$|\,1-d^{t/T}\,|=|\,1-d^{8000/45000}\,|<\frac{0.03}{5}=0.006$$

or:

$$0.967 < d < 1.03$$

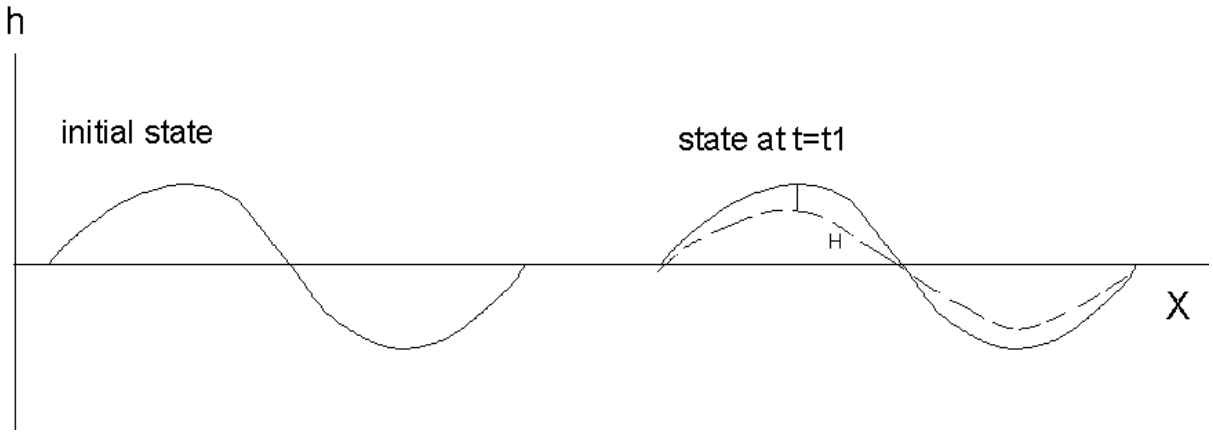


Figure 6.10. Error in computed water level due to numerical damping. The continuous line represents the exact solution, the dashed line the numerical solution, both after the same propagation time.

When choosing Δx and Δt we need to know the numerical scheme which has been chosen, or we have to choose one ourselves. Here we choose the box scheme with $\theta=0.55$ (the standard option in DUFLOW). We know that the amplitude error due to the semi-discretization in x is zero, the amplitude error due to Δt is approximately (for small values of μ): $1-2\pi\mu(\theta-\frac{1}{2})$

So we require $2\pi\mu(\theta-½)<0.03$ *or* $\mu<0.09$

The propagation error due to Δt is approximately: $\mu^2/6$

and the propagation error due to Δx is approximately: $-\xi^2/6$.

The contributions due to Δx and Δt have opposite sign, so we remain on the safe side if we require that the absolute value of each is smaller than 0.005. It follows that $\xi<0.17$ and $\mu<0.17$.

so the following choice seems to fulfill the accuracy requirements:

$$\Delta x = 12000m \quad \Delta t = 600\ s$$

The time step seems acceptable, the spatial step is too long to give a good representation of the changes in profile along the river, so 5000 m seems to be better.

It is always necessary the check experimentally (by varying the step sizes) whether the accuracy is sufficient indeed; remember that a considerable number of simplifications have been made to arrive at the estimated step sizes.

Exercise 6.I:

Use the program DUFLOW to confirm whether or not in the above case the accuracy is as expected by making two computations with different values of time and space step.

Exercise 6.J:

For a river with dimensions as in example 6.A and using the measurement in M as upstream boundary condition the water level at a place 100 km downstream of M must be computed. The accuracy requirement is that the maximum water level must be computed with an error of not more than 10 cm. Choose appropriate values for time and space step assuming the same numerical scheme is chosen as in the previous example.

Exercise 6.K:

Use the program DUFLOW to confirm whether or not in the above exercise the accuracy is as expected by making two computations with different values of time and space step.

As explained before a transport computation needs data from a flow computation. In some computer programs different space and time steps can be used in the flow and the transport computation; in others the same steps have to be used. In the latter case a combination of Δx and Δt must be chosen which fulfills the requirements of both computations.

6.4. What to do if something goes wrong

First and most important rule for users of computer programs: Never trust the results blindly, also if they look nice and smooth. Check whether they conform to your expectations, and to known behaviour of the physical system. Remember that computer programs are awfully complicated, and that many combinations of options are not tested properly, simply because there too many of such combinations.

If you do get results that you do not trust the cause may one of the following:

1. The equation (partial differential equation) is not a good description of reality in your case;
2. The partial differential equation may be incompletely represented in the computer program;
3. The initial and boundary conditions and/or the differential equation are in disagreement;
4. There may be an error in the computer program;
5. The numerical approximation may be inadequate.

You should first try to find the cause and then look for remedies. In the sequel we try to give hints for finding causes and for possible remedies.

1: Good thinking helps; most other things don't. Carefully read the literature on the equation that is used in your program, and check whether it seems applicable in your problem. If you have access to it try another more powerful computer program (more powerful in the sense that it is valid in a broader range of physical conditions).

2: Sometimes terms in the partial differential equation have been deleted when developing the program; for instance in the first generation of programs for the shallow water equation often the advective terms were neglected. If you expect such a cause take the results of the computation and evaluate the various terms including the ones neglected at various points in the domain; check whether the neglected terms are indeed negligible.

3: It may happen that you specified initial and/or boundary conditions which have a discontinuity at the transition point. Some numerical approximations cannot cope with such a discontinuity; as a result instability or other oscillatory behaviour may develop. Study the output and find out whether the undesired behaviour started at a boundary.

4: This is not very likely if you use a program that is used by many people in many places, but it does happen every once in a while. Before attacking the developer of the computer program make sure that you did read the instructions carefully, and that the errors are not caused by points 1, 2, 3 or 5. When you contact the developer of the program collect as much output concerning the problem, as you possibly can. Together with the developer you may be able to find a way around the error. If this developer is a sensible person he/she will be grateful.

5: Numerical problems are the most common of cause of erroneous results. Usually they can be recognized because the results are "jumpy" on the scale of the individual space or time steps, i.e. wavelengths of $2\Delta x$, $3\Delta x$ etc. or wave periods of $2\Delta t$, $3\Delta t$ etc. are observed.

In order to see whether you really have jumpy results you need output at every space and time step. Normally you would request output at larger intervals, for instance every half hour if you are running a tidal problem, while the time step is only 5 minutes. Re-run the program with output requested at every time step, and check whether the results in various points are jumpy in time.

Another way to determine whether the erroneous results are due to numerics is: change space and time steps (take for instance half of their original value) and see whether the results change more than a few percents. If there is little or no change the results are apparently not sensitive to the numerics; then you must look for other causes.

If numerical approximations are not accurate or not stable enough, possible remedies are: smaller time steps and/or smaller spatial steps, larger θ coefficient, usage of another more stable numerical scheme (if there is such an option in the program you are using).

Exercise 6.L:

Take the model described in exercise 6.C (only the flow model, not the transport model), and deliberately make Δt very large. Study the behaviour of the results.

Exercise 6.M:

Take the model described in exercise 6.C (only the flow model, not the transport model), and deliberately generate a discontinuity between initial and boundary condition. Study the behaviour of the results.

References

General:

Hofstadter, D.R. (1979), Gödel, Escher, Bach: an Eternal Golden Braid, Basic Books, New York.

Simon, H.A. (1969), The Sciences of the Artificial, The MIT Press, Cambridge, Massachusetts.

General Mathematics:

Almering, J.H.J. et al., Analyse, D.U.M. Delft 1990.

Garabedian, P.R. (1967), Partial Differential Equations, Chelsea Publishing Company, New York.

Roman, P. (1975), Some Modern Mathematics For Physicists and Other Outsiders, Volumes I and II, Pergamon Press Inc., New York.

Yosida, K. (1965), Functional Analysis, Academic Press, New York.

Fourier analysis:

Hsu, H.P., Fourier Analysis, Simon and Schuster, New York 1970

Box models and their applications:

Forrester, Jay W. (1961), Industrial Dynamics. M.I.T. Press, Cambridge U.S.A.

Goodman, M.R. (1974), Study notes in System Dynamics. Wright-Allen Press, Cambridge U.S.A.

Meadows, D.L., W.W. Behrens, D.H. Meadows, R.F. Nail, J. Randers and E.K.O. Zahn (1974), Dynamics of growth in a finite world. Wright-Allen Press, Cambridge U.S.A..

Richardson, G.P. and A.L. Pugh (1981), Introduction to System Dynamics Modelling with DYNAMO. M.I.T. Press, Cambridge U.S.A.

General Fluid Mechanics:

Batchelor, G.K. (1967), An Introduction to Fluid Dynamics, Cambridge University Press, Cambridge.

Shallow water equations:

Dronkers, J.J. (1964), Tidal computations in rivers and coastal waters, publ: North Holland.

Mahmood, K. and V. Yevyevich (1975), Unsteady flow in open channels, Vol. I-II. Fort Collins, Water Resources Publ.

Spaans, W., N. Booij, N. Praagman, R. Noorman and J. Lander, (1989), DUFLOW, a micro-computer package for the simulation of one-dimensional unsteady flow in open channel systems, version 2.0. Bureau ICIM, PO Box 5809, NL-2280 HV Rijswijk, The Netherlands.

Ven te Chow, P.D. (1983), Open-channel Hydraulics.

Mc Graw-Hill.

Environmental modelling:

Fischer, H.B., E.J. List, R.C.Y. Koch, J. Imberger, N.H. Brooks (1979), Mixing in inland and coastal waters. Academic Press, New York.

Orlob, G.T. (1983), Mathematical modeling of water quality: streams, lakes and reservoirs. John Wiley & sons.

Somlyody, L. and G. van Straten (1986), Modelling and managing shallow lake eutrophication. Springer Verlag.

R.V. Thomann and J.A. Mueler (1987), Principles of Water Quality Modelling and Control, Harper Collins Publishers, New York.

Computational Hydrodynamics:

Abbott, M.B. (1979), Computational Hydraulics, elements of the theory of free surface flow. Pitman, London.

M.B. Abbott and D.R. Basco (1989), Computational Fluid Dynamics; an Introduction to Engineers, Longman.

Chow, C.Y. (1979), An introduction to computational fluid mechanics. Wiley, New York.

Cunge, J.A., F.M. Holly and A. Verwey (1980), Practical aspects of computational river hydraulics. Pitman, London.

C. A. J. Fletcher (1988), Computational Techniques for Fluid Dynamics, Volumes 1 and 2. Springer

C. Hirsch (1991), Numerical computation of internal and external flows, Volumes 1 and 2. John Wiley and Sons, New York.

Patankar, S.V. (1980), Numerical heat transfer and fluid flow. Hemisphere-Mc Graw Hill.

Peyret, R. and T.D. Taylor (1983), Computational methods in fluid flow. Springer-Verlag, Berlin.

Potter,D. (1973), Computational Physics, John Wiley and Sons, New York.

Roache, P.J. (1976), Computational fluid dynamics. Hermosa Publ., Albuquerque, New Mexico.

Vreugdenhil, C.B. (1989), Computational Hydraulics, an introduction. Springer-Verlag, Berlin.

Numerical Analysis:

Ames, W.F. (1977), Numerical Methods for partial differential equations. Nelson, London.

R. Courant, K. Friedrichs and H. Lewy (1928), Uber die partiellen Differenzen Gleichungen der mathematischen Physik, Mathematischen Annalen, vol. 100, pp. 215-234

C.W Gear (1971), Numerical Initial Value Problems in Ordinary Differential Equations, Prentice Hall.

S.K. Godunov and V.S. Ryabenki (1964), Theory of Difference Schemes, North Holland Publishing Company, 1964

G.H Golub and C. F. Van Loan (1983), Matrix computations. North Oxford Academic, London.

G.K Gupta, R. Sacks Davis and P.E. Tischer (1985), A review of Recent Developments in Solving ODEs, Computing Surveys, vol. 17,pp. 5-47.

B. Gustafson (1975), The Convergence Rate for Difference Approximations to Mixed Initial Boundary Value Problems, Mathematics of Computation, V26, pp. 396-406.

P. Henrici (1962), Discrete Variable Methods in Ordinary Differential Equations, John Wiley and Sons.

C.W. Hirt (1968), Heuristic Stability Theory for Finite Difference Equations, Journal of Computational Physics, vol. 2, pp. 339-355.

J.D. Lambert (1990), Numerical Methods for Ordinary Differential Systems, John Wiley and Sons, New York (or: J.D. Lambert (1976), Computational methods in ordinary differential equations. John Wiley and Sons, New York).

H. Lauwerier (1989), Fractals, Aramith uitgevers.

J.J.H. Miller (1971), On the location of zeros of certain classes of polynomialswith application to numerical analysis, J. Int. Math. Appl., vol. 8, pp. 397-406.

Mitchell, A.R. (1977), Computational methods in partial differential equations. Wiley, New York.

A.R Mitchell and D.F. Griffiths (1980), The Finite Difference Method in Partial Difference Equations, John Wiley and Sons.

W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling (1988), Numerical Recipes. Cambridge University Press

Richtmyer, R.D. and K.W. Morton (1967), Difference methods for initial value problems, Interscience Publishers, New York.

P.J. Roache (1972), Computational Fluid Dynamics. Hermosa Publishers, Albuquerque, New Mexico.

Th. L. van Stijn, J.C.H. van Eijkeren, N. Praagman (1987), A comparison of numerical methods for air-quality problems. KNMI report WR nr 87-6 or RIVM report nr. 958702007

H. van der Vorst (1988), Parallel rekenen en super computers, Academic service.

R.F Warming and B.J. Hyett (1974), The Modified Equation Approach to the Stability and Accuracy Analysis of Finite Difference Methods, Journal of Computational Physics, vol. 14, pp. 159-179

Sensitivity analysis, Calibration:

Beck, M.B. (1983a). A procedure for modeling, in: Mathematical modeling of water quality: streams, lakes and reservoirs, ed. by: G.T. Orlob. John Wiley & sons.

Beck, M.B. (1983b). Sensitivity analysis, calibration and validation, in: Mathematical modeling of water quality: streams, lakes and reservoirs, ed. by: G.T. Orlob. John Wiley & sons.

Booij, N. and L.H. Holthuijsen (1988), The statistical analysis of deterministic model results, in: Computer modelling in Ocean engineering, ed. by: B.A. Schrefler and O.C. Zienkewicz. Balkema, Rotterdam.

Meyer, W.J. (1984), Concepts of Mathematical Modeling, Mc Graw Hill, New York.

List of symbols

A_b	total wetted cross-section of the channel
A_s	flow cross-section of the channel
c	concentration or an unknown function in general
C	concentration
d	depth
g	acceleration due to gravity
h	water level with respect to a horizontal plane of reference
i	(unless used as sub or superscript) imaginary constant
k	(unless used as sub or superscript) spatial wavenumber = $2\pi/L$
K	diffusion coefficient
L	wavelength
Q	discharge (i.e. volume of water passing a cross-section per unit time)
r	(complex) amplification factor
R	Reynolds number
t	time
T	wave period
u	particle velocity of the water, in x-direction
v	propagation velocity
v_r	relative propagation velocity (numerical propagation velocity divided by exact propagation velocity)
x	horizontal coordinate, usually along channel axis
Δt	time step
Δx	spatial step
ξ	non-dimensional spatial stepsize = $k\Delta x$
θ	time weighting coefficient in numerical schemes
λ	eigenvalue of a matrix
σ	Courant number = $v\Delta t/\Delta x$
π	circular constant = 3.14...
ω	(angular) frequency = $2\pi/T$

Appendix A. Fourier series

A.1. complex exponential function

Any **linear** differential equation with **constant coefficients** can be solved by substituting an exponential function. If we deal with an ordinary differential equation with for instance t as the independent variable we can substitute e^{rt} as solution.

example: diff.eq. $A\frac{dy}{dt} + By = 0$ has the solution $e^{-At/B}$.

In second and higher order equations r often turns out to be complex. We obtain a real solution by simply taking the real part (or the imaginary part) of the complex solution. This is correct if the coefficients in the differential equation are **real**, as they obviously are in a physically relevant equation.

We will use the well-known relation: $e^{i\psi} = \cos\psi + i\sin\psi$ (A.1)

In a complex solution we can recognize a **damping** effect (related with the real part of r, i.e. r_1) and an **oscillatory** effect (related with the imaginary part of r, r_2):

$$ae^{rt} = |a|e^{i\arg(a)} \cdot e^{r_1 t + ir_2 t} = |a|e^{r_1 t}e^{i(r_2 t - \arg(a))}$$

The real part of this expression is

$$|a|e^{r_1 t}\cos(r_2 t - \arg(a))$$

Exercise:

The differential equation for a damped mass-and-spring system is

$$m\frac{d^2y}{dt^2} + w\frac{dy}{dt} + ky = 0$$

substitute the general complex solution $y=ae^{-rt}$, find the value of r, and determine the real part of the solution.

In partial differential equations we can do the same. Let x and t be the independent variables. Into any **linear** partial differential equation with **constant coefficients** we can substitute:

$$e^{rt-px}$$

Very often in the cases that we consider in this book p is purely imaginary and then we write:

$$e^{rt-ikx}$$

where k is called the wave number. Sometimes also r is purely imaginary and then we write:

$$e^{i\omega t-ikx}$$

The real part of this is:

$$\cos(\omega t-kx)$$

which represents a progressive wave as can be seen by making a graph of this function for two values of t a small interval Δt apart. We see that the function values we get remain completely the same if we consider values of x a

distance $\Delta x=\omega\Delta t/k$ further to the right. Thus the propagation velocity of this sinusoidal wave is $\Delta x/\Delta t=\omega/k$. Note that $\omega=2\pi/T$ where T is the wave period, and $k=2\pi/\lambda$ where λ is the wavelength. Figure A.1 illustrates the propagation character of this function.

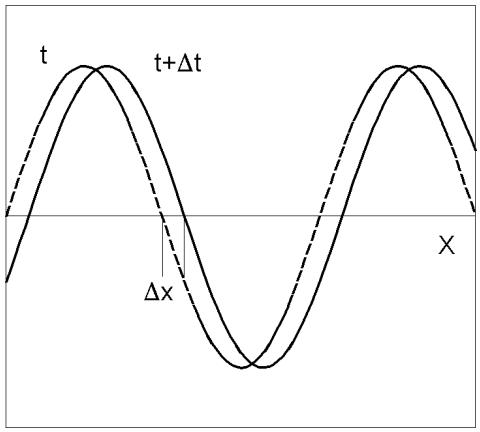


Figure A.1. propagation of a sinusoidal function

A.2. Fourier series for a finite interval

Since sinusoidal functions have many attractive properties it is valuable to be able to decompose an arbitrary function of x into a series of sinusoidal functions. Such a series is called a Fourier series.

We consider a finite interval on the x-axis: [0,L]. The series is built up from cos and sin functions with wavelengths L, L/2, L/3, ... and one constant term. So any function $F(x)$ (even most discontinuous finite functions) on the interval [0,L] can be written as:

$$F(x) = a_0 + \sum_{n=1}^{\infty}\left(a_n \cos\frac{2\pi nx}{L} + b_n \sin\frac{2\pi nx}{L} \right) \qquad \text{(A.2)}$$

The coefficients in this expression can be found in a fairly simple way thanks to special properties of the set of functions we are using for the development:

$$a_0 = \frac{1}{L}\int_0^L F(x)dx \qquad \text{(A.3a)}$$

$$a_n = \frac{1}{2L}\int_0^L F(x)\cos\frac{2\pi nx}{L}dx \qquad \text{for } n\geq 1 \qquad \text{(A.3b)}$$

$$b_n = \frac{1}{2L}\int_0^L F(x)\sin\frac{2\pi nx}{L}dx \qquad \text{(A.3c)}$$

For details see e.g. Hsu (1970).

example: consider a function which is 1 on the interval [0,L/2) and 0 on (L/2,L]. We find:

$$a_0 = \frac{1}{L}\int_0^L F(x)dx = \frac{1}{L}\int_0^{L/2} dx = \frac{1}{2}$$

$$a_n = \frac{1}{2L}\int_0^L F(x)\cos\frac{2\pi nx}{L}dx = \frac{1}{2L}\int_0^{L/2} \cos\frac{2\pi nx}{L}dx = 0 \qquad \text{for } n\geq 1$$

$$b_n = \frac{1}{2L}\int_0^L F(x)\sin\frac{2\pi nx}{L}dx = \frac{1}{2L}\int_0^{L/2}\sin\frac{2\pi nx}{L}dx = \frac{2}{n\pi} \qquad \text{for odd } n$$

$$= 0 \qquad \text{for even } n$$

Figure A.2 shows two approximations of $F(x)$ by means of a Fourier series, one taking into account terms up to n=3 and one with terms up to n=7.

We see from figure A.2 that the series has the most difficulty in approximating F near the discontinuity; this is understandable because we try to approximate a discontinuous function by means fo a sum of continuous functions. We see that the amplitudes of a_n and b_n decrease and eventually go to 0. If we consider a smoother $F(x)$ we will see that the values of the coefficients go to 0 much more quickly, in other words that we do not need so many terms of the Fourier series to get a good approximation.

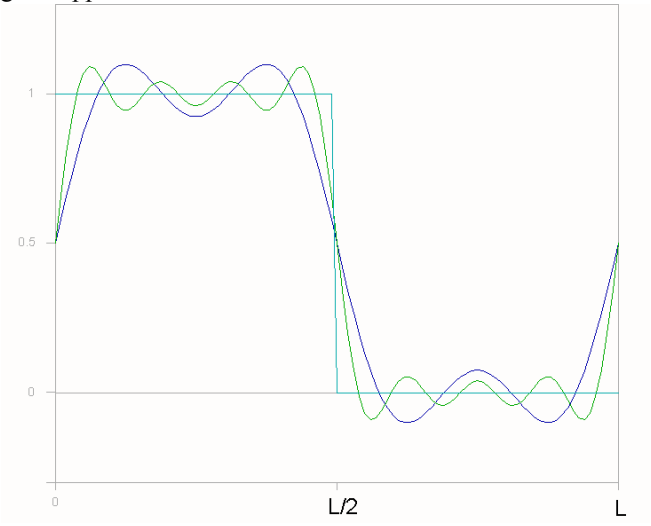


Figure A.2 Fourier approximation of a step function

Exercise:

Find the Fourier series for the function (see figure A.3):

$F(x)=2x/L$ if $x<L/2$

$F(x)=2-2x/L$ if $x>L/2$

and verify that the coefficients decrease more quickly than in the previous example.

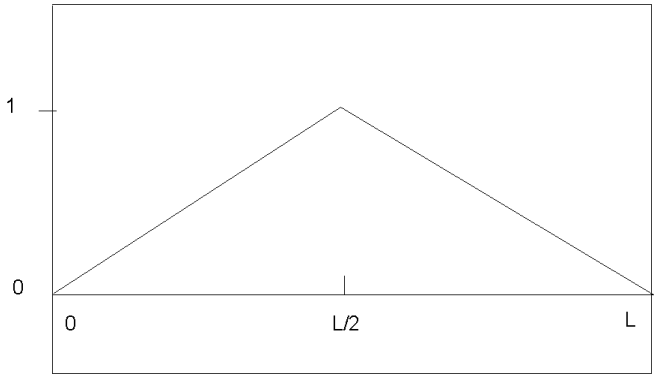


Figure A.3 a function of x

Note that there is a very efficient method to determine the Fourier transform numerically; it is called the Fast Fourier Transform (FFT), described in many textbooks, e.g. Press et al. (1988).

It is said above that the Fourier series is developed for a function on a finite interval. If we evaluate the Fourier series for values of x outside the interval we see that we get a periodical function with period L. So we can interpret the Fourier series also as a development of a **periodical** function.

A.3. Complex Fourier Series

In view of the first section of this appendix it is often convenient to have a Fourier series expressed in complex exponential functions. We can rewrite equation (A.2) as:

$$F(x) = a_0 + \sum_{n=1}^{\infty}\left(\frac{1}{2}(a_n - ib_n)e^{2\pi inx/L} + \frac{1}{2}(a_n + ib_n)e^{2\pi inx/L}\right) \tag{A.5}$$

or in terms of complex coefficients:

$$F(x) = \sum_{n=-\infty}^{\infty}\left(c_n e^{2\pi inx/L}\right) \tag{A.6}$$

where

$$c_n = \frac{1}{L}\int_0^L F(x)e^{-2\pi inx/L}dx \tag{A.7}$$

If $F(x)$ is real (as is usually the case in our applications) c_0 is real and $c_{-n}=c_n^*$ where the * denotes the **complex conjugate**.

A.4. Fourier analysis

In many applications we need the analogy of the Fourier series for an infinite domain; this is called the **Fourier Transform**. We can make the transition by considering a finite interval [-L,L] and letting L go to infinity. We will see then that instead of the discrete series we get a continuous function of the wavenumber k. Very often this transform is written in complex form:

$$F(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} C(k)e^{ikx}dk \tag{A.8}$$

where the complex function:

$$C(k) = \int_{-\infty}^{\infty} F(x)e^{-ikx}dx \tag{A.9}$$

is called the Fourier Transform. If $F(x)$ is real $C(-k)=C^*(k)$.

Fourier transforms are used frequently in the description of continuous stochastic processes, such as sea waves and wind-induced vibrations of structures.

example:

For the function

$$F(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-x^2/2\sigma^2} \tag{A.10}$$

the Fourier transform is:

$$C(k) = e^{-\sigma^2/k^2} \tag{A.11}$$

From the transform we can see that the contribution of wavelengths longer than e.g. σ is very small. So by looking at the Fourier transform we can conclude what a relevant characteristic wavelength for a given function is.

Appendix B. Taylor series

B.1. Taylor series in 1 dimension

It is well known from any book on calculus (e.g. Almering et al., 1990) that we can develop a well-behaved function (n times differentiable) into a Taylor series:

$$F(x+a) = F(x) + F_x(x)\,a + \frac{1}{2!}F_{xx}(x)\,a^2 + \frac{1}{3!}F_{xxx}(x)\,a^3 + \ldots \qquad \text{(B.1)}$$

Here the subscript x denotes differentiation with respect to x.

The series will converge usually only in a restricted neighbourhood of the point x.

B.2. Taylor series in 2 dimensions

In chapter 4 we often need a two-dimensional form of the Taylor series, i.e. we need to develop a function of two independent variables $F(x+a_x\,,\, y+a_y)$ into a series containing powers of a_x and a_y with respect to the point of reference (x,y). We do this in two steps, first we develop with respect to x and then with respect to y:

$$F(x+a_x, y+a_y) = F(x, y+a_y) + F_x(x, y+a_y)\,a_x + \frac{1}{2!}F_{xx}(x, y+a_y)\,a_x{}^2 + \frac{1}{3!}F_{xxx}(x, y+a_y)\,a_x{}^3 + \ldots$$

Next we develop each term of this equation with respect to y:

$$F(x,y) + F_y(x,y)\,a_y + \frac{1}{2!}F_{yy}(x,y)\,a_y{}^2 + \frac{1}{3!}F_{yyy}(x,y)\,a_y{}^3 + \ldots$$

$$+F_x(x,y)\,a_x + F_{xy}(x,y)\,a_x a_y + \frac{1}{2!}F_{xyy}(x,y)\,a_x a_y{}^2 + \frac{1}{3!}F_{xyyy}(x,y)\,a_x a_y{}^3 + \ldots \qquad \text{(B.2)}$$

$$+\frac{1}{2!}F_{xx}(x,y)\,a_x{}^2 + \frac{1}{2!}F_{xxy}(x,y)\,a_x{}^2 a_y + \left(\frac{1}{2!}\right)^2 F_{xxyy}(x,y)\,a_x{}^2 a_y{}^2 + \frac{1}{2!}\frac{1}{3!}F_{xxyyy}(x,y)\,a_x{}^2 a_y{}^3 + \ldots$$

$$+\frac{1}{3!}F_{xxx}(x,y)\,a_x{}^3 + \frac{1}{3!}F_{xxxy}(x,y)\,a_x{}^3 a_y + \frac{1}{2!}\frac{1}{3!}F_{xxxyy}(x,y)\,a_x{}^3 a_y{}^2 + \left(\frac{1}{3!}\right)^2 F_{xxxyyy}(x,y)\,a_x{}^3 a_y{}^3 + \ldots$$

Note that a number of mixed derivatives appear.

We can write the above expansion shorter (and easier to remember) as:

$$F(x+a_x, y+a_y) = F(x,y) + \left(a_x\frac{\partial}{\partial x} + a_y\frac{\partial}{\partial y}\right)F(x,y) + \frac{1}{2!}\left(a_x\frac{\partial}{\partial x} + a_y\frac{\partial}{\partial y}\right)^2 F(x,y) + \frac{1}{3!}\left(a_x\frac{\partial}{\partial x} + a_y\frac{\partial}{\partial y}\right)^3 F(x,y) + \ldots \qquad \text{(B.3)}$$

where we make use of the well known rule for the power of a sum of two variables:

$$(a+b)^n = a^n + \binom{n}{1}a^{n-1}b + \binom{n}{2}a^{n-2}b^2 + \ldots.$$

Equation (B.3) can be written shorter, retaining only terms up to the third power of a_x and a_y:

$$F(x+a_x, y+a_y) = F(x,y) + F_x(x,y)a_x + F_y(x,y)a_y$$
$$+\frac{1}{2}\left(F_{xx}(x,y)a_x^{\,2} + 2F_{xy}(x,y)a_x a_y + F_{yy}(x,y)a_y^{\,2}\right) \qquad \text{(B.4)}$$
$$+\frac{1}{6}\left(F_{xxx}(x,y)a_x^{\,3} + 3F_{xxy}(x,y)a_x^{\,2}a_y + 3F_{xyy}(x,y)a_x a_y^{\,2} + F_{yyy}(x,y)a_y^{\,3}\right) + \ldots$$

B.3. Linearization of an equation

Most equations describing real-life physical phenomena (partial differential equation or other type of equation) are nonlinear which makes it often impossible to find analytical solutions. If we linearize those equations we often can find analytical solutions (which then are only approximative solutions of the original problem).

Linearization of an equation means making the Taylor expansion of the equation with respect to the **dependent** variables, retaining only the zero and first order terms.

example: in the shallow water equation (see section 2.3) the dependent variables (the principal unknowns) are water level h and discharge Q. We consider the equation of motion (2.21) which contains a number of nonlinear terms:

$$\frac{\partial Q}{\partial t} + \frac{\partial QU}{\partial x} + g A_s \frac{\partial h}{\partial x} + C_{fr}\frac{|Q|Q}{A_s R} + \frac{F_W}{\rho} = 0 \qquad \text{(B.5)}$$

When we linearize this equation we assume that there is a small variation of h and Q on top of a uniform flow situation. The small variations of Q and h are called q' and h', resp., i.e. $Q=Q_0+q'$ and $h=h_0+h'$. Next we consider each term of the equation as a function of Q and h and we make the Taylor series with Q_0 and h_0 as our "point of reference". Consider for instance the friction term:

$$F(Q,h) = C_{fr}\frac{|Q|Q}{A_s R}$$

where A_s and R are functions of h; we assume: $A_s=b_s d$ and $R=d$. Then we can make the Taylor series:

$$F(Q,h) = C_{fr}\frac{|Q|Q}{b(h-z_b)^2}$$

$$= F(Q_0,h_0) + \frac{\partial F}{\partial Q}q' + \frac{\partial F}{\partial h}h' = C_{fr}\frac{|Q_0|}{b(h-z_b)^2} + 2C_{fr}\frac{|Q_0|}{b(h_0-z_b)^2}q' - 3C_{fr}\frac{|Q_0|Q_0}{b(h_0-z_b)^3}h'$$

We use here that the derivative of the function $|x|x$ is: $2|x|$.

We leave to the reader to verify that the result for the complete equation of motion is:

$$\frac{\partial q'}{\partial t} + 2U\frac{\partial q'}{\partial x} - U^2 b_s \frac{\partial h'}{\partial x} + g A_s \frac{\partial h'}{\partial x} - g I_b b_s h' + 2C_{fr}\frac{|U|}{d}q' - 2C_{fr}\frac{|U|Q_0}{d^2}h' = 0 \qquad \text{(B.6)}$$

Note that the terms of zero order have disappeared; they cancel because the uniform flow situation must also fulfill the equation of motion. This is not a particular property of this equation; the zero order terms always drop from a linearized equation, leaving a purely linear equation, containing only first order terms.