Course CIE4340

Computational Modelling of Flow and Transport

January 2015

Faculty of Civil Engineering and Geosciences

M. Zijlema

TU Delft Delft University of Technology

Artikelnummer 06917300083

CIE4340
Computational modelling of flow and transport

by	:	M. Zijlema
mail address	:	Delft University of Technology Faculty of Civil Engineering and Geosciences Environmental Fluid Mechanics Section P.O. Box 5048 2600 GA Delft The Netherlands
website	:	*http://fluidmechanics.tudelft.nl*

Contents

Chapter 1

Introduction

In this chapter, the general outline of the course CIE4340, **computational hydraulics**, will be explained. Computational hydraulics is an *applied* science aiming at the simulation by computers of various physical processes involved in seas, estuaries, rivers, channels, lakes, etc. It is one of the many fields of science in which the application of computers gives rise to a new way of working, which is intermediate between purely theoretical and experimental. This discipline is not an *independent* development, but rather a synthesis of various disciplines like applied mathematics, fluid mechanics, numerical analysis and computational science.

There is not a great deal of difference with the discipline **computational fluid dynamics** (CFD), but it is too much restricted to the fluid as such. It seems to be typical of practical problems in hydraulics that they are rarely directed to the flow by itself, but rather to some consequences of it, such as wave propagation, transport of heat, sedimentation of a channel or decay of a pollutant.

Mathematical and computational models are at the foundation of much of computational hydraulics. They can be viewed as a series of mappings from a part of the real world via abstract number spaces, as will be discussed in Section 1.1, onto a computer, as will be outlined in Section 1.2. This is illustrated in Figure 1.1. In this way **similarity** with respect to dynamics and shape of structure between the real world and the computer is obtained. Generally, this similarity is *not* isomorphic[1] because

- a mathematical model is invariably a *simplification* and cannot describe every aspect of the real world, and
- a computational model may contain *artifacts* that have no corresponding property in the real world.

To get insight into the shortcomings of the mathematical and computational models, we first consider a part of the **real world** of which the models are to be set up. In this course

[1]An isomorphism is a mapping between two systems that are structurally the same even though the names and notation for the elements are different.

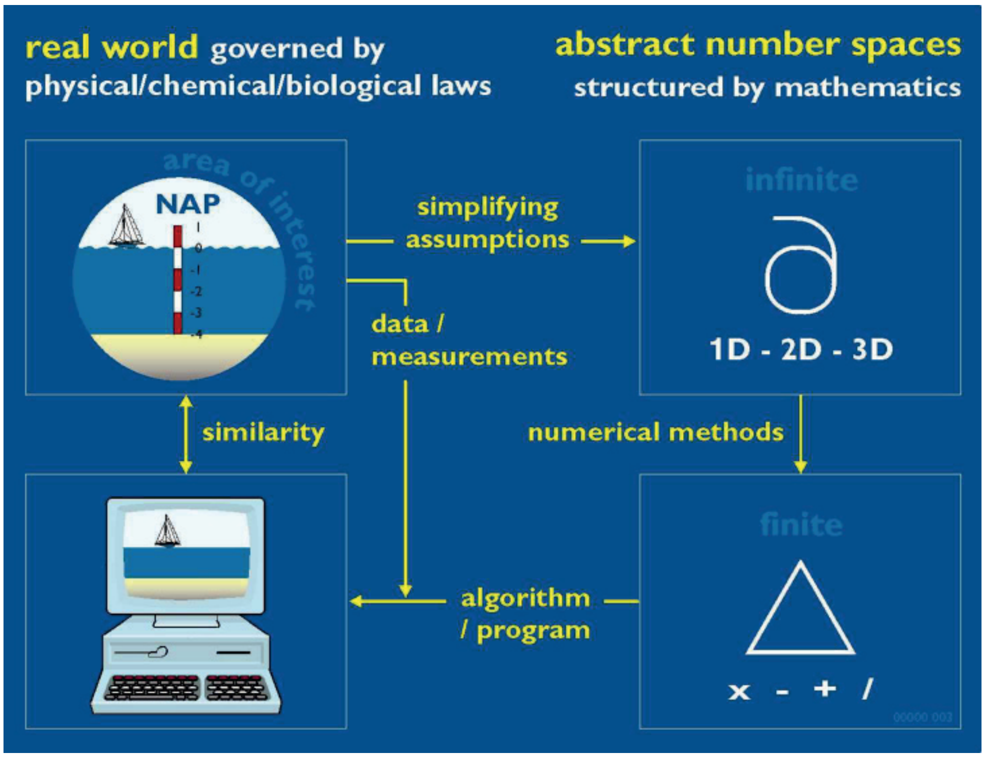


Figure 1.1: Mappings of (a part of) the real world onto a computer.

we restrict ourselves to applications with open water bodies as seas, estuaries, rivers, lakes and channels. Figure 1.2 shows some examples.

1.1 A mathematical model

Many important concepts of mathematics were developed in the framework of physical science and engineering. For example, **calculus** has its origins in efforts to describe the motion of bodies. Mathematical equations provide a language in which to formulate concepts in physics. A **mathematical model** is an equation or set of equations whose solution describes the physical behaviour of a related physical system. For instance, Newton's equations describe mechanical systems, Maxwell's equations describe electrodynamical phenomena, Schrödinger's equation describes quantum phenomena, etc.

In this way, we have created a mapping from the real world to an **abstract number space**. In this space, real numbers are represented as symbols (x, y, ω, π, e, etc.). Moreover, this **continuous space** is structured by mathematical notions such as **continuity** and **differentiability**, and is governed merely by mathematical laws. For instance, **well posedness** is a mathematical notion that follows only from fulfilling some basic mathematical rules, so that well posed problems can produce *useful* solutions.

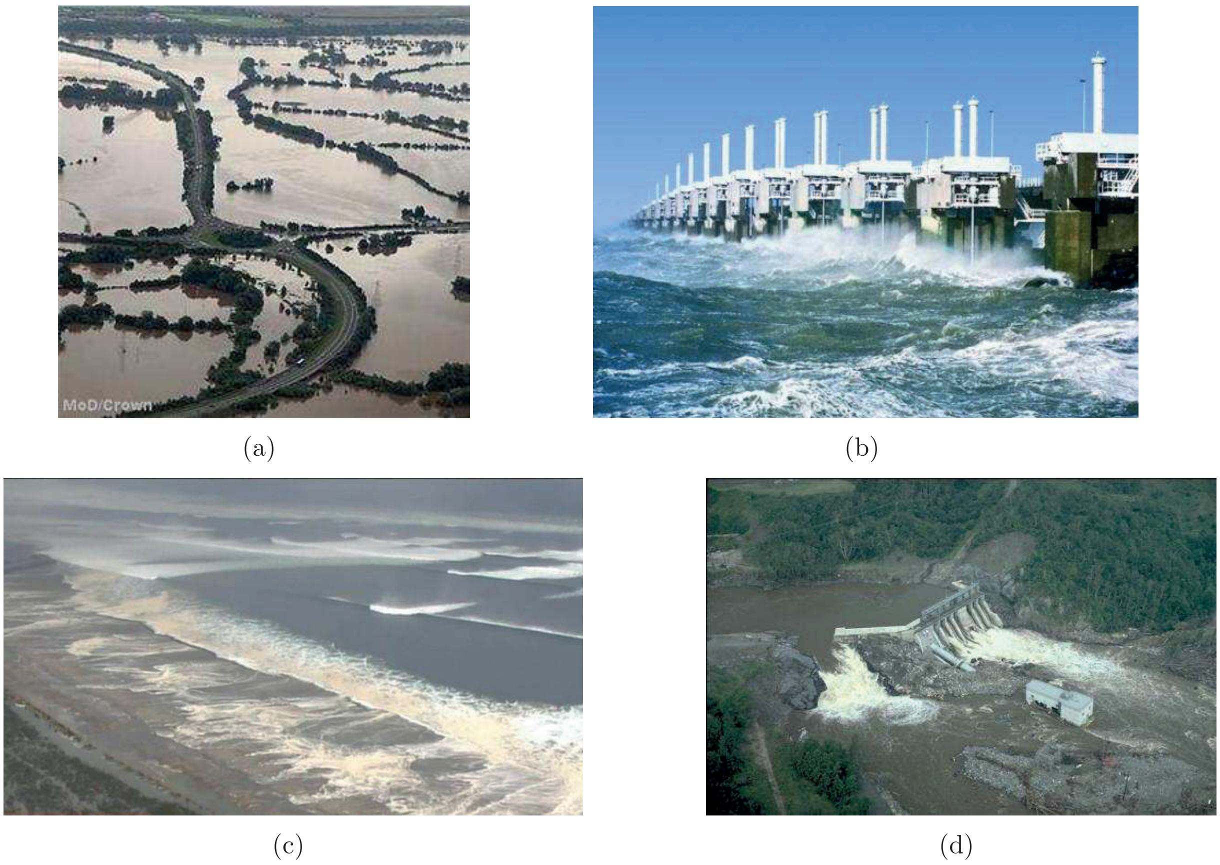


Figure 1.2: Different areas of interest and applications in hydraulics: (a) coastal flooding, (b) Delta works, (c) tsunami waves and (d) dam break.

Physical phenomena depend in complex ways on time and space. Scientists often seek to gain understanding of such phenomena by casting *fundamental principles* in the form of mathematical models. Physical, chemical and biological theories are rooted in the concept that certain quantities, such as mass, momentum, energy, charge, spin, population of organisms, etc. are conserved. A **conservation law** is simply the mathematical formulation of the basic fact that the rate at which a quantity changes in a given domain must equal the rate at which the quantity flows into or out of the domain.

Rate of change equals input minus output.

In deriving mathematical equations from the conservation laws, a number of simplifying assumptions needs to be introduced in order to make these equations *feasible*. Though these assumptions are a source of errors and limit the generality of the applicability of a mathematical model, they are often not so much restrictive. An example is the **hydrostatic pressure asssumption**. For coastal waters, this is not a harmful one. Another example is **turbulence modelling** which is required since most flows in hydraulic applications are turbulent. A key feature of this modelling is averaging the governing equations of turbulence, while some closure assumptions are introduced to represent scales of the flow that are not resolved. Sometimes these assumptions can be severe.

Usually a mathematical model requires more than one independent variable to characterize the state of the physical system. For example, to describe flow and transport in coastal waters usually requires that the physical variables of interest, say flow velocity, water level, salinity, temperature and density, be dependent on time and three space variables. They are governed by conservation of mass, momentum and energy. They form a set of **partial differential equations** (PDEs). Together with boundary and initial conditions, they represent a **boundary value problem**. If a mathematical model involves only one independent variable (usually time) and the physical variables of interest are nonconstant with respect to that independent variable, then the mathematical model will contain **ordinary differential equations** (ODEs). They need to be initiated by certain starting conditions. This is called an **initial value problem**.

Formulation of the equations is based on experimentation, observations and intuition. A mathematical model is always a *simplified* description of physical reality expressed in mathematical terms. The recipe for mathematical modelling of a physical system, is essentially a *bedrock* for the scientific method, and is as follows

1. experimentation and physical observations
2. selection of the relevant physical variables
3. formulation of equations governing the inter-dependance of these variables
4. solution of the equations via analysis and numerical simulation
5. validation of the mathematical model via comparison with observations

The last step, validation, involves comparison of simulation results with observations to ascertain whether the model describes the physical phenomenon. Since the mathematical model is a simplification, there are always discrepancies between model solutions and observations. These discrepancies motivate refinement of the model.

Refinement of the mathematical model eventually results in a set of equations that are not amiable to analytical solution, particularly when nonlinear terms are included. Numerical simulation provides an avenue for solving nonlinear equations but requires considerable care to ensure that the numerical solution is a *valid* solution of the equations. An important prerequisite for numerical simulation is the use of a **computational model**.

1.2 A computational model

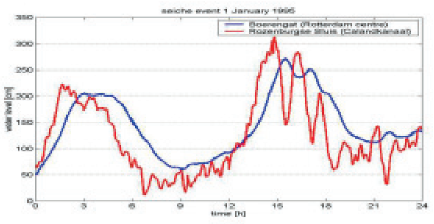
A computational model is an approximation which leads towards some degree of errors. This is a direct consequence of the fact that digital computers have a *finite* amount of memory. In this respect, we consider a **discrete space** or **grid** containing a finite number of **grid points**. The notions continuity and differentiability do not exist but real numbers still exist. Functions are represented as series of samples in the discrete space. The only operations are addition, subtraction, multiplication, division and some other similar operations. ODEs and PDEs are replaced by **recurrent relations**. The solution to these recurrent relations is not always trivial. It may not **converge** to the exact solution or it may become **unstable**. This requires **numerical analysis** of the used approximations.

Computational models consist of various numerical techniques and are usually tedious and repetitive arithmetic, which is not possible to carry out without the help of **computer**. This computer is the final stage of our sequence of mappings; see Figure 1.1. The numerical methods can be synthesized into an **algorithm** written in a high-level programming language, such as Fortran90, C++ or Matlab, with which the computer can be controlled. Below is an excerpt of a typical computer code for the Fortran implementation of the solution of shallow water equations.

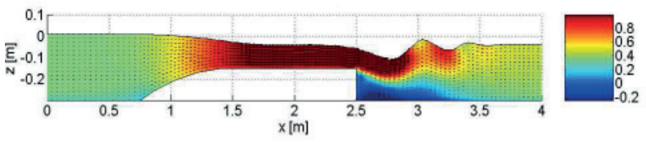
```
!
! compute v-velocity component and velocity magnitude
!
v = 0.25 * ( v0(nm) + v0(nmu) + v0(ndm) + v0(ndmu) )
!
utot = sqrt( u0(nm)*u0(nm) + v*v )
!
! compute water level gradient
!
zgrad = grav * (s0(nmu) - s0(nm)) / gvu(nm)
!
! compute bottom friction
!
cbot = cfricu(nm) * utot / hum(nm)
!
! compute total contributions of the momentum equation
!
contrib = advecx(nm) + advecy(nm) + zgrad - visc(nm)
!
denom = 1. + dt * cbot
!
! compute flow velocity
!
u1(nm) = ( u0(nm) - dt*contrib ) / denom
!
```

Often algorithms are a part of a general-purpose **simulation tool** and they are not supposed to be reprogrammed for each new application. Only input parameters need to be changed. Most essential input data is boundary conditions and bathymetry. The completeness and quality of this data depends on the scale of measuring programme and the resolution at with which the measurements have been carried out.

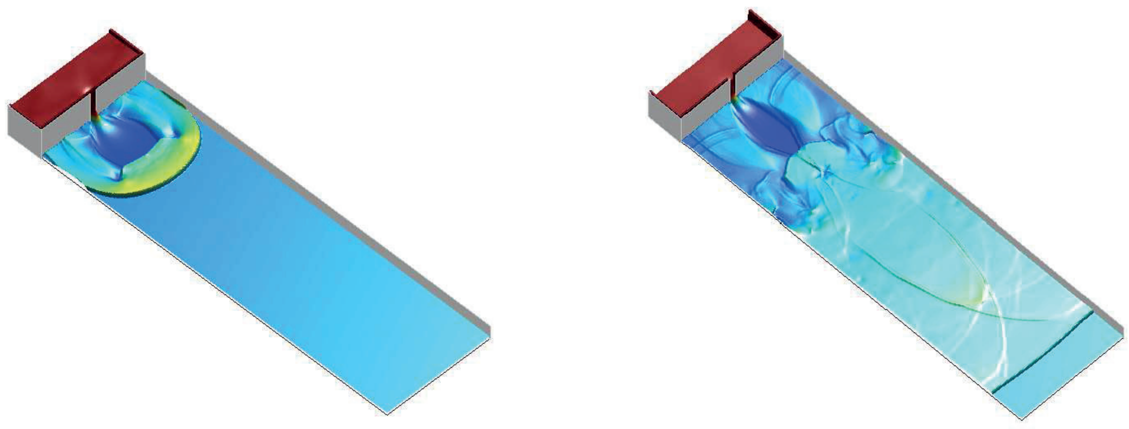
Graphical devices as part of computer systems permit us to represent outcomes of a simulation as graphs or animations that make the similarity between a computer model and the real world visible. For illustrative purposes, in Figure 1.3 some examples are presented.



(a)



(b)



(c)

Figure 1.3: Various kinds of plots of simulation results: (a) a time series graph of water level, (b) a $(x, z)-$map of the flow velocity (in m/s) over a weir, and (c) some snapshots of a two-dimensional dambreak.

Due to the approximation, we may encounter the following error sources in any computational model.

Truncation error

The transcendental function e^x is computable as an infinitely long series as given below

$$e^x = \sum_{j=0}^{\infty} \frac{x^j}{j!}$$

However, on digital computers, it can only be approximatated, for instance, through cubic polynomial as

$$e^x \approx p(x) = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3$$

and the error of this approximation is due to the truncation of the series.

Round-off error
Digital computers always use floating point numbers of fixed word length. The true values are not expressed exactly by such representations. Such error due to this computer imperfection is round-off error.

Measurement error
Any physical problem is controlled by external data, often measured ones, such as boundary conditions, bottom topography and other (empirical) coefficients that are imperfectly known.

Gross error
Computers make mistakes very seldom, but since humans are involved in programming, input preparation, and output interpretation, gross errors or blunders do occur more frequently than we like to admit.

One of the main objectives of computational hydraulics is to obtain simulations of 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.

The purpose of this course is to teach methods for numerically solving ordinary and partial differential equations arising in the area of hydraulics. The emphasis will be on

- **recognizing the strengths and weaknesses of the various numerical schemes and**
- **understanding how numerical algorithms used by many well-known numerical packages work.**

1.3 Scope of this lecture notes

This course CIE4340 aims to provide basic skills in the use of numerical techniques to solve several problems arising in hydraulics and analyze the numerical results. Initially, the problems considered are those where there is a single independent variable, and we solve ordinary differential equations (ODEs). Subsequently, we consider problems such that variation with time and space are involved, and we solve partial differential equations (PDEs). This is followed by the consideration of shallow water equations that describe important phenomena in hydraulics, and develop numerical methods for them.

We have attempted to present the material in this lecture notes at an easy pace, explaining carefully both the ideas and details of the derivations. This is particularly the case in the first parts of the chapters but subsequently less details are included and some steps are left for the reader to fill in. There are a lot of exercises included, ranging from the simple to more challenging ones. These exercises are essentially for a better understanding of the discussed topics in this lecture notes. We strongly encourage students not to skip these exercises.

The remainder of this lecture notes is subdivided as follows.

- Chapter 2 deals with various time integration schemes for ODEs. Concepts and definitions of consistency, stability and convergence are outlined.
- In Chapter 3, first, a brief review of PDEs is given. Next, discretizations of the diffusion equation by employing the finite difference method are considered. This is followed by the approximations of the convection equation and the convection-diffusion equation. Also, special attention is paid to the analysis of discretization errors.
- Chapter 4 is concerned with discussing several numerical techniques for the approximation of the shallow water equations.
- Appendices A and B give short overviews on the theory of Taylor and Fourier series, respectively.
- A bibliography is included at the end of this lecture notes. This list contains most essential reference material that you may consult during your study.

Chapter 2

Numerical methods for initial value problems

In this chapter we start with a study of time-dependent ordinary differential equations (ODE). First, in Section 2.1, we focus on one equation with one unknown solution. Later, we extend this problem with a system of multiple ODEs with equal amount of unknown solutions as presented in Section 2.2.

2.1 First order ordinary differential equation

In this section we discuss the numerical solution of the first order ODE using the time integration. The specific first order differential equation we wish to solve is the following

$$\frac{dy}{dt} = f(y) \tag{2.1.1a}$$

$$y(0) = y_0 \tag{2.1.1b}$$

This is called an **initial value problem**. We seek for a solution $y(t)$ in the range $0 \leq t \leq T$. We thereby assume that Eq. (2.1.1a) has a unique solution. This assumption imposes some requirements on $f(y)$, most notably Lipschitz continuity, which means that there must be a certain amount of smoothness in the function $f(y)$.

Two properties of computer hardware must be considered when devising a numerical solution for an ODE problem. These place strict constraints on the size of the problem we can practically solve and also on the accuracy with which we may calculate the solution.

- Computers have a finite amount of memory (bytes).
- Computers have a finite computation speed (operations/second).

A finite amount of memory places two constraints on any numerical solution. Firstly, we cannot represent real numbers to arbitrary precision, i.e. an infinite number of decimal places. Secondly, we can only store a finite number of values in memory at any given time. Consequently, the notion *continuity* is meaningless for the actual implementation on the computer as we cannot compute the values of a function at every single time t. Also, it is not possible to represent infinitesimal changes dt and hence, it is not possible to implement the derivative of Eq. (2.1.1a) at once. In fact, a computer can only performs simple operations like addition, substraction, multiplication and division. So, instead of solving Eq. (2.1.1a) exactly, we shall try to construct another equation that can be solved on a computer, and that generates an approximate solution of Eq. (2.1.1a). The following sections will deal with such a construction.

2.1.1 Time integration

To be able to find an approximate solution for Eq. (2.1.1a) a **continuous** time domain has to be replaced by a **discrete** grid domain. For this, we discretize the time frame $0 \leq t \leq T$ into a number of steps with a step size of Δt: $t_0 = 0, t_1 = \Delta t, t_2 = 2\Delta t$, $t_3 = 3\Delta t, \cdots, t_n = n\Delta t$. Given the initial solution, $y(0) = y_0$, we should be able to march forward in time, computing approximations, $y^1, y^2, y^3, \cdots, y^n$ at successive times $t_1, t_2, t_3, \cdots, t_n$; see Figure 2.1.

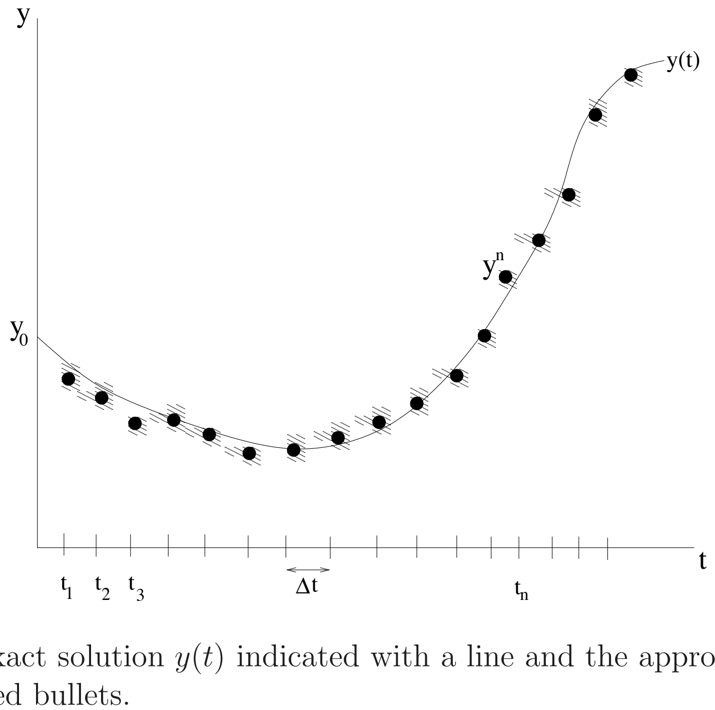


Figure 2.1: The exact solution $y(t)$ indicated with a line and the approximate solution y^n indicated with filled bullets.

These approximate solutions should satisfy

$$y^n \approx y(t_n) = y(n\Delta t)$$

with $y(t)$ a *continuous* function defined in the continuous domain $0 \leq t \leq T$ and y^n a *discrete* function defined in the discrete domain $t_n = n\Delta t$, $n = 0, 1, 2, \cdots$. We shall use superscripts to denote the time step index for the approximate solution.

A finite computation speed places a practical constraint on the size of the problem we can solve in a reasonable amount of time. Specifically there is a limitation upon the number of time steps ($= T/\Delta t$) we can consider within the time available to perform the computation. Nevertheless, thanks to the rapid increase in computer speeds and the advent of parallel super computers, we can now solve quite large problems in relatively short time, even on a desktop PC.

Recalling the definition of the first derivative

$$\frac{dy}{dt} = \lim_{\Delta t \to 0} \frac{y(t + \Delta t) - y(t)}{\Delta t} \tag{2.1.2}$$

it follows that

$$\frac{dy}{dt} = \frac{y(t + \Delta t) - y(t)}{\Delta t} - \tau_{\Delta t} \tag{2.1.3}$$

with $\tau_{\Delta t}$ the so-called **truncation error**. Since,

$$\lim_{\Delta t \to 0} \tau_{\Delta t} = 0 \tag{2.1.4}$$

we may expect that

$$\frac{y(t + \Delta t) - y(t)}{\Delta t}$$

is a good approximation for the derivative dy/dt for small Δt. This approximation is called the **forward difference** quotient and is easily to be calculated on a digital computer. Similarly, we can also have a **backward difference** quotient, as follows

$$\frac{y(t) - y(t - \Delta t)}{\Delta t} \tag{2.1.5}$$

We must realised, however, that $y(t)$, $y(t - \Delta t)$ and $y(t + \Delta t)$ are afflicted with some errors due to an error of approximation and a round-off error, since the computer can only perform a finite number of digits in case of floating point calculations. Thus, replacing $y(t_n)$ by y^n and dy/dt in Eq. (2.1.1a) by

$$\frac{y^{n+1} - y^n}{\Delta t}$$

we have the following scheme

$$\frac{y^{n+1} - y^n}{\Delta t} = f(y^n) \tag{2.1.6}$$

This scheme is known as the **forward Euler** or **explicit Euler** scheme. With this scheme and $y^0 = y_0$, we can generate a sequence $\{y^n\}, n = 1, 2, 3, \cdots$, by repeating the formula

$$y^{n+1} = y^n + \Delta t\, f(y^n) \tag{2.1.7}$$

This is called **time integration** or **time marching**, and the scheme, Eq. (2.1.7), to be implemented on a computer with which a numerical solution can be constructed is called a **numerical recipe**[1] or an **algorithm**.

There are many ways to implement an algorithm on the computer by means of a high-level computer language. Typical languages often used in the science and engineering environment are Fortran, C and Matlab. Below an example is given of the Matlab implementation for explicit Euler based on Eq. (2.1.7) with $f(y) = -y^2$.

```
function y = euler( y0, dt, n )
%
% this Matlab function computes the solution of the following ODE
%
%   y' = -y * y
%
% by means of the explicit Euler scheme
%
% Meaning of input variables:
%
% y0 is the initial value
% dt is the time step
% n  is the number of steps to carry out
%
y(1) = y0;
%
for i = 2:n
    y(i) = y(i-1) - dt * y(i-1)^2;
end
```

With this Matlab function we can compute an approximate solution of the differential equation $dy/dt = -y^2$. We choose an end time $T = 50$ s, a time step of $\Delta t = 0.5$ s and we start the model with $y_0 = 1$. Hence, the number of time steps are $n = 101$ (including the initial step $t = 0$). We can visualize the solution as depicted in Figure 2.2 with the following Matlab commands

```
dt= 0.5;
t = 0:dt:50;
y = euler(1,dt,101);
plot(t,y)
```

From this figure we see that the numerical solution is more or less closely to the exact one that is given by $y(t) = (1 + t)^{-1}$. The central question concerning the accuracy of the obtained approximate solution will be addressed in the next section.

[1]This phrase is taken from Press et al. (1996).

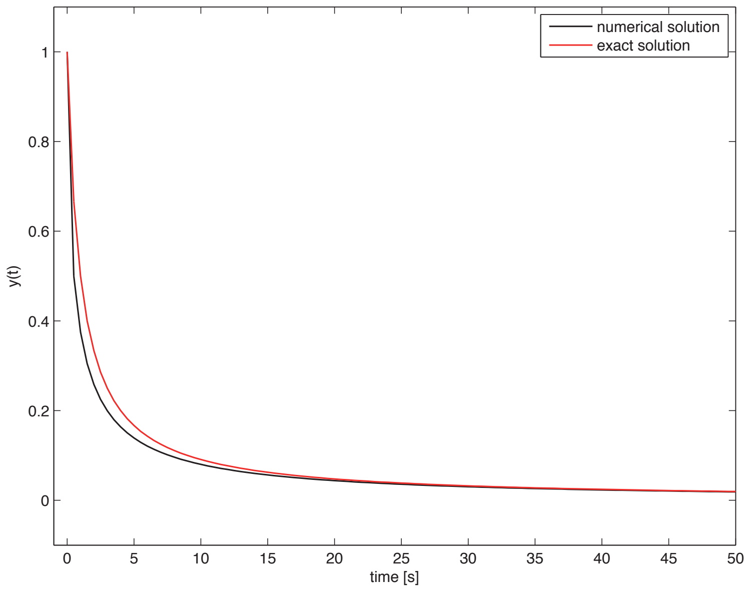


Figure 2.2: Approximate solution of $y' = -y^2$ obtained with explicit Euler with $\Delta t = 0.5$ s, compared with the exact one.

Exercise 2.1.1 Try to implement the above Matlab function on your own computer and do some experiments by varying the time step. Choose, for instance, $\Delta t = 1.1$ s. What happens?

The **backward Euler** or **implicit Euler** scheme is based on Eq. (2.1.5), and is given by

$$\frac{y^{n+1} - y^n}{\Delta t} = f(y^{n+1}) \tag{2.1.8}$$

Eq. (2.1.8) is called **implicit** because the function f needs to be evaluated based on the solution at the next time step, i.e. y^{n+1}, which is not known yet. Compare this with the **explicit** Euler scheme, Eq. (2.1.6), in which y^{n+1} is given explicitly in terms of y^n, the known value at the current time step, and $f(y^n)$ that can be computed directly based on y^n.

If we take the average of the forward and backward Euler schemes, we get

$$\frac{y^{n+1} - y^n}{\Delta t} = \frac{1}{2} f(y^n) + \frac{1}{2} f(y^{n+1}) \tag{2.1.9}$$

which is known as the **Crank-Nicolson** scheme or the **trapezoidal rule** (see also Appendix A). Clearly, this scheme can be regarded as implicit as well.

All three schemes can be generalized to an arbitrary weighted average of $f(y^n)$ and $f(y^{n+1})$,

$$\frac{y^{n+1} - y^n}{\Delta t} = \theta\, f(y^{n+1}) + (1 - \theta)\, f(y^n) \tag{2.1.10}$$

with $0 \leq \theta \leq 1$ a weight parameter to be chosen. This scheme is called the $\theta-$**method** and is implicit if $0 < \theta \leq 1$ and explicit if $\theta = 0$. The parameter θ is also called the **implicitness parameter**. In practice, we shall be almost exclusively interested in the explicit Euler ($\theta = 0$), implicit Euler ($\theta = 1$), and Crank-Nicolson ($\theta = 1/2$) cases.

2.1.2 Convergence, consistency and stability

In the previous section we have constructed a number of numerical recipes for the solution of $dy/dt = f(y)$. The main question is now: does such a numerical recipe or algorithm actually lead to a solution which is close to the exact or analytical solution $y(t)$ of the differential equation $dy/dt = f(y)$?

To answer this question we shall assume a fixed (but arbitrary) end time $T > 0$ and consider the **global error** e^n in our approximation to $y(t)$ computed with a numerical scheme using time step Δt,

$$e^n = y(t_n) - y^n$$

This error is due to an approximation and may also contain a round-off error due to floating point arithmetic.

The solution generated by the scheme converges to the exact solution as $\Delta t \to 0$. Hence, the numerical scheme is said to be **convergent** if

$$\lim_{\substack{\Delta t \to 0 \\ n = \frac{t}{\Delta t}}} e^n = 0$$

for all $t \in (0, T]$. Note that the number of time steps that we need to take to reach *fixed* end time T tends to infinity as $\Delta t \to 0$.

While the concept of convergence is seemingly obvious, a *proof of convergence* is far from trivial. Instead, we introduce two notions: **consistency** and **stability**. A basic rule of thumb which emerges is

$$\textit{consistency} + \textit{stability} \Rightarrow \textit{convergence}$$

Since convergence is the hard one, effort in the other two aspects is easily elaborated. Consistency defines a relation between the numerical scheme and the differential equation we wish to solve, whereas stability establishes a relation between the computed solution and the exact solution(!) of the numerical scheme. It should be emphasised that stability is a requirement *just* on the numerical scheme and contains no condition on the differential equation.

The concepts consistency and stability can be easily understood from an analysis of time integration schemes for the so-called **test equation**,

$$\frac{dy}{dt} = f(y) = \lambda y \tag{2.1.11}$$

with $\lambda < 0$ a parameter. The exact solution of this equation reads

$$y(t) = y_0 e^{\lambda t} \tag{2.1.12}$$

Since $\lambda < 0$, the solution is bounded, i.e. it does not grow to infinity when $t \to \infty$. This also means that the associated physical behaviour is not sensitive to small disturbances. Otherwise, it would be unfeasible when the solution behaves totally different when a small change occurs, for instance, in the initial solution because of round-off errors. This is illustrated in Figure 2.3. Assume that the initial condition y_0 is afflicted with a small

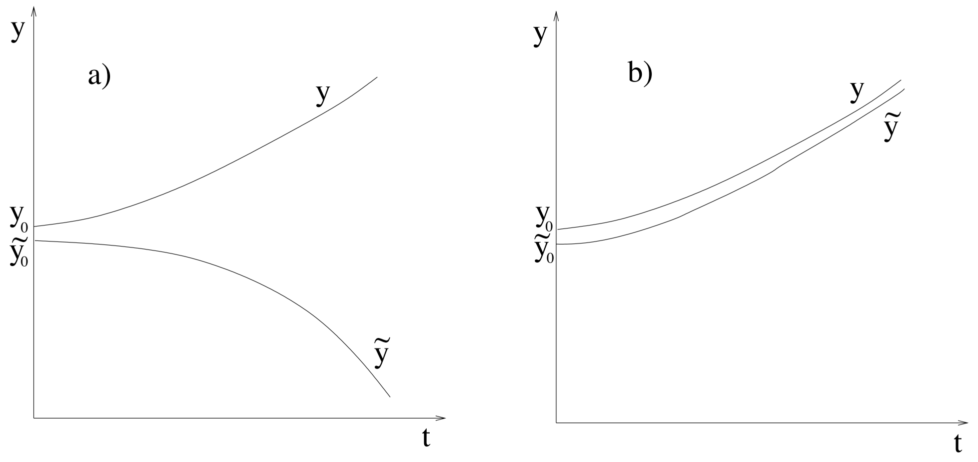


Figure 2.3: An example of a) unstable and b) stable differential equation.

round-off error and the computation is thus initialized with $\tilde{y_0}$. In the case of an **unstable** ODE this error grows to infinity when time proceeds and the solution is completely spoiled. On the other hand, the ODE is said to be **stable** if the error keeps small when time evolves. The ODE is **absolutely stable** if the error goes to zero when $t \to \infty$.

By comparing the exact solution, Eq. (2.1.12), with the numerical solution obtain from a variety of numerical schemes, some firm conclusions can be made in terms of **order of accuracy** and in terms of **stability** with respect to long term behaviour.

Let us consider the ratio y^{n+1}/y^n as a function of the time step Δt, and compare this with the exact expression for $y(t + \Delta t)/y(t) = \exp(\lambda \Delta t)$. Here, we are making *one step* from n to $n + 1$ with the time step Δt which is *fixed* and nonzero. During this step we assume that we are making an error due to the approximation, while at time level n there is no error. In other words, we assume $e^n = 0$, i.e. $y^n = y(t_n)$, whereas $e^{n+1} \neq 0$. Hence, this so-called **local error** is made during one step from $t_n = n\Delta t$ to $t_{n+1} = (n + 1)\Delta t$ with a numerical recipe whereby everything is exact at $t_n = n\Delta t$.

From Eq. (2.1.10) with $f(y) = \lambda y$, we can derive the ratio y^{n+1}/y^n for the $\theta-$method, which is given by

$$\frac{y^{n+1}}{y^n} = \frac{1 + (1 - \theta)\lambda \Delta t}{1 - \theta \lambda \Delta t} \tag{2.1.13}$$

The results for all the considered schemes are shown in Figure 2.4 for the case $\lambda = -1$. We can make the following observations.

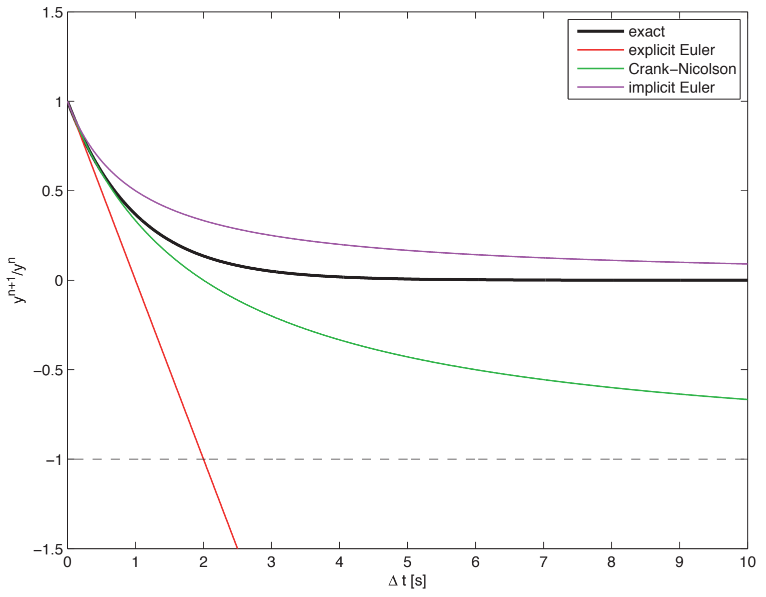


Figure 2.4: The behaviour of ratio y^{n+1}/y^n for both small and large Δt.

- All three schemes approximate the exact, exponential relation for small Δt.
- For small Δt the implicit Euler scheme approximates the exact solution to the same accuracy as explicit Euler.
- The Crank-Nicolson scheme is a better approximation to the exact solution for small Δt than either of the other two methods. That is the Crank-Nicolson curve is seen to follow the exact solution better at small Δt.
- The explicit Euler seems particularly bad for moderate and large Δt. In fact, when $\Delta t > 2$ s, $|y^{n+1}| > |y^n|$, i.e. the numerical solution is growing, which is certainly wrong.
- In the case of Crank-Nicolson, for most of the time it holds $|y^{n+1}| < |y^n|$ but in the limit $\Delta t \to \infty$ the solution will oscillate since $y^{n+1} = -y^n$.
- For implicit Euler, we have $y^{n+1} < y^n$ for all Δt. This means that the solution is decaying and will approach 0 in the limit $\Delta t \to \infty$.
- The implicit Euler scheme appear to be the best because is never gets too far from the exact solution and has the correct limit for large Δt.

To establish these observations, we further examine the schemes for both small and large Δt.

Consistency and order of accuracy

The standard approach to analyzing the error in a numerical scheme is to apply Taylor series expansion (see also Appendix A). First, recall the definition of the $\mathcal{O}-$symbol of Landau:

A function $R(\Delta t)$ is $\mathcal{O}(\Delta t^p)$ when $\Delta t \to 0$ if there exists a constant $M > 0$, independent of Δt, such that

$$|R| \leq M \Delta t^p\,, \text{ for sufficiently small } \Delta t$$

In this context, we say the function R is of **big O** of Δt^p and is a statement about how R behaves in the limit $\Delta t \to 0$.

Big O notation is commonly used to describe how closely a Taylor series approximates a given function. The most significant terms are written explicitly, while the least significant terms are summarized in a single big O term. This is illustrated with the following example

$$e^x = 1 + x + \frac{1}{2}x^2 + \mathcal{O}(x^3) \quad \text{as} \quad x \to 0$$

This indicates that the difference $e^x - (1 + x + x^2/2)$ is smaller in absolute value than $M|x^3|$, with $M > 0$ a constant, when x is close enough to 0.

Exercise 2.1.2 Expand $\sqrt{1+x}$ about $x = 0$.

We now return to our story. For small Δt, the ratio y^{n+1}/y^n for the considered schemes is expanded by means of the Taylor series, as follows:

$$\theta = 0 : \quad \frac{y^{n+1}}{y^n} = 1 + \lambda \Delta t = 1 + \lambda \Delta t$$

$$\theta = 1 : \quad \frac{y^{n+1}}{y^n} = (1 - \lambda \Delta t)^{-1} = 1 + \lambda \Delta t + \lambda^2 \Delta t^2 + \cdots \tag{2.1.14}$$

$$\theta = \frac{1}{2} : \quad \frac{y^{n+1}}{y^n} = (1 + \frac{1}{2}\lambda \Delta t)(1 - \frac{1}{2}\lambda \Delta t)^{-1} = 1 + \lambda \Delta t + \frac{1}{2}\lambda^2 \Delta t^2 + \frac{1}{4}\lambda^3 \Delta t^3 + \cdots$$

By comparing these ratios with the exact one

$$\frac{y(t + \Delta t)}{y(t)} = e^{\lambda \Delta t} = 1 + \lambda \Delta t + \frac{1}{2}\lambda^2 \Delta t^2 + \frac{1}{6}\lambda^3 \Delta t^3 + \cdots \tag{2.1.15}$$

we see that both explicit and implicit Euler schemes agree with the exact solution to $\mathcal{O}(\Delta t)$ and makes a local error of $\mathcal{O}(\Delta t^2)$, whereas Crank-Nicolson agrees with the exact to $\mathcal{O}(\Delta t^2)$ making a local error of $\mathcal{O}(\Delta t^3)$. Hence, for the Euler schemes, the error will be dominated by the term containing Δt^2 and and all the other terms will be negligible compared to this term, if Δt is small enough. We may therefore expect the error to behave roughly like a constant times Δt^2, where the constant has the value $\lambda^2/2$. This error is introduced in the step from n to $n+1$ with length Δt. The same conclusion holds for the trapezoidal rule where its local error behaves like $\lambda^3 \Delta t^3/12$.

In reality, when approximating y^{n+1} with, for instance, the explicit Euler scheme, we need to take into account all the errors from the previous time steps. Then we would expect the resulting *global error* at $t = t_{n+1}$ to be simply the sum of all these local errors. Since each local error is $\mathcal{O}(\Delta t^2)$ and we are adding up $t_{n+1}/\Delta t$ of them, we end up with a global error of $\mathcal{O}(\Delta t)$. Likewise, when the trapezoidal rule is applied in $t_{n+1}/\Delta t$ time steps, the global error in y^{n+1} is expected to be $\mathcal{O}(\Delta t^2)$. In this context, we say that the explicit Euler scheme is **first order accurate** in time, meaning that the size of the global error is roughly proportional to Δt. Similarly, the trapezoidal rule is **second order accurate** in time which indicates that the error is proportional to Δt^2. This error is much smaller than that of the explicit Euler scheme when Δt is small. For instance, if we half the time step, then the error in y^{n+1} approximated with explicit Euler becomes two times smaller, while with Crank-Nicolson it becomes four times smaller.

Generally speaking, a pth order accurate scheme has a local error of $\mathcal{O}(\Delta t^{p+1})$, while its global error is $\mathcal{O}(\Delta t^p)$. The **order of accuracy** of this scheme is said to be pth order.

A simple and practical way to determine the order of accuracy is to compute the **truncation error**. This error is the error in approximating the differential equations or differential operators by discrete representations such as numerical schemes. In our context, it is a measure of how well the numerical scheme replaces the test equation, see Richtmyer and Morton (1967). As an example, we consider the explicit Euler scheme,

$$\frac{y^{n+1}-y^n}{\Delta t} = \lambda y^n \tag{2.1.16}$$

Truncation error can be computed by replacing y^n by the exact solution $y(t_n)$ in Eq. (2.1.16). (For background details, see Appendix A). In general, the exact solution will not satisfy this approximation and the discrepancy is thus the truncation error, which is denoted by $\tau_{\Delta t}$ and given by

$$\tau_{\Delta t} = \frac{y(t_{n+1})-y(t_n)}{\Delta t} - \lambda y(t_n)$$

or

$$\tau_{\Delta t} = \frac{y(t+\Delta t)-y(t)}{\Delta t} - \lambda y(t) \tag{2.1.17}$$

at an arbitrary time level. In practice, however, we do not know what the exact solution $y(t)$ is, but if we assume it is sufficiently smooth then we can expand $y(t+\Delta t)$ in Taylor series, as follows

$$y(t+\Delta t) = y(t) + \Delta t y'(t) + \frac{1}{2}\Delta t^2 y''(t) + \cdots + \frac{\Delta t^{p-1}}{(p-1)!} y^{(p-1)}(t) + \mathcal{O}(\Delta t^p)$$

after which it is substituted in Eq. (2.1.17), giving

$$\tau_{\Delta t} = \frac{1}{2}\Delta t y''(t) + \frac{1}{6}\Delta t^2 y'''(t) + \mathcal{O}(\Delta t^3) \tag{2.1.18}$$

Clearly, the truncation error depends on $y(t)$ and this would not seem very useful. However, we are interested in the *scaling* of $\tau_{\Delta t}$ with time stepping, i.e. doubling or halving the time steps. Since $y(t)$ is sufficiently smooth, we can expect that higher derivatives are smooth as well, and thus bounded, so that the truncation error goes to zero as $\Delta t \to 0$ and we write

$$\tau_{\Delta t} = \mathcal{O}(\Delta t) \tag{2.1.19}$$

The order of accuracy is determined as the leading error term in the expansion of the truncation error, i.e. the lowest power of Δt. So, the order of accuracy of explicit Euler is one. As a whole, we expect that the global error and truncation error is to be roughly the same in magnitude.

A scheme is said to be **consistent** with the associated differential equation, if and only if

$$\lim_{\Delta t \to 0} \tau_{\Delta t} = 0$$

From Eq. (2.1.19) we conclude that explict Euler is consistent with the test equation, $dy/dt = \lambda y$. In other words, by taking the limit $\Delta t \to 0$ Eq. (2.1.16) goes over to the test equation. This viewpoint of consistency is nothing else than the consequence of the definition of the derivative, see Eqs. (2.1.2)−(2.1.4). Alternatively, scheme, Eq. (2.1.16), is said to be consistent of order one. This clarifies the notion of the **order of consistency**.

In a similar fashion we can show that the trapezoidal rule is consistent of order two, while implicit Euler is first order accurate, just like its explicit counterpart. Their truncation errors are given by

$$\theta = \frac{1}{2}: \quad \tau_{\Delta t} = -\frac{1}{12}\Delta t^2 y'''(t) + \mathcal{O}(\Delta t^3)$$

$$\theta = 1: \quad \tau_{\Delta t} = -\frac{1}{2}\Delta t y''(t) + \mathcal{O}(\Delta t^2)$$

Exercise 2.1.3 Verify these truncation errors.

The question whether it is appropriate to employ a higher order scheme is a matter of a cost benefit analysis. We return to our first order ODE, Eq. (2.1.1a). Next, let us consider the explicit Euler and Crank-Nicolson schemes, Eq. (2.1.6) and Eq. (2.1.9), respectively. They are first order and second order accurate, respectively. Not only the order of accuracy is an issue, also the amount of work per time step need to be taken into account. For each time step, explicit Euler needs to carry out one evaluation of $f(y)$, while the Crank-Nicolson scheme has two $f-$evaluations per time step. Hence, for an arbitrary function f, we can carry out approximately twice the Euler steps against one Crank-Nicolson step for the same amount of work. However, the Crank-Nicolson scheme is preferable over explicit Euler as it is profitable. This is illustrated in Table 2.1, which shows the results of time integration with $T = 1$ s, whereby the same global error is required for both considered schemes. Particularly, when very high accurate solutions are desired, the superiority of higher order accurate methods is spectacular.

Table 2.1: Cost benefit analysis of the application of explicit Euler and Crank-Nicolson schemes for time integration of $y' = f(y)$ with $T = 1$ s.

	Δt (s)	number of steps	$f-$evaluation per step	total amount of work
$e^n = 10^{-1}$				
explicit Euler	0.1	10	1	10
Crank-Nicolson	0.3	3	2	6
$e^n = 10^{-2}$				
explicit Euler	0.01	100	1	100
Crank-Nicolson	0.1	10	2	20
$e^n = 10^{-4}$				
explicit Euler	0.0001	10000	1	10000
Crank-Nicolson	0.01	100	2	200

Stability

For large Δt, we can take limits directly to obtain (recall $\lambda < 0$)

$$\theta = 0 : \quad \lim_{\Delta t \to \infty} \frac{y^{n+1}}{y^n} = \lim_{\Delta t \to \infty} 1 + \lambda \Delta t = -\infty$$

$$\theta = 1 : \quad \lim_{\Delta t \to \infty} \frac{y^{n+1}}{y^n} = \lim_{\Delta t \to \infty} (1 - \lambda \Delta t)^{-1} = 0$$

$$\theta = \frac{1}{2} : \quad \lim_{\Delta t \to \infty} \frac{y^{n+1}}{y^n} = \lim_{\Delta t \to \infty} (1 + \frac{1}{2}\lambda \Delta t)(1 - \frac{1}{2}\lambda \Delta t)^{-1} = -1$$

whereas for the exact solution, we have

$$\lim_{\Delta t \to \infty} \frac{y(t + \Delta t)}{y(t)} = \lim_{\Delta t \to \infty} e^{\lambda \Delta t} = 0$$

This explains the behaviour seen in Figure 2.4.

We now consider the results of *repeated* time stepping for each of the schemes. Our primary interest is understanding instability that arises from repeated time stepping. This can be readily established from our previous one-step analysis. Each scheme considered above can be written as a **recurrent relation**,

$$y^{n+1} = ry^n \tag{2.1.20}$$

so that, by induction, we have

$$y^n = r^n y^0 = r^n y_0 \tag{2.1.21}$$

(Note that some of the superscripts are powers while others are indices!) Factor r is called the **amplification factor**. This factor equals the ratio y^{n+1}/y^n and indicates the growth ($|r| > 1$) or decay ($|r| < 1$) of the numerical solution. If $|r| = 1$ then the solution is said to be neutral. Note that if $r < 0$, the solution y^n oscillates. Depending on the amplification factor, the evolution of the numerical solution due to the repetition of time stepping is illustrated in Figure 2.5. It follows that as long as $|r| \leq 1$, y^n will be bounded as $n \to \infty$.

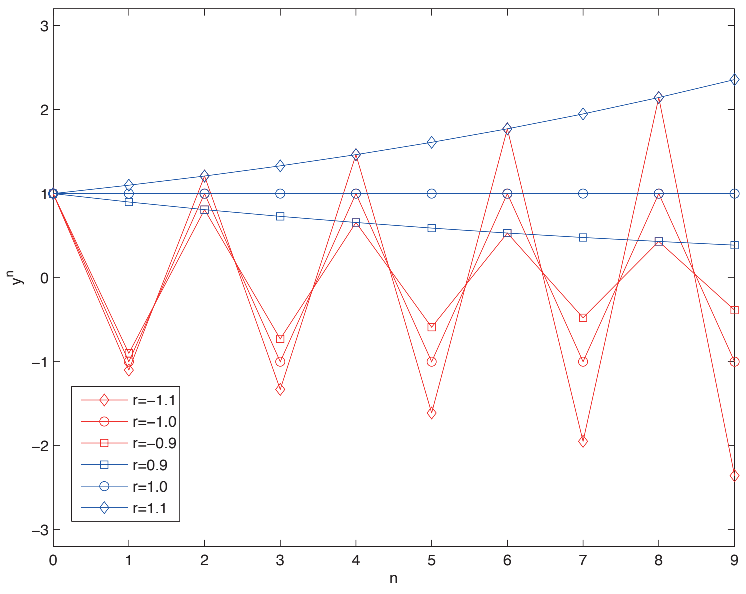


Figure 2.5: Advancing of the solution in time due to repeated time stepping depending on the amplification factor.

In fact, $y^n \to 0$ as $n \to \infty$ for $-1 < r < 1$. In other words, y^n will diverge, or will become unbounded, as $n \to \infty$, as soon as $|r| > 1$, even though the exact solution, Eq. (2.1.12), decays to zero as $t \to \infty$. This is the essence of the numerical instability. Thus, stability requires that the numerical solution *remains bounded* as the time stepping is *repeated an infinite number of times* for a *fixed* time step. This is called **absolute stability**.

The amplification factors for the considered schemes are

$$\begin{aligned} \theta = 0 : &\quad r = 1 + \lambda \Delta t \\ \theta = 1 : &\quad r = (1 - \lambda \Delta t)^{-1} \\ \theta = \frac{1}{2} : &\quad r = (1 + \frac{1}{2}\lambda \Delta t)(1 - \frac{1}{2}\lambda \Delta t)^{-1} \end{aligned} \tag{2.1.22}$$

These are the quantities plotted in Figure 2.4.

From the above expressions and Figure 2.4, one can easily show $|r| \leq 1$ for all Δt for implicit Euler and Crank-Nicolson schemes. These schemes are said to be **unconditionally stable** because there is no requirement on Δt for stability. However, for the explicit Euler scheme it is evident that $|r|$ can be larger than 1 for a certain value of Δt. We can derive the **stability limit** for the explicit Euler scheme by requiring $|r| \leq 1$:

$$-1 \leq 1 + \lambda \Delta t \leq 1 \Rightarrow -2 \leq \lambda \Delta t \leq 0$$

The latter inequality is always true (for $\lambda < 0$) while the first one results in a time step restriction

$$\boxed{\Delta t \leq \frac{-2}{\lambda}} \tag{2.1.23}$$

The explicit Euler scheme is said to be **conditionally stable** because condition, Eq. (2.1.23), must be satisfied for stability. Note that when instability develops, i.e $r < -1$, it is oscillatory in time with y^{n+1} and y^n of opposite sign. This is one of the key signatures of numerical instabilities.

Exercise 2.1.4 Consider the $\theta-$scheme, Eq. (2.1.13). Show that this scheme is unconditionally stable when $1/2 \leq \theta \leq 1$.

It is very important to realised that there is no *direct* relation between the (final) recurrent relation, Eq. (2.1.20), and the (original) differential equation, Eq. (2.1.1a), describing a physical process. In fact, the recurrent relation is purely governed by algebraic rules. And these rules have nothing to do with the physical laws. However, these rules may help us to understand why the outcomes of a simulation sometimes behave weird or seem to have no physical justification. To a certain extent, there must be some resemblance between the physical reality and the phenomena to be simulated. This implies an extra necessary condition solely for a numerical recipe in order to be useful for simulation. An example is the stability as considered before. Another example is the oscillation of numerical solutions when $r < 0$. This will produces negative solutions, $y^n < 0$, which in some cases, from a physical point of view, are not realistic. For instance, when the solution $y(t)$ represents the concentration of a dissolved matter. This implies another requirement on the numerical recipe for the prevention of numerical oscillation.

2.1.3 Two-step schemes

So far we have focused entirely on so-called **one-step** schemes. Only one step is made to march in time, i.e. the solution on the new time step $n+1$ is solely based on the solution of the previous time step n. One way to get higher order accuracy is to use a **multi-step** method that involves other previous values, see Lambert (1991). An example of a **two-step** scheme is the **midpoint rule**,

$$\frac{y^{n+1}-y^{n-1}}{2\Delta t} = f(y^n) \tag{2.1.24}$$

in which two previous steps n and $n-1$ are involved for the computation of the solution on the new step $n+1$. Another example is the so-called implicit **BDF** scheme which is given by

$$\frac{3y^{n+1}-4y^n+y^{n-1}}{2\Delta t} = f(y^{n+1}) \tag{2.1.25}$$

One difficulty with using a multi-step scheme is that we need the starting values y^0, y^1, y^2, $\cdots$ before we can begin to apply this scheme. The value $y^0 = y_0$ is known from the initial data, but the other values are not and must typically be generated by some other one-step scheme. For instance, if we want to use the midpoint rule, Eq. (2.1.24), then we need to generate y^1 before we start to apply Eq. (2.1.24) with $n=1$. We can obtain y^1 from y^0 using the explicit Euler scheme. Since this Euler scheme will be applied only once and has a local error of $\mathcal{O}(\Delta t^2)$, the overall second order accuracy will not be affected as the truncation error of the midpoint rule is $\mathcal{O}(\Delta t^2)$, which will be shown below.

Let us consider the test equation, Eq. (2.1.11). The midpoint rule applied to this equation reads

$$\frac{y^{n+1}-y^{n-1}}{2\Delta t} = \lambda y^n \tag{2.1.26}$$

The truncation error is obtained by

$$\tau_{\Delta t} = \frac{y(t+\Delta t)-y(t-\Delta t)}{2\Delta t} - \lambda y(t) \tag{2.1.27}$$

Assuming $y(t)$ is smooth enough and expanding in Taylor series gives

$$y(t\pm\Delta t) = y(t) \pm \Delta t y'(t) + \frac{1}{2}\Delta t^2 y''(t) \pm \frac{1}{6}\Delta t^3 y'''(t) + \cdots$$

Substituting in Eq. (2.1.27) yields

$$\tau_{\Delta t} = \frac{1}{6}\Delta t^2 y'''(t) + \mathcal{O}(\Delta t^4)$$

so that the midpoint rule is second order in Δt and consistent with the test equation.

Exercise 2.1.5 Verify the truncation error of the midpoint rule.

Exercise 2.1.6 Derive the truncation error of the BDF scheme, Eq. (2.1.25), with $f(y) = \lambda y$.

The stability analysis for the midpoint rule is rather different from the one-step schemes. Eq. (2.1.26) can be rewritten as

$$y^{n+1} - 2\lambda \Delta t y^n - y^{n-1} = 0 \tag{2.1.28}$$

This is another example of a recurrent relation, which is of second order. (Eq. (2.1.20) is an example of a first order recurrent relation.) To find a solution, we assume that a solution is of the form Eq. (2.1.21). Substitution gives

$$r^{n+1} - 2\lambda \Delta t r^n - r^{n-1} = 0$$

Next, division by r^{n-1} gives ($r \neq 0$)

$$r^2 - 2\lambda \Delta t r - 1 = 0 \tag{2.1.29}$$

This equation is called the **characteristic equation** belonging to the midpoint rule. This characteristic equation is a quadratic one and has two *different* roots:

$$r_1 = \lambda \Delta t + \sqrt{1 + \lambda^2 \Delta t^2}, \quad r_2 = \lambda \Delta t - \sqrt{1 + \lambda^2 \Delta t^2} \tag{2.1.30}$$

Since Eq. (2.1.28) is linear in y^n the final solution of Eq. (2.1.28) is thus the superposition of two different elementary solutions,

$$y^n = \alpha r_1^n + \beta r_2^n$$

The constants α and β can be determined by means of the starting values y^0 and y^1.

Exercise 2.1.7 Find the constants α and β.

For small Δt, the roots r_1 and r_2, Eq. (2.1.30), are expanded using the Taylor series (see Exercise 2.1.2), as follows,

$$\begin{aligned} r_1 &= 1 + \lambda \Delta t + \frac{1}{2}\lambda^2 \Delta t^2 - \frac{1}{8}\lambda^4 \Delta t^4 + \mathcal{O}(\Delta t^6) \\ r_2 &= -1 + \lambda \Delta t - \frac{1}{2}\lambda^2 \Delta t^2 + \frac{1}{8}\lambda^4 \Delta t^4 + \mathcal{O}(\Delta t^6) \end{aligned} \tag{2.1.31}$$

By comparing these roots with Eq. (2.1.15), we conclude that solution r_1^n approaches the exact solution, $e^{\lambda n \Delta t}$, with an error of $\mathcal{O}(\Delta t^2)$. This root is called the **principal root**. On the other hand, root r_2 is called the **spurious root** since it does not approach the exact solution at all. Again, this has nothing to do with the test equation. This spurious root is just introduced by the chosen numerical scheme. In this case the midpoint rule has two distinct roots, while the test equation only has one solution mode.

Generally, a multi-step method will generate equal amount of roots. So, a kth-step scheme, involving $k+1$ distinct time levels, will produces k roots. If this scheme is consistent with Eq. (2.1.1a), then one of the roots should represent an approximation to the exact solution. This root is thus the principal root. The other roots are spurious roots and represent a *non-physical* time behaviour of the numerical solution introduced by the scheme. Spurious roots are often strongly influencing the stability of the method. For this reason, control of spurious roots is important.

For stability of the midpoint rule, we must require that both roots r_1 and r_2 must be smaller in absolute value than or equal 1, i.e. $|r_1| \leq 1$ and $|r_2| \leq 1$. From Eqs. (2.1.31) it is clear that it is possible to choose a time step Δt such that $|r_1| \leq 1$, but impossible for the second root as $|r_2| > 1$ for all $\Delta t > 0$ (recall $\lambda < 0$). Hence, the midpoint rule is **unconditionally unstable** for the test equation, Eq. (2.1.11).

To see what happens when we apply the midpoint rule to find an approximate solution of $dy/dt = -y$ with $y(0) = 1$, we consider the time integration till $T = 10$ s as shown in Figure 2.6. In the range $0 \leq t \leq 5$ s the solution resembles to that of the exact one

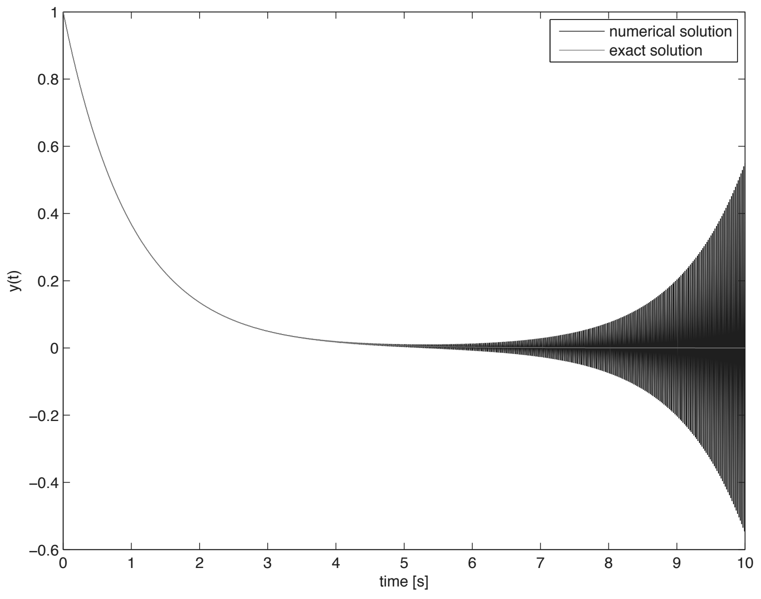


Figure 2.6: Approximate solution of $y' = -y$ obtained with the midpoint rule with $\Delta t = 0.01$ s, compared with the exact one.

and looks stable as well. But afterwards, the solution becomes quickly unbounded. This example shows that despite the consistency a numerical scheme can be rather useless for some practical applications. The midpoint rule is an example which cannot be applied to a single test equation without having stability problems. However, in the next section, we shall see that the midpoint rule might be useful for a multiple system of first ordinary differential equations.

Exercise 2.1.8 Carry out a stability analysis for the BDF scheme, Eq. (2.1.25), with $f(y) = \lambda y$. What is your conclusion about the stability of this scheme? Does this scheme have spurious roots?

2.2 System of first order differential equations

So far we have examined the numerical solution for a scalar ODE with one unknown. In this section we now consider a *linear* system of first order ODEs given by

$$\frac{d\vec{y}}{dt} = A\,\vec{y} + \vec{b}(t) \tag{2.2.1a}$$

$$\vec{y}(0) = \vec{y}_0 \tag{2.2.1b}$$

where A is a given $m \times m$ matrix with real, constant elements and $\vec{b}(t)$ is a given vector with m real elements as function of time. If $\vec{b} \equiv \vec{0}$, then the system is called **homogeneous**. Again, we assume that the right-hand side of Eq. (2.2.1a) is Lipschitz continuous in any norm $\|\cdot\|$ (see Section 2.2.2), which means that it is sufficiently smooth. We seek for the solution $\vec{y}$ of the system with m real unknowns, whereas the system is initialized with the given vector $\vec{y}_0$ containing m initial data.

As an example, we consider a **spring-mass system** of which the displacement x of the mass m attached to a spring of spring constant k and under influence of an external sinusoidally force of period T_f is governed by the Newton's second law,

$$m\frac{d^2x}{dt^2} = -kx + \sin(\frac{2\pi\,t}{T_f})$$

which can be rewritten as a first order system of two equations by introducing the velocity $v = dx/dt$,

$$\frac{d}{dt}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{pmatrix}\begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{m}\sin(\omega t) \end{pmatrix}$$

with $\omega = 2\pi/T_f$. By computing eigenvalues with

$$\det\begin{pmatrix} -\lambda & 1 \\ -\frac{k}{m} & -\lambda \end{pmatrix} = \lambda^2 + \frac{k}{m} = 0$$

giving

$$\lambda_\pm = \pm \mathrm{i}\sqrt{\frac{k}{m}}$$

we find that this system is an **undamped** one as the eigenvalues are purely imaginary ($\mathrm{i}^2 = -1$), while its **eigen frequency**, $T = 2\pi/\,Im(\lambda) = 2\pi\sqrt{m/k}$ indicates the period of oscillation which is independent of both the amplitude and gravitational acceleration. Generally, the time scale of this harmonic motion is much smaller than the time scale of the forcing T_f.

Exercise 2.2.1 Consider the above spring-mass system, this time with a frictional force added,

$$m\frac{d^2x}{dt^2} = -kx - a\frac{dx}{dt} + \sin(\frac{2\pi\, t}{T_f})$$

with a the friction coefficient. Calculate the eigenvalues of this system and explain why this system is to be considered as **damped**.
(*Hint*: see Section 2.2.1.)

2.2.1 The solution of system of first order ODEs

Let us consider a homogeneous, linear system $d\vec{y}/dt = A\vec{y}$. We assume that this matrix A with rank m has m *distinct* eigenvalues. Furthermore, it has a complete set of m linearly independent eigenvectors $\vec{e}_j$ satisfying

$$A\vec{e}_j = \lambda_j \vec{e}_j\,, \quad j = 1, \ldots, m$$

The eigenvalues may be complex, i.e. $\lambda_j = \alpha_j + \beta_j\mathrm{i}$ with $\alpha_j = Re(\lambda_j)$ the real part and $\beta_j = Im(\lambda_j)$ the imaginary part. Note that A contains *real* elements only! As a consequence, the so-called **complex conjugate** of λ_j, denoted as $\overline{\lambda}_j = \alpha_j - \beta_j\mathrm{i}$, is also an eigenvalue of A. The general solution is given by

$$\vec{y}\,(t) = \sum_{j=1}^{m} \gamma_j \vec{e}_j\, e^{\lambda_j t}$$

where $\gamma_j\,, j = 1, \ldots, m$ are constants which are determined by means of the initial data.

Exercise 2.2.2 Consider the next system of equations

$$\frac{d\vec{y}}{dt} = \begin{pmatrix} -5 & 1 \\ 4 & -2 \end{pmatrix} \vec{y}\,, \quad \vec{y}\,(0) = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Show that the solution of this system equals

$$\vec{y}\,(t) = \frac{3}{5}e^{-t}\begin{pmatrix} 1 \\ 4 \end{pmatrix} - \frac{2}{5}e^{-6t}\begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

Just like the single case, the solution of the considered system should not be sensitive to small disturbances in e.g. initial data. The time evolution of the solution follows directly from

$$e^{\lambda_j t}\,, \quad j = 1, \ldots, m$$

Since the eigenvalues are generally complex, this implies that the time evolution of each solution mode is like (see Appendix B.1 for details)

$$e^{\lambda_j t} = e^{(\alpha_j + \beta_j \mathrm{i})t} = e^{\alpha_j t}(\cos\beta_j t + \mathrm{i}\sin\beta_j t)$$

If $\beta_j \neq 0$ then the solution behaves like a harmonic motion. Three types of oscillation can be distinguished:

- If at least one λ_j with $Re(\lambda_j) = \alpha_j > 0$, then we have a growing harmonic motion. The solution becomes unbounded as $t \to \infty$. In this case the system is said to be **unstable** and is sensitive to small disturbances.
- If $Re(\lambda_j) = \alpha_j = 0\,, \forall j = 1, \ldots, m$, then the oscillation is **undamped**. The solution is bounded and thus we have a **stable** system.
- If $Re(\lambda_j) = \alpha_j < 0\,, \forall j = 1, \ldots, m$, then the oscillation is **underdamped**. The solution will disappear as $t \to \infty$. The system is said to be **absolutely stable**.

If $\beta_j = 0$ then the solution is non-oscillatory. This solution is called **overdamped** and thus stable if all the distinct (real) eigenvalues are negative. A special case is **critical damping** in which at least one eigenvalue is repeated. A critically damped system returns to zero amplitude as quickly as possible without oscillating. In this case the solution decays exponentially. One may think of a surface-mounted door closer attached to the hinge door in public buildings. Note that a critically damped system will return to zero more quickly than an overdamped or underdamped system, see Figure 2.7.

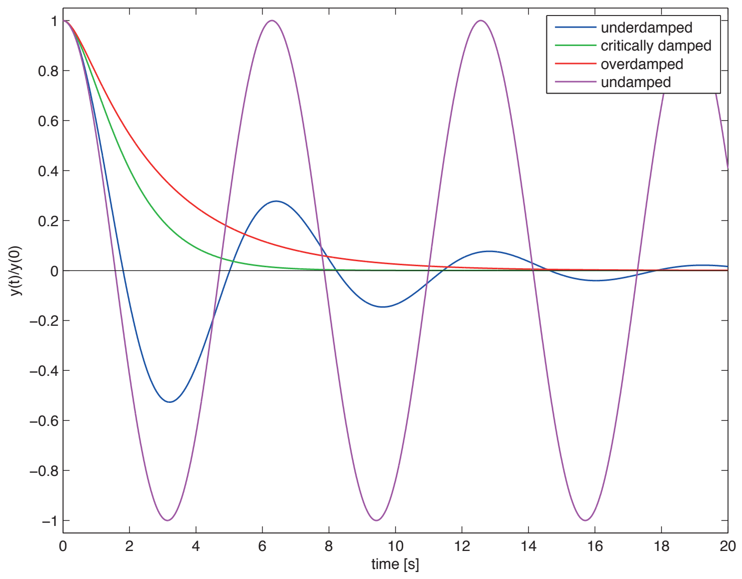


Figure 2.7: Different damping mechanisms for the homogeneous solution.

Hence, on physical grounds, we are only interested in system of ODEs with

$$Re(\lambda_j) \leq 0\,, \quad j = 1, \ldots, m$$

As an example we consider a system of ODEs with

$$A = \begin{pmatrix} -4 & -17 \\ 2 & 2 \end{pmatrix}$$

Its eigenvalues are $\lambda_{1,2} = -1 \pm 5\text{i}$. So, this system is (absolutely) stable.

Exercise 2.2.3 Show that the system of ODEs with

$$A = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

is unstable.

Let us consider the spring-mass system as given in Exercise 2.2.1 with mass of $m = 1$ kg and spring constant of $k = 100$ N/m. The initial conditions are determined as $x(0) = 1$ cm and $x'(0) = 0$ m/s. Without damping and external forcing, the solution of the system is depicted in Figure 2.8. It represents a **free motion** of period $2\pi\sqrt{m/k} = 0.628$ s. This is

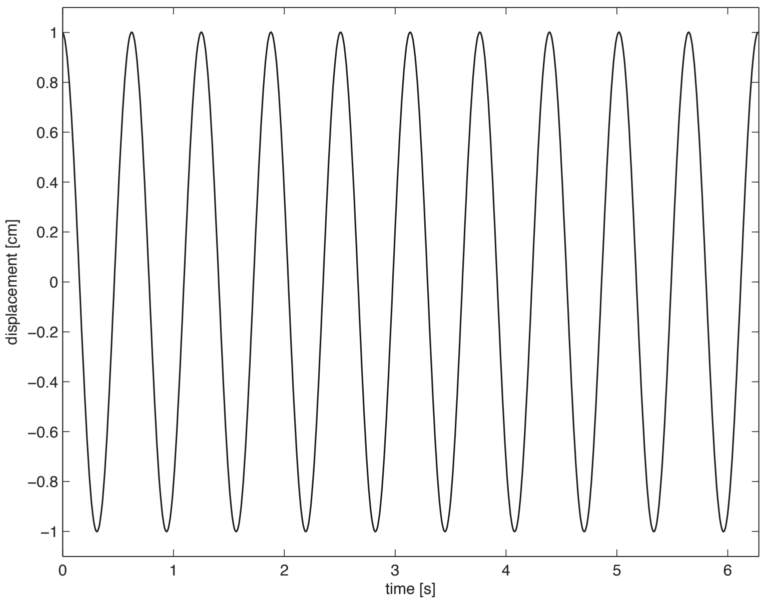


Figure 2.8: Time series of free motion of the spring-mass system.

usually referred to as the natural or eigen period of the motion. Note that the amplitude of this free motion is determined by the initial displacement.

When adding an external force with period of $T_f = 4$ s, we obtain a solution as shown in Figure 2.9, which is the result of the principle of superposition. (The considered ODE is linear.) This resultant solution represents a **forced motion** and is the sum of the *homogeneous* solution representing the eigen motion and the *particular* solution describing the effect of the external forcing. This particular solution is given by

$$x_p = \frac{f_0}{\sqrt{m^2\left(\omega_0^2 - \omega^2\right)^2 + a^2\omega^2}} \sin(\omega t - \varphi)$$

where f_0 is the amplitude of the external force, $\omega_0 = \sqrt{k/m}$ is the eigen frequency of the system, $\omega = 2\pi/T_f$ is the frequency of the force, a is the friction coefficient, and

$$\cos\varphi = \frac{m\left(\omega_0^2 - \omega^2\right)}{\sqrt{m^2\left(\omega_0^2 - \omega^2\right)^2 + a^2\omega^2}}$$

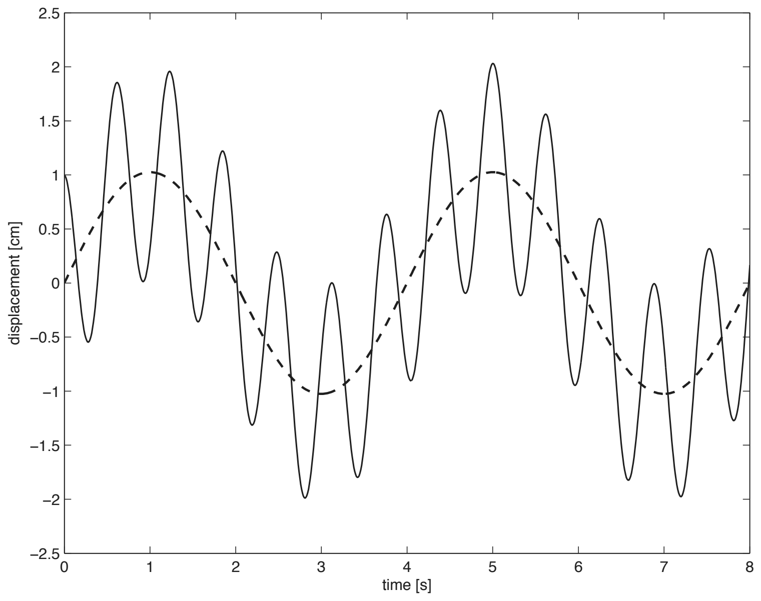


Figure 2.9: Time series of forced motion of the spring-mass system. The dashed line represents the effect of the external force.

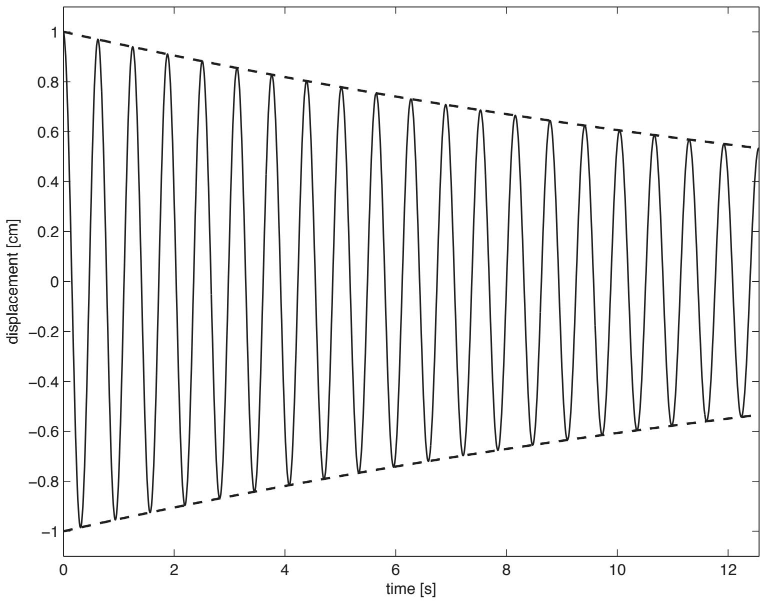


Figure 2.10: Time series of damped vibration of the spring-mass system. The dashed line indicates the decay due to the frictional force.

Here, $f_0 = 1$ m (see Exercise 2.2.1), whereas the amplitude of the particular solution is approximately 1 cm.

Exercise 2.2.4 Verify this amplitude of the particular solution.

The two observed periodic motions usually have different frequencies. The period of the force is typically much larger than the eigen period. In fact, it can be so large that the system can be regarded as stiff (see Section 2.2.4). However, if $\omega \approx \omega_0$, then **resonance** will occur, i.e. the motion will become unbounded. The eigen mode is due to the initial conditions whereas the particular solution is due to the external force. The physical interpretation of the resultant motion is that it represents the response of the spring-mass system to the external force, while the effect of the initial conditions persists for all time.

Let us consider the case where we include the effect of damping but without external forcing. The corresponding equation is given by

$$m\frac{d^2x}{dt^2} + a\frac{dx}{dt} + kx = 0$$

The roots of the associated characteristic equation are found to be

$$\lambda_\pm = -\frac{a}{2m} \pm \sqrt{\frac{a^2}{4m^2} - \frac{k}{m}}$$

With a friction coefficient of, for instance, $a = 0.1$ kg/s, the eigenvalues are thus

$$\lambda_\pm \approx -0.05 \pm 10\mathrm{i}$$

which means that the harmonic motion is an underdamped one; see Figure 2.10. This motion dies out as time proceeds. The time scale of this decay is given by

$$T \sim \frac{2\pi}{|\,Re(\lambda)|} = 2\pi\,\frac{2m}{a} \approx 125.7\,\mathrm{s}$$

which is much longer than the eigen period.

Next we add the external force to the damped system. The motion of this system is displayed in Figure 2.11. Again, the homogeneous solution is superimposed on the particular solution induced by external forcing. Recall that the homogeneous solution has a much shorter time scale than that of the force. As expected, this homogeneous solution disappears with increase in time. This is to say that the energy put into the spring-mass system by the initial displacement is dissipated by the frictional force until it is gone. This is illustrated in Figure 2.12. What remains is the motion describing the response of the system to the external force. This motion is usually not affected by damping. This is typical of underdamping. This motion is often called the **steady state** motion, while the homogeneous solution is called the **transient** solution. The transient solution allows us to satisfy the imposed initial conditions, whereas it takes some time before the influence of

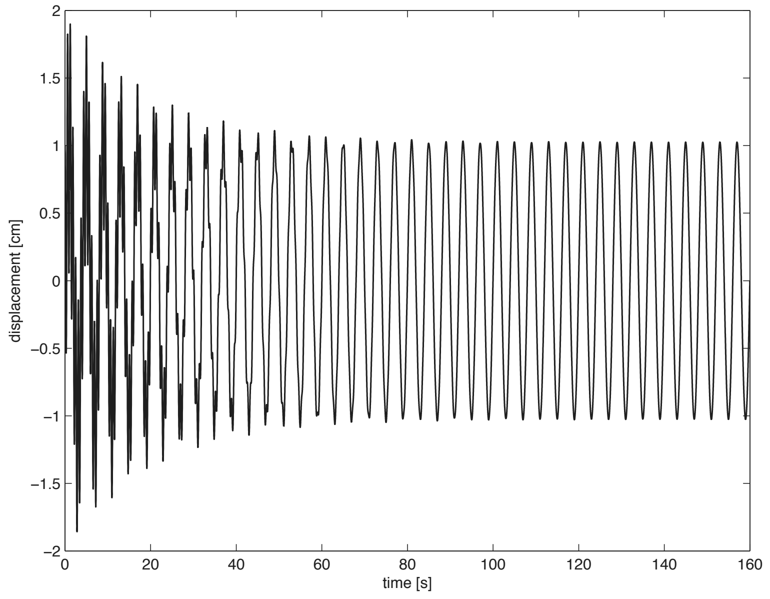


Figure 2.11: Time series of forced motion with underdamping of the spring-mass system. The effect of initial displacement vanish after about 120 s.

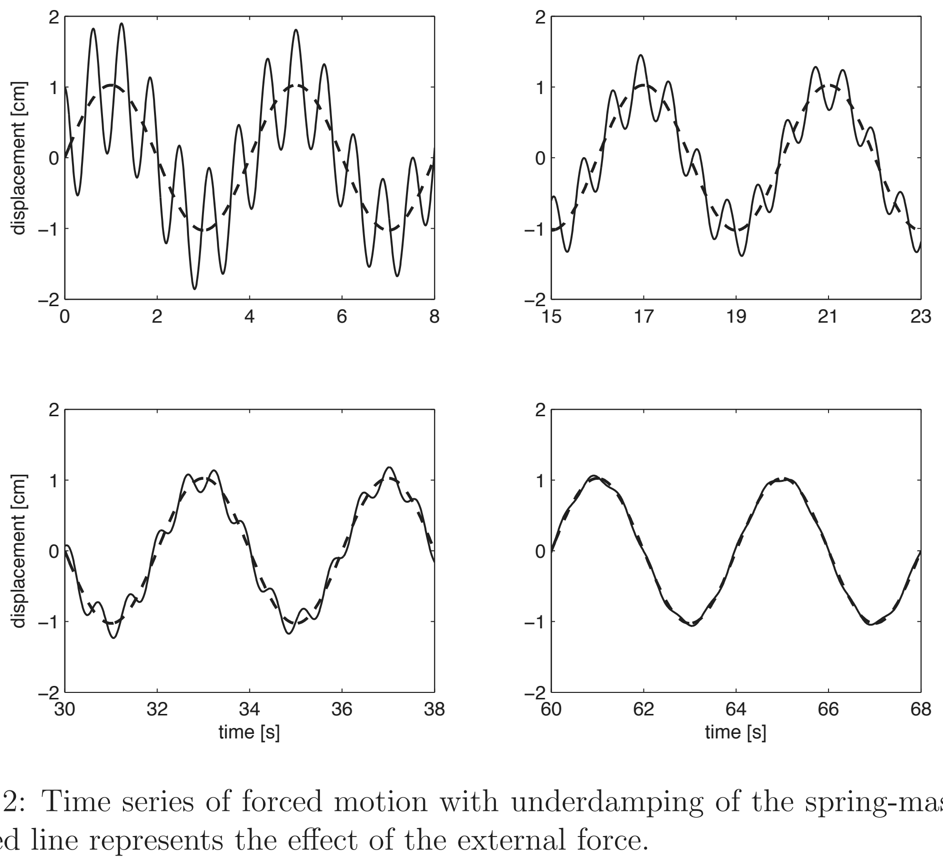


Figure 2.12: Time series of forced motion with underdamping of the spring-mass system. The dashed line represents the effect of the external force.

these conditions on the motion vanish. Such duration is called the **spin-up time**. Without damping this spin-up time would be infinite.

Let us enlarge the friction coefficient. We consider the case with $a = 20$ kg/s. This time we have a double eigenvalue (the discriminant is zero) and equals $\lambda = -10$. The corresponding damping period is $2\pi/10 \approx 0.628$ s, which equals the eigen period. This is critical damping and is the fastest damping possible (see Figure 2.7). In this case the free motion approaches to zero exponentially without oscillation (the displacement is initially started at 1 cm), whereas the steady state motion is barely damped; see Figure 2.13.

Exercise 2.2.5 Verify the eigenvalue.

Exercise 2.2.6 Why is in this case the damping period equal to the eigen period?

Exercise 2.2.7 Calculate the amplitude of the steady state solution and compared this to the one in the frictionless case (see Exercise 2.2.4).

Finally, we consider the case of overdamping. We further increase the friction coefficient: $a = 100$ kg/s. The two eigenvalues are real and negative

$$\lambda_+ \approx -1.01\,, \quad \lambda_- \approx -98.99$$

The decay period is approximately 6.2 s. So with more damping the approach to zero amplitude is more slowly than for the case of critical damping. Figure 2.14 depicts the solution behaviour which shows that not only the transient solution is damped but also the steady state motion is significantly damped. This is typical of overdamping.

Exercise 2.2.8 Verify the eigenvalues and the damping period.

2.2.2 Consistency and truncation error

Just as for the single first order ODE, Eq. (2.1.1a), we can apply all the numerical schemes, considered in this chapter, in exactly the same way to any system of first order ODEs. For instance, approximating Eq. (2.2.1a) using explicit Euler gives

$$\frac{\vec{y}^{n+1} - \vec{y}^{n}}{\Delta t} = A\,\vec{y}^{n} + \vec{b}(t_n) \tag{2.2.2}$$

Also, the notions consistency and order of accuracy can easily be extended to *linear* systems of ODEs. Both global and truncation errors become vectors (instead of scalars) and they are measured in some suitable norm. Hence, the requirement for consistency now becomes

$$\lim_{\Delta t \to 0} \|\vec{\tau_{\Delta t}}\| = 0$$

while a numerical scheme applied to Eq. (2.2.1a) is said to be convergent if

$$\lim_{\substack{\Delta t \to 0 \\ n = \frac{t}{\Delta t}}} \|\vec{e}^{n}\| = 0$$

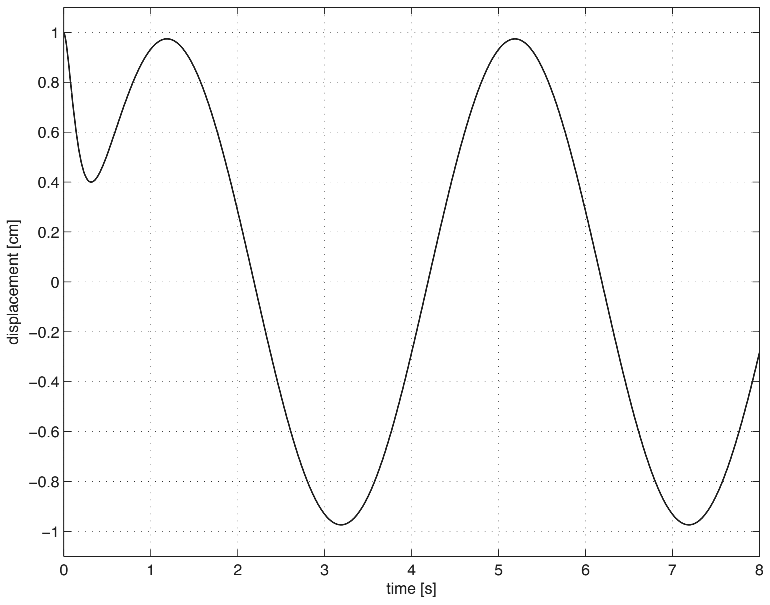


Figure 2.13: Time series of forced motion with critical damping of the spring-mass system. The effect of initial displacement dies out very fast.

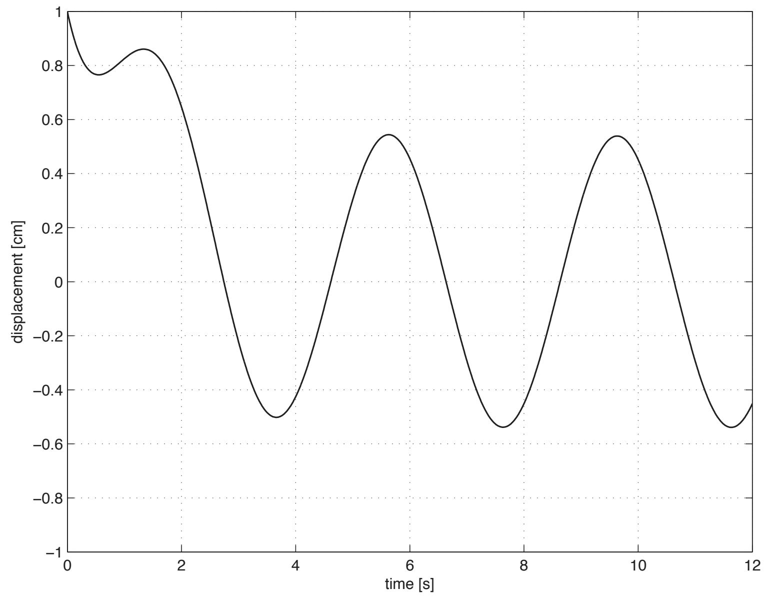


Figure 2.14: Time series of forced motion with overdamping of the spring-mass system. The steady state motion is substantially damped.

for all $t \in (0, T]$. The norm can be chosen as an L^p-norm, given by

$$\|\vec{v}\|_p = \left(\sum_{j=1}^{m} |v_j|^p \right)^{1/p}$$

or a max-norm or $L^\infty-$norm, given by

$$\|\vec{v}\|_\infty = \max\{|v_1|, |v_2|, \cdots, |v_m|\}$$

Recalling the spring-mass system, the explicit Euler scheme reads

$$\begin{aligned} \frac{x^{n+1} - x^n}{\Delta t} &= v^n \\ \frac{v^{n+1} - v^n}{\Delta t} &= -\frac{k}{m} x^n + \frac{1}{m} \sin(\omega\, n\Delta t) \end{aligned} \tag{2.2.3}$$

Next, simply fill in the Taylor series expansions for a solution of Eq. (2.2.1a) at $t = n\Delta t$,

$$\begin{aligned} x^{n+1} &\to x(t + \Delta t) = x(t) + \Delta t\; x'(t) + \mathcal{O}(\Delta t^2), \\ v^{n+1} &\to v(t + \Delta t) = v(t) + \Delta t\; v'(t) + \mathcal{O}(\Delta t^2), \end{aligned}$$

to determine the truncation error (for both equations!)

$$\begin{aligned} \frac{x(t) + \Delta t\; x'(t) + \mathcal{O}(\Delta t^2) - x(t)}{\Delta t} - v(t) &= x'(t) - v(t) + \mathcal{O}(\Delta t) \\ &= \mathcal{O}(\Delta t) \end{aligned}$$

and

$$\begin{aligned} \frac{v(t) + \Delta t\; v'(t) + \mathcal{O}(\Delta t^2) - v(t)}{\Delta t} + \frac{k}{m}x(t) - \frac{\sin(\omega t)}{m} &= v'(t) + \frac{k}{m}x(t) - \frac{\sin(\omega t)}{m} + \mathcal{O}(\Delta t) \\ &= \mathcal{O}(\Delta t) \end{aligned}$$

Again, explicit Euler is first order accurate. Here, we did not calculate the truncation error *explicitly*, but simply gathered all terms of at least order one in Δt, and showed that all other terms vanish using the fact that we filled in a solution of the system of ODEs. Because the truncation error is at least order one, and thus vanishes for $\Delta t \to 0$, the scheme is consistent with the system of ODEs.

Exercise 2.2.9 Compute the leading terms of the truncation error in the example above. Compare this with Eq. (2.1.18).

Exercise 2.2.10 Write down the Crank-Nicolson scheme for the spring-mass system and derive the truncation error. What are your conclusions?

2.2.3 Stability and stability regions

Generally, the system $d\vec{y}/dt = A\vec{y}$ is considered to be **coupled**, since for each unknown, knowledge of some other unknowns are required in order to find this unknown. To demonstrate stability analysis for this system, we **diagonalize** matrix A so that the equations are then to be **decoupled**. Let $H = [\vec{e}_1\ \vec{e}_2\ \dots\ \vec{e}_m]$ be the matrix of eigenvectors and $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_m)$ be the diagonal matrix containing eigenvalues of A. Then we have $AH = [\lambda_1\vec{e}_1\ \lambda_2\vec{e}_2\ \dots\ \lambda_m\vec{e}_m]$. This matrix can also be obtained by multiplying H by Λ, i.e. $H\Lambda$. Hence,

$$H\Lambda = AH \Leftrightarrow \Lambda = H^{-1}AH$$

Matrices A and Λ are called **similar** and the transformation between these matrices with different bases is called the **similarity transformation**. Now let $\vec{y} = H\vec{c}$. Substituting this into $d\vec{y}/dt = A\vec{y}$ gives

$$H\frac{d\vec{c}}{dt} = AH\vec{c}$$

or

$$\frac{d\vec{c}}{dt} = H^{-1}AH\vec{c} = \Lambda\vec{c}$$

This is a diagonal system of equations that *decouples* into m independent scalar equations, one for each component of $\vec{c}$. The jth such equation is

$$\frac{dc_j}{dt} = \lambda_j c_j$$

A numerical scheme applied to $d\vec{y}/dt = A\vec{y}$ can also be decoupled in the same manner. For example, if we apply explicit Euler, then we have

$$\vec{y}^{n+1} = \vec{y}^n + \Delta t A\vec{y}^n$$

which, by the same similarity transformation, can be rewritten as

$$\vec{c}^{n+1} = \vec{c}^n + \Delta t\Lambda\vec{c}^n$$

where $\vec{c}^n = H^{-1}\vec{y}^n$. This decouples into m independent numerical recipes, one for each component of $\vec{c}^n$. These take the form

$$c_j^{n+1} = (1 + \lambda_j\Delta t)c_j^n$$

Explicit Euler is thus stable when $|1 + \lambda_j\Delta t| \leq 1\,, \forall j = 1, \dots, m$; otherwise it is unstable. Since λ_j may be complex, it is more common to speak of the **stability region** as a region in the complex h plane with $h \equiv \lambda\Delta t$. Note that there are two parameters Δt and λ, but only their product h matters. The stability region for explicit Euler is the circle of radius 1 centered at point $h = -1$ in the h plane, since within this circle we have $|1 + h| \leq 1$; see Figure 2.15a. So, explicit Euler is stable when $\lambda_j\Delta t$ lies inside the stability region for each eigenvalue of A, $\lambda_j\,, j = 1, \dots, m$. Hence, there is a restriction on the time step which means that explicit Euler is conditionally stable. Since, the matrices A and Λ are similar, this time step restriction is exactly the same when the explicit Euler method is applied to the original system of ODEs as given by Eq. (2.2.1a).

Exercise 2.2.11 Consider the explicit Euler method for the above *undamped* spring-mass system as given by Eq. (2.2.3). Derive the stability limit. What is your conclusion?

For the implicit Euler scheme, based on its amplification factor (see Eq. (2.1.22)), we have the following stability requirement

$$\frac{1}{|1-h|} \leq 1 \Leftrightarrow |1-h| \geq 1$$

so the stability region is the *exterior* of the circle of radius 1 centered at $h = 1$, as shown in Figure 2.15b. Since our interest are the cases with $Re(\lambda_j) \leq 0\,, \forall j$, this scheme is therefore always stable as there is no time step restriction.

For the trapezoidal rule, the stability requirement implies (see Eq. (2.1.22))

$$\frac{|1+\frac{1}{2}h|}{|1-\frac{1}{2}h|} \leq 1$$

It can be shown that this inequality implies $Re(h) \leq 0$. So the stability region is the whole left half-plane as shown Figure 2.15c. Hence, this scheme is unconditionally stable.

Exercise 2.2.12 Verify the last mentioned implication.

The characteristic equation of the midpoint rule reads (see Eq. (2.1.29))

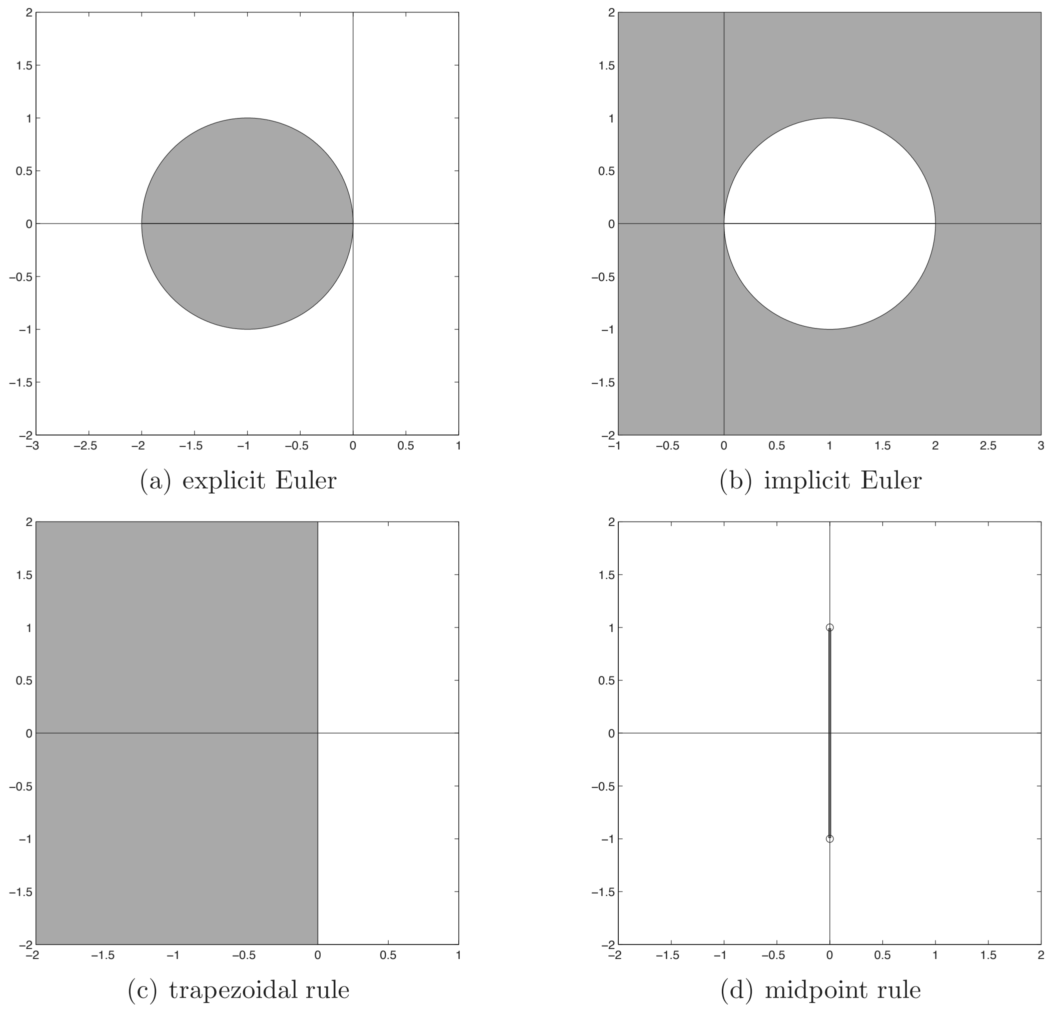
$$r^2 - 2hr - 1 = 0$$

and its roots are

$$r_1 = h + \sqrt{1+h^2}\,, \quad r_2 = h - \sqrt{1+h^2}$$

It can be shown that if h is purely imaginary, i.e. $h = \gamma\mathrm{i}$ with $|\gamma| < 1$, then $|r_1| = |r_2| = 1$ and $r_1 \neq r_2$. In this case each root has a **multiplicity** of 1. If, however, $r_1 = r_2 = r$, i.e. one root, then we say that this root has a multiplicity of 2, which means that it appears twice as a solution of the characteristic equation. This is the case when $\gamma = \pm 1$ or $h = \pm\mathrm{i}$. Generally, when the root has a multiplicity of $k > 1$, i.e. $r_1 = r_2 = \ldots = r_k = r$, then we must require $|r| < 1$. Hence, the stability region of the midpoint rule consists only of the *open* interval from $-\mathrm{i}$ to i on the imaginary axis, as depicted in Figure 2.15d. The midpoint rule is useful only if all the eigenvalues of A in Eq. (2.2.1a) are purely imaginary. For instance, this scheme can be applied to an undamped spring-mass system. Another example is a real, skew symmetric matrix ($A^T = -A$) that arises naturally in the space discretization of hyperbolic partial differential equations, as will be discussed in Chapter 3.

Exercise 2.2.13 Show that the midpoint rule becomes unconditionally unstable when an eigenvalue λ of the system of ODEs is real, $\lambda < 0$, or complex with $Re(\lambda) < 0$.
(*Hint*: reconsider the roots and let h be a complex number with $Re(h) < 0$.)



(a) explicit Euler (b) implicit Euler

(c) trapezoidal rule (d) midpoint rule

Figure 2.15: Stability regions for (a) explicit Euler, (b) implicit Euler, (c) trapezoidal rule and (d) midpoint rule.

If a numerical method is stable for each h with $Re(h) \leq 0$, i.e. its stability region is the whole left half-plane of the complex h plane, then this method is said to be **A-stable**. Thus, any unconditionally stable scheme is A-stable, at least for linear systems of ODEs. Both the Crank-Nicolson and implicit Euler schemes are A-stable, while explicit Euler and the midpoint rule are not. In this respect, the following statements for a multi-step scheme hold (known as the second Dahlquist barrier):

- An explicit multi-step scheme cannot be A-stable.
- The order of accuracy of an A-stable multi-step scheme cannot exceed two.
- The second order A-stable multi-step scheme with the smallest truncation error is the trapezoidal rule.

Exercise 2.2.14 Consider the Simpson rule as given by

$$\frac{c^{n+1} - c^{n-1}}{2\Delta t} = \lambda \left(\frac{1}{6} c^{n+1} + \frac{2}{3} c^n + \frac{1}{6} c^{n-1} \right)$$

Here λ is a complex number. Answer the following questions.

1. Indicate whether this rule is explicit or implicit.
2. Calculate the truncation error.
3. Calculate the stability limit.

Regarding the above statements, what can you conclude about the A-stability of the Simpson rule?

2.2.4 Stiff ODEs

Often the eigenvalues of A in Eq. (2.2.1a) are of vastly different orders of magnitude and so representing very diverse time scales. This leads to **stiff** system of equations. In this context, we consider the so-called **stiffness ratio**, which is defined as the ratio of the eigenvalue with the largest modulus and the eigenvalue with the smallest modulus,

$$S = \frac{\max |\lambda_j|}{\min |\lambda_j|}, \quad j = 1, \ldots, m$$

Generally, we consider a system to be stiff if the stiffness ratio is large, i.e. $S >> 1$. Stiffness is a phenomenon that only play a role in systems of *more than one* equation.

Let us consider the following example

$$\frac{d\vec{y}}{dt} = \begin{pmatrix} -500.5 & 499.5 \\ 499.5 & -500.5 \end{pmatrix} \vec{y}, \quad \vec{y}(0) = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \tag{2.2.4}$$

The solution of this system is given by

$$\vec{y}\,(t) = \frac{3}{2}e^{-t}\begin{pmatrix}1\\1\end{pmatrix} + \frac{1}{2}e^{-1000t}\begin{pmatrix}1\\-1\end{pmatrix}$$

The eigenvalues are $\lambda_1 = -1$ and $\lambda_2 = -1000$ and the stiffness ratio is $S = 1000$. The solution mode e^{-1000t} decays very rapidly when time evolves. That is to say the corresponding time scale is very short (recall the dimension of eigenvalue is s^{-1}). This mode is only relevant during the **spin-up time**. So, this mode is typically not an interesting one as it will not influence the physical behaviour of the system when time proceeds. This solution mode is called the **fast mode**. On the other hand, the mode e^{-t} is a **slow** one and is in our interest. This mode has a larger time scale.

Typically we would like to choose a time step based on accuracy considerations. For a given numerical scheme, we would like to choose Δt so that the local error in each time step is sufficiently small that the accumulated or global error will satisfy our error tolerance. Based on this the chosen time step should correspond to the time scale of the slowest modes as they determined the decay of the associated errors after some time during the evolution. However, if this scheme is not A-stable, then stability considerations come into play and will force us to use a *much* smaller time step since it is determined by the largest eigenvalues. Hence, the fastest modes, which are usually unimportant for the overall solution, is the limiting factor because of stability. This happens, for example, if we try to use an explicit scheme on a stiff problem.

We shall return to our stiff example above and we shall use the explicit Euler scheme. Based on accuracy considerations a time step of 0.05 s or 0.1 s would be desirable. However, the stability limit is determined by the largest eigenvalue λ_2 and is thus given by $\Delta t \leq 0.002$ s. This leads to very inefficient time integration and thus not feasible.

Unconditionally stable or A-stable schemes would in principle be appropriate for stiff problems. In this case we can adjust the chosen time step to the desired accuracy of the slow modes. If we consider the above example and apply the trapezoidal rule with a chosen time step of 0.05 s, then the obtained solution appeared not to be accurate as it is oscillated as shown in Figure 2.16. (In the figure, only the solution $y_1(t)$ is shown.) The underlying explanation for this strange behaviour is by considering the amplification factor of the trapezoidal rule and taking the limit $\Delta t \to \infty$,

$$\lim_{Re(\lambda)\Delta t \to -\infty} \frac{1 + \frac{1}{2}\,Re(\lambda)\Delta t}{1 - \frac{1}{2}\,Re(\lambda)\Delta t} = -1$$

(see also Figure 2.4). Although the rapidly decaying modes are bounded, they are damped only very mildly when a relatively large time step is chosen. In the considered example, the fast mode causes oscillating, slowly decaying solution as can be seen in Figure 2.16. So, even for the trapezoidal rule a much smaller time step has to be taken in order to get accurate results.

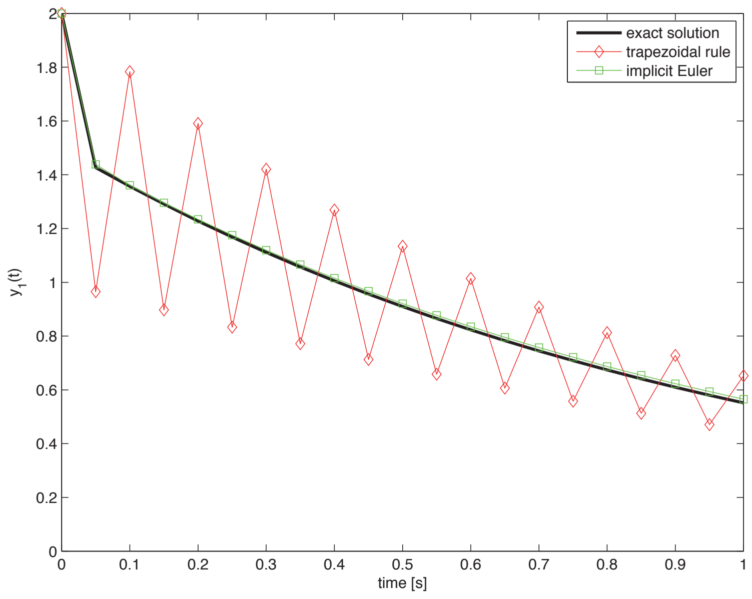


Figure 2.16: Solution of stiff problem, Eq. (2.2.4), obtained with $\Delta t \ = \ 0.05$ s.

When considering the amplification factor of implicit Euler in the limiting case $\Delta t \to \infty$,

$$\lim_{Re(\lambda)\Delta t \to -\infty} \frac{1}{1 - Re(\lambda)\Delta t} = 0$$

we conclude that it does damp all the solution modes including the rapidly decaying ones. So, implicit Euler seems to have a perfect stability property. This is confirmed by Figure 2.16 showing the result with $\Delta t = 0.05$ s. This is called **L-stability** or **stiff stability**. An L-stable scheme is always A-stable. Implicit Euler is thus an example of an L-stable scheme while the trapezoidal rule is not L-stable. Despite the fact that implicit Euler is only first order accurate while the trapezoidal rule is second order accurate, it is very valuable for stiff problems. So, from a practical point of view, only stiffly stable schemes are useful for stiff systems of ODEs; see Gear (1971). The second order BDF scheme, Eq. (2.1.25), is also L-stable. See also Lambert (1991) for more details.

Exercise 2.2.15 Verify that the BDF scheme is indeed L-stable.

For completeness, a few words about the solution obtained with implicit methods. Though L-stable schemes are very useful for stiff problems, a main disadvantage is to find a **solution of a system of equations** in each time step, which can be very costly. For instance, implicit Euler applied to $d\vec{y}/dt = A\vec{y}$ gives

$$(I - \Delta t A)\, \vec{y}^{n+1} = \vec{y}^n$$

with I the $m \times m$ identity matrix. This is a system of equations for which a solution, i.e. $\vec{y}^{n+1}$, needs to be sought. This amounts to the computation of the inverse of matrix

$(I - \Delta t A)$. This is typical of implicit methods. Also, a system of equations needs to be solved when the trapezoidal rule is employed,

$$\left(I - \frac{1}{2}\Delta t A\right) \vec{y}^{n+1} = \left(I + \frac{1}{2}\Delta t A\right) \vec{y}^{n}$$

Well-known techniques for finding the solution of a system of equations are the **direct** methods, like the **Gaussian elimination**, and the **iterative** methods, like the **Jacobi and Gauss-Seidel methods** or more advanced approaches such as **multigrid** techniques and **Krylov subspace methods** with **preconditioning**. This topic is, however, beyond the scope of this lecture notes. A good textbook for the introduction to this topic is Golub and Van Loan (1989).

2.3 Concluding remarks

We conclude this chapter with some remarks about the numerical solution of first ordinary differential equations. Only a basic introduction to this subject has been given. Some well-known, popular schemes, both one-step and two-step schemes, have been treated. We have employed these schemes as the simplest possible device to introduce some important notions, like consistency and stability, that have a general validity. For instance, these notions will be considered in the following chapters about the numerical solution of partial differential equations. However, they are just a small subset of many time integration methods. We did not consider other families of linear multi-step methods like (explicit) **Adams-Bashforth** methods, (implicit) **Adams-Moulton** methods and also **predictor corrector** techniques of which the **method of Heun** is the most prominent one. Another well-known class of time integrators are the **Runge-Kutta** methods. These methods take some intermediate steps (for example, a half-step) to obtain higher order accuracy, but discard all previous information before considering the next time step.

Detailed background information on all these matters can be found at wikipedia.org and scholarpedia.org. You may also consult classical textbooks like Gear (1971), Henrici (1962), Hirsch (1990), Lambert (1991) and Richtmyer and Morton (1967).

Chapter 3

Numerical methods for boundary value problems

3.1 Introduction to PDEs

In Chapter 2 ordinary differential equations have been treated. Their solutions are simply functions of a single independent variable representing time. Frequently, physical systems often evolve not only in time but also in at least one spatial dimension. This chapter is devoted to partial differential equations. A partial differential equation (PDE) permits one to describe solutions that depend on more than one independent variable. Often these independent variables are time and one or more spatial dimensions or in **steady state** problems, two or more spatial dimensions only.

This section is not intended to be a complete treatment of partial differential equations. Instead, its aim is to serve as an introduction to a minimal amount of terminology from the field of PDEs. A good introductory textbook on PDEs is Andrews (1986).

3.1.1 Examples of PDEs

Partial differential equations arise in many areas of physics and engineering. On physical grounds, the form of these equations will involve the time rate-of-change of the solution and/or the spatial rate-of-change (or gradient) of the solution. Some examples are:

Diffusion equation:

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}$$

where t is time, x is the coordinate in space, $T(x,t)$ is the temperature, and κ is a thermal diffusivity coefficient.

Convection equation:

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

where $c(x,t)$ is the concentration, and u is a constant propagation velocity.

Wave equation:

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

Laplace equation:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0$$

where $\phi(x,y)$ is a potential or harmonic function.

Poisson equation:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = f$$

with $f(x,y,t)$ a source term.

Burger's equation:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}$$

with $u(x,t)$ the flow velocity and ν the (molecular) viscosity.

Shallow water equations:

$$\frac{\partial \zeta}{\partial t} + \frac{\partial hu}{\partial x} = 0\,, \quad \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial \zeta}{\partial x} + c_f\frac{u|u|}{h} = 0$$

with $\zeta(x,t)$ the water level above the still water level, $h(x,t) = \zeta(x,t) + d(x)$ the water depth, $d(x)$ the bottom level measured from the still water level positively downwards, $u(x,t)$ the flow velocity, c_f the dimensionless bottom friction coefficient, and g the acceleration of gravity.

In addition to the partial differential equations, some boundary conditions and initial data must be provided to uniquely specify a solution. Solving the equations means finding the **dependent** variable or variables, e.g. T, u, ζ, ϕ, c, as function of the **independent** variables, e.g. t, x, y, such that they fulfill the PDEs and the boundary conditions are satisfied. This is called the **boundary value problem**.

3.1.2 Relevant notions on PDEs

Let u be the dependent variable. A shorthand notation to represent a general PDE is

$$\mathcal{L}(u) = g$$

and $\mathcal{L}$ is called the **differential operator**. The right-hand side g is not a function of u. If $g \equiv 0$ then the PDE is called **homogeneous**, otherwise it is **nonhomogeneous**. Example: the Laplace equation is homogeneous and the Poisson equation is nonhomogeneous.

The **order** of a PDE is the order of highest derivative that appears in the equation. Example: the order of the convection equation is one, whereas that of the wave equation is two. The **dimension** of a PDE is the dimension of the spatial domain. Example: the diffusion equation is one dimensional (1D), while the Poisson equation is two dimensional (2D).

A differential operator $\mathcal{L}$ is called **linear** if and only if the following hold

$$\begin{aligned}\mathcal{L}(u+v) &= \mathcal{L}(u) + \mathcal{L}(v)\\ \mathcal{L}(\alpha u) &= \alpha\mathcal{L}(u)\end{aligned}$$

with α a constant. An operator that is not linear is called **nonlinear**. Example of a nonlinear PDE is the Burger's equation. Do not confuse non-constant or variable-coefficient PDEs with nonlinear PDEs. For example

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \sin^2(x)\, u$$

is linear, but with non-constant coefficient, while

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u\frac{\partial u}{\partial x}$$

is nonlinear.

The distinguishing of PDEs into linear and nonlinear is an important issue. Solutions of linear equations **superimpose**. Assume u and v are both solutions of a given linear PDE. Then all linear combinations of these two solutions, $\alpha u + \beta v$, are also solutions of the PDE, where α and β are constant coefficients. This is a very important advantage for solving linear PDEs. If we are able to find a set of particular solutions of the PDE, we can construct all other solutions as linear combinations of these. Nonlinear PDEs do not share this property of superposition and are usually much harder to solve and the solutions more difficult to analyze. It is common in mathematical modelling to attempt to approximate a nonlinear phenomenon in nature with a linear model. (An example can be found in Appendix A.3.) While this linear model provides insight into the nature of the phenomenon, often it is insufficient to describe some of the important aspects and one must introduce nonlinear terms to the model. Usually nonlinear PDEs cannot be solved by hand, so numerical methods must be devised. In this reader, however, we shall largely ignore nonlinear equations.

Exercise 3.1.1 Consider the above 1D shallow water equations. Verify that these equations are first order, nonlinear and nonhomogeneous.

3.1.3 Well posed problems

In the previous sections we saw some examples of PDEs. We now consider some important issues regarding the **solvability** of PDE problems. In general, a PDE alone, without any boundary or initial conditions, will either have an infinity of solutions, or have no solution. Thus, in formulating a PDE problem there are at least three ingredients:

1. The equation itself.
2. The enclosed spatial domain Ω on which the PDE is required to be satisfied.
3. The boundary conditions that the solution must to be met at the boundaries of Ω.
4. In case of a time-dependent problem, initial condition(s) must be included as well.

For a PDE based mathematical model of a physical system to give *useful* results, it is generally necessary to formulate that model as what mathematicians call a **well posed** PDE problem. A PDE problem is said to be well posed if

- a solution to the problem **exists**,
- the solution is **unique**, and
- the solution depends **continuously** on the problem data[1]. This is closely related to the notion of stability of the PDE.

If one of these conditions is not satisfied, the PDE problem is said to be **ill posed**. In practice, the question of whether a PDE problem is well posed can be difficult to settle. Roughly speaking the following guidelines apply.

- The boundary conditions imposed must not be too many or a solution will not exist.
- The boundary conditions imposed must not be too few or the solution will not be unique.
- The kind of boundary conditions must be correctly matched to the type of the PDE or the solution will not be stable.

In this reader, only well posed problems will be considered.

[1]The problem data consists of the coefficients in the PDE, the boundary and initial conditions and the domain on which the PDE is required to hold.

3.1.4 Types of boundary conditions

Since the domain Ω is *finite*, boundary conditions are thus required and represent the influence of the outside world. Different types of boundary conditions can be applied on the boundary of Ω. The most common occurring in practice are **Dirichlet**, **Neumann** and **Robin** conditions. Physically speaking, a Dirichlet condition usually corresponds to setting the value, a Neumann condition usually specifies a flux condition on the boundary, and a Robin condition typically represents a radiation condition. Mathematically speaking, they are given as follows. Dirichlet condition provides a function of time, $g(t)$, as the constraint on the solution at a specific boundary point $\vec{x}_b$,

$$u(\vec{x}_b, t) = g(t)$$

In the case of Neumann condition the flux of u normal to the boundary is specified by a given function $g(t)$ at a boundary point $\vec{x}_b$,

$$\frac{\partial u}{\partial \vec{n}}(\vec{x}_b, t) = g(t)$$

with $\vec{n}$ the normal to the boundary. If $g(t) = 0$ then we say that the boundary is **insulated**: u cannot flow across the boundary. The Robin condition is a combination of Dirichlet and Neumann conditions,

$$\alpha u(\vec{x}_b, t) + \beta \frac{\partial u}{\partial \vec{n}}(\vec{x}_b, t) = g(t)$$

where α and β may in general depend on position along the boundary.

As with PDEs, boundary conditions can be classified into linear or nonlinear and homogeneous or nonhomogeneous. The above are linear. They are homogeneous if $g(t) \equiv 0$ and nonhomogeneous otherwise.

3.1.5 Classification of PDEs

We classify ODEs in terms of their order and whether they are linear or nonlinear. PDEs are more difficult to classify, because of the greater variety of basic forms the PDE may take. Not only are the order and linearity important, but also the PDE formulation. It is the PDE formulation that dictates what types of boundary conditions can be imposed and ultimately what types of physical processes the PDE describes.

A classification is possible for second order, linear PDEs in two independent variables. Any such 2D equation can be written as

$$a\frac{\partial^2 u}{\partial x^2} + 2b\frac{\partial^2 u}{\partial x \partial y} + c\frac{\partial^2 u}{\partial y^2} + d\frac{\partial u}{\partial x} + e\frac{\partial u}{\partial y} + fu + g = 0 \tag{3.1.1}$$

where the coefficients a, b, c etc. may depend on (x, y). One of the independent variables may represent time t, which we then have a 1D equation. The classification of Eq. (3.1.1) is based upon the first three terms. It can be classified into three types according to the sign of the discriminant $b^2 - 4ac$,

elliptic : $b^2 - 4ac < 0$

parabolic : $b^2 - 4ac = 0$

hyperbolic : $b^2 - 4ac > 0$

Generally, different numerical methods are required for the different classes of PDEs. The need for this specialization in numerical approach is rooted in the physics from which the different classes of PDEs arise. The physical problem has certain properties that we would like to preserve with our numerical approximation, and it is important to understand the underlying problem and be aware of its mathematical properties before blindly applying a numerical method.

Elliptic equations generally arise from a physical problem that involves a diffusion process that has reached equilibrium, a steady state temperature distribution, for example. Hyperbolic equations are able to support solutions with discontinuities, for example a shock wave. Hyperbolic PDEs usually arise in wave propagation and convection driven transport problems. Mathematically, parabolic PDEs serve as a transition from the hyperbolic PDEs to the elliptic PDEs. Physically, parabolic PDEs tend to arise in time dependent diffusion problems, such as the transient flow of heat in accordance with Fick's law of heat conduction.

It turns out that all elliptic equations are steady state, all parabolic equations are diffusion-like, and all hyperbolic equations are wave-like. Therefore, we outline the following **prototypes**.

Elliptic

Prototype is the Laplace equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

Elliptic equations do not depend upon time, but rather only spatial variables. The boundary conditions are usually of the type Dirichlet or Neumann.

Parabolic

Prototype is the diffusion equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$$

Here, the independent variable y of the general PDE, Eq. (3.1.1), is time. An initial condition is thus necessary. The problem is only well posed in the forward time direction (otherwise the solution becomes unstable in the backward time direction). Ω is bounded at two sides in the $x-$direction. Any boundary conditions, Dirichlet, Neumann and Robin, though in certain combinations, are appropriate at both sides of Ω. (See also Exercise 3.2.9.)

Hyperbolic

Prototype is the wave equation

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

or the convection equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0$$

Here, y variable is again time. An initial condition is necessary. The problem is well posed in both direction of time. Boundary conditions of the Dirichlet type are imposed at boundaries with ingoing characteristics. Note that the convection equation is a first order PDE and hence cannot be classified in the above sense.

Exercise 3.1.2 Show that the wave equation and the convection equation are mathematically equivalent.

Important differences between the three prototypes are:

- The Laplace equation is not time dependent. The convection equation is valid in both directions in time. The diffusion equation is well posed only in forward time direction.
- Information is propagated at *finite speed* without changing its shape in the convection equation. For the diffusion equation information is transmitted at *infinite speed* but it is also *dissipated* and thus the shape of the solution becomes more smooth as time proceeds.
- The Laplace equation is fundamentally different from the diffusion and convection equation in that the solution at each point depends on all other points. Solutions, e.g. potential fields, cannot be found by marching in from the boundary. Instead, solutions must be found at all points *simultaneously*.

Since both diffusion and convection equations are typically time dependent, the associated numerical approach is therefore time marching: the numerical solution is determined at any time through a sequence of time steps. On the other hand, the solution method for the Laplace equation is typically **iterative** of nature: the solution in each point in the domain is **updated** a number of times by taking into account the surrounding points until an acceptable error tolerance is reached.

It is not possible to give a meaningful classification for PDEs in more than two independent variables. However, there are natural extensions for the three prototype equations such as

3D Laplace equation:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0$$

2D diffusion equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

2D convection equation:

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = 0$$

Diffusion and convection equations exist also in three space dimensions.

Later in this lecture notes, we shall deal with free surface flows, such as occurring in seas, estuaries, lakes, rivers and canals. The underlying equations to describe such flows are the shallow water equations. These equations are a set of *hyperbolic* partial differential equations that describe the flow below a free surface. Not only flow but also *convective* transport of dissolve substances and their *diffusive* processes in open water bodies will be discussed. For these reasons, we shall focus on the numerical solution of the diffusion equation (Section 3.2), the convection equation (Section 3.3) and the convection-diffusion equation (Section 3.4). Though elliptic equations are very important in the field of computational fluid dynamics (CFD), they will not dealt with here. Detailed discussion on the numerical solution of elliptic equations may be found in Abbott and Basco (1989), Ferziger and Perić (1996), Fletcher (1988) and Wesseling (2001).

3.2 Diffusion equation

In this section we discuss the numerical solution of the 1D diffusion equation. The specific (initial) boundary value problem we wish to solve is the following

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}, \quad x \in (0, L), \quad t > 0 \tag{3.2.1a}$$

$$T(x, 0) = T^0(x), \quad x \in [0, L] \tag{3.2.1b}$$

$$T(0, t) = 1, \quad t > 0 \tag{3.2.1c}$$

$$\frac{\partial T}{\partial x}(L, t) = 0, \quad t > 0 \tag{3.2.1d}$$

and is known as the **heat equation**. This equation describes the flow of heat in a rod with a finite length, made out of some heat-conducting material, subject to some boundary conditions at each end. Specifically, this heat *spreads* out spatially as time increases. This phenomenon is called **diffusion**. The solution to the heat equation is the temperature distribution at any time t and vary with x along the rod. We assume that $T(x, t)$ is smooth enough, so that we can differentiate this function many times as we want and each derivative is a well-defined *bounded* function. Here L is the given finite length of the domain and $T^0(x)$ is a given function defining the initial condition. The coefficient $\kappa > 0$ is the thermal diffusivity. In this specific case, we have imposed a Dirichlet condition at $x = 0$ and a Neumann condition at $x = L$. Since the boundary conditions are all time independent, we expect the solution to eventually reach a **steady state** solution $T(x, t) \to \tilde{T}(x)$ which then remains essentially unchanged at later times. Hence, we shall marching forward in time until a steady state solution is found. During this marching a **transient** solution will be obtained that must fulfill the initial condition.

Exercise 3.2.1 Show that the steady state solution is $\lim\limits_{t\to\infty} T(x, t) = \tilde{T}(x) = 1, x \in [0, 1]$.

The first step is to **discretize** the domain into a finite number of **grid points** $M+1$ where we intend to compute the solution of the PDE. We shall use a uniform or **equidistant** grid, with a **grid spacing** $\Delta x = L/M$. We shall refer to one of the points in the grid as $x_m = m\Delta x\,, m = 0, \ldots, M$. The first and last grid points of the domain, x_0 and x_M, are called the **boundary points**, while the other grid points are the **internal** ones. Likewise we discretize the time interval into a number of time steps, separated by a time increment Δt, and only compute the solution for times $t_n = n\Delta t\,, n = 1, 2, 3, \ldots$ until a steady state is found. Our aim is to compute the solution of the PDE for all values $T_m^n \approx T(m\Delta x, n\Delta t)$.

We now have a **spatial grid** or **computational domain** that approximates the physical domain $x \in [0, L]$ and a discrete time frame. The next step it to define approximations to the partial derivatives appearing in the PDE, Eq. (3.2.1a), as the finite precision of real values in a computer precludes exact computation of these derivatives. Generally, a distinction is made between approximating the time derivative and approximating the spatial derivative. The whole approximation process is done in two steps:

1. Approximate the spatial derivative first. This step is called the **semi discretization** step and leads to a system of first order ODEs.

2. The resulting system of ODEs is approximated by means of time integration. This step has been dealt with in Chapter 2.

This two-step process is called the **method of lines** (MOL)[2].

A number of methods have been proposed to approximate spatial derivatives. Popular methods are the **finite difference**, **finite volume** and **finite element** methods. In this lecture notes, we restrict ourselves to one of the most natural approximations, finite differences.

[2]The origin of the name *method of lines* is best illustrated at
http://en.wikipedia.org/wiki/Method_of_lines#mediaviewer/File:Method_of_lines.gif.

3.2.1 Finite difference method

The basic methodology of the finite difference method (FDM) is to approximate the spatial derivatives with **differences** of the unknowns on the grid. A variety of different approximations are possible and the choice depends on the desired accuracy.

The definition of the partial derivative is

$$\frac{\partial T}{\partial x}(x,t) = \lim_{\Delta x \to 0} \frac{T(x+\Delta x, t) - T(x,t)}{\Delta x}$$

We do not have the luxury of computing the limit numerically so we must select a small value for the **mesh width** Δx and approximate the derivative as

$$\frac{\partial T}{\partial x}(m\Delta x, t) \approx \frac{T_{m+1}(t) - T_m(t)}{\Delta x} \tag{3.2.2}$$

with $T_m(t) \approx T(m\Delta x, t)$. Note that the quantity $T_m(t)$ is *continuous* in time as there is no approximation in time yet! This approximation is known as the **forward finite difference** formula. This approximation is also called the **one-sided** approximation since T is evaluated only at $x_{m+1} > x_m$.

We now quantify the accuracy of the forward difference approximation. By means of the Taylor series expansion we can derive the associated truncation error of this approximation. We expand $T(x + \Delta x, t)$, as follows

$$T(x+\Delta x, t) = T(x,t) + \Delta x T_x(x,t) + \frac{1}{2}\Delta x^2 T_{xx}(x,t) + \frac{1}{6}\Delta x^3 T_{xxx}(x,t) + \frac{1}{24}\Delta x^4 T_{xxxx}(x,t) + \ldots$$

with $T_x = \partial T/\partial x$, $T_{xx} = \partial^2 T/\partial x^2$ etc. Here we assume that $T(x,t)$ is sufficiently smooth. Substitution gives

$$\begin{aligned}
\frac{T(x+\Delta x, t) - T(x,t)}{\Delta x} &= \frac{1}{\Delta x}\left(\Delta x \frac{\partial T}{\partial x}(x,t) + \frac{1}{2}\Delta x^2 \frac{\partial^2 T}{\partial x^2}(x,t) + \ldots\right) \\
&= \frac{\partial T}{\partial x}(x,t) + \frac{1}{2}\Delta x \frac{\partial^2 T}{\partial x^2}(x,t) + \ldots \\
&= \frac{\partial T}{\partial x}(x,t) + \mathcal{O}(\Delta x)
\end{aligned}$$

Hence, the forward difference formula is accurate only to order Δx.

There are, however, two other possibilities. The first is the **backward finite difference** approximation

$$\frac{\partial T}{\partial x}(m\Delta x, t) \approx \frac{T_m(t) - T_{m-1}(t)}{\Delta x} \tag{3.2.3}$$

and the second is the **centred finite difference** formula or **central differences**

$$\frac{\partial T}{\partial x}(m\Delta x, t) \approx \frac{T_{m+1}(t) - T_{m-1}(t)}{2\Delta x} \tag{3.2.4}$$

since this approximation is centred around the point of consideration x_m. Central differences is an example of a **two-sided** approximation and we shall see that, for a sufficiently small value of Δx, this approximation leads to a more accurate numerical solution of the diffusion equation than a one-sided approximation. This does not necessarily imply that one-sided approximations are not appropriate. For instance, for *convective* transport, it may be appropriate to use either the forward or backward approximation. This will be discussed in Section 3.3.4.

Exercise 3.2.2 Show that the backward and centred finite difference approximations are accurate to Δx and Δx^2, respectively.

The above central differences is measured using *double* **grid size**, i.e. $2\Delta x$. Alternatively, the partial derivative may also be approximated as follows

$$\frac{\partial T}{\partial x}(m\Delta x, t) \approx \frac{T_{m+1/2}(t) - T_{m-1/2}(t)}{\Delta x} \tag{3.2.5}$$

which is central differences using *single* grid size. However, the quantities $T_{m\pm 1/2}$ are not defined on the grid. They must be computed by means of **linear interpolation**, as follows

$$T_{m-1/2} = \frac{1}{2}\left(T_{m-1} + T_m\right), \quad T_{m+1/2} = \frac{1}{2}\left(T_m + T_{m+1}\right)$$

Exercise 3.2.3 Show that Eq. (3.2.5) is equivalent to Eq. (3.2.4).

The approximation of the second derivative can be obtained by recalling that

$$\frac{\partial^2 T}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial T}{\partial x}\right)$$

Hence,

$$\frac{\partial^2 T}{\partial x^2}(m\Delta x, t) \approx \frac{\frac{\partial T}{\partial x}\big|_{m+1/2} - \frac{\partial T}{\partial x}\big|_{m-1/2}}{\Delta x} \approx \frac{\frac{T_{m+1}-T_m}{\Delta x} - \frac{T_m - T_{m-1}}{\Delta x}}{\Delta x} = \frac{T_{m+1} - 2T_m + T_{m-1}}{\Delta x^2}$$

Substituting this expression into our original PDE, Eq. (3.2.1a), we obtain

$$\frac{dT_m}{dt} = \kappa\,\frac{T_{m+1} - 2T_m + T_{m-1}}{\Delta x^2}, \quad m = 1, \ldots, M-1 \tag{3.2.6}$$

which is a semi discretization of Eq. (3.2.1a) of the **interior** of the domain. Note that Eq. (3.2.6) is an *ordinary* differential equation.

The consistency of the *semi-discretized* equation, Eq. (3.2.6), with the heat equation, Eq. (3.2.1a), can be verified by replacing T_m by the exact solution $T(x_m, t)$ in the finite difference formula, defining the spatial **truncation error**

$$\tau_{\Delta x} = \frac{\partial T}{\partial t}(x_m, t) - \kappa \frac{T(x_{m+1}, t) - 2T(x_m, t) + T(x_{m-1}, t)}{\Delta x^2}$$

and subsequently check the limiting case

$$\lim_{\Delta x \to 0} \tau_{\Delta x} = 0$$

The exact solution $T(x, t)$ is assumed to be smooth enough, so $T(x \pm \Delta x, t)$ can be expand in Taylor series, as follows

$$T(x \pm \Delta x, t) = T(x, t) \pm \Delta x T_x + \frac{1}{2}\Delta x^2 T_{xx} \pm \frac{1}{6}\Delta x^3 T_{xxx} + \frac{1}{24}\Delta x^4 T_{xxxx} + \ldots$$

Substitution in

$$\tau_{\Delta x} = \frac{\partial T}{\partial t}(x, t) - \kappa \frac{T(x + \Delta x, t) - 2T(x, t) + T(x - \Delta x, t)}{\Delta x^2}$$

and after some calculus, we have

$$\tau_{\Delta x} = \frac{\partial T}{\partial t} - \kappa \left(\frac{\partial^2 T}{\partial x^2} + \frac{\Delta x^2}{12} \frac{\partial^4 T}{\partial x^4} + \ldots \right)$$

Using our original PDE, Eq. (3.2.1a), this becomes

$$\tau_{\Delta x} = -\frac{\kappa \Delta x^2}{12} \frac{\partial^4 T}{\partial x^4} + \mathcal{O}\left(\Delta x^4\right) = \mathcal{O}\left(\Delta x^2\right)$$

and we conclude that the leading term contains Δx^2 which means that the **order of accuracy** is two. Moreover, the considered **space discretization** of the diffusion operator is **consistent** which means that the approximation converges to the original diffusion term as $\Delta x \to 0$. It must be emphasised that this does not show how well the numerical solution approximates the solution of the PDE.

Exercise 3.2.4 Verify the above truncation error.

Exercise 3.2.5 Let us consider the following semi discretization of Eq. (3.2.1a),

$$\frac{1}{12}\frac{dT_{m+1}}{dt} + \frac{5}{6}\frac{dT_m}{dt} + \frac{1}{12}\frac{dT_{m-1}}{dt} = \kappa \frac{T_{m+1} - 2T_m + T_{m-1}}{\Delta x^2}$$

Show that $\tau_{\Delta x} = \mathcal{O}(\Delta x^4)$.

Finally, we also need to discretize the boundary conditions. The Dirichlet condition, Eq. (3.2.1c), is simply given by

$$T_0(t) = 1\,, \quad t > 0$$

The implementation of the Neumann condition, Eq. (3.2.1d), is less trivial. When using the central differences for spatial derivatives, we often need to consider values of the numerical solution that *lie* outside the computational domain in order to compute spatial derivatives on the boundaries. The usual approach is to introduce **virtual points** that lie outside the domain and to use the Neumann condition and the numerical scheme for the interior domain, to eliminate the values at virtual points. For example, the centred approximation for the first derivative at boundary $x = L$, Eq. (3.2.1d), is (substitute $m = M$ in Eq. (3.2.4))

$$\frac{\partial T}{\partial x}(L,t) \approx \frac{T_{M+1}(t) - T_{M-1}(t)}{2\Delta x} = 0 \quad \Rightarrow \quad T_{M+1}(t) = T_{M-1}(t)$$

involving the virtual point $x_{M+1} = L + \Delta x$. We also apply the semi discretization for the PDE at the boundary point x_M (see Eq. (3.2.6)),

$$\frac{dT_M}{dt} = \kappa\,\frac{T_{M+1} - 2T_M + T_{M-1}}{\Delta x^2}$$

and eliminate T_{M+1} from this equation. We are left with the numerical treatment for the Neumann condition, involving only values of the numerical solution located within the domain

$$\frac{dT_M}{dt} = 2\kappa\,\frac{T_{M-1} - T_M}{\Delta x^2}$$

We shall leave it as an

Exercise 3.2.6 to show that this approximation is first order accurate.

Our first thought might be to conclude that our *overall* approximation in the whole domain, including the boundary points, might have lost second order accuracy. However, this is not the case. Generally, we achieve second order accuracy in the overall numerical method even if the truncation error is $\mathcal{O}(\Delta x)$ at a *single* boundary point as long as it is $\mathcal{O}(\Delta x^2)$ everywhere else. So, we can often get away with one order of accuracy lower in the local error for the boundary conditions than what we have elsewhere.

Summarising, we now have the following linear system of first order ODEs with the unknowns $T_m(t)\,, m = 0, \dots, M$,

$$T_0 = 1\,, \quad t > 0 \tag{3.2.7a}$$

$$\frac{dT_m}{dt} = \kappa\,\frac{T_{m+1} - 2T_m + T_{m-1}}{\Delta x^2}\,, \quad m = 1, \dots, M-1\,, \quad t > 0 \tag{3.2.7b}$$

$$\frac{dT_M}{dt} = 2\kappa\,\frac{T_{M-1} - T_M}{\Delta x^2}\,, \quad t > 0 \tag{3.2.7c}$$

$$T_m(0) = T^0(x_m)\,, \quad m = 0, \dots, M \tag{3.2.7d}$$

Exercise 3.2.7 Verify this system of first order ODEs.

It is standard practice to write this linear system in the matrix-vector notation

$$\frac{d\vec{T}}{dt} = A\vec{T} + \vec{g}\,, \quad \vec{T}(0) = \vec{T}^0$$

with

$$\vec{T} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_{M-2} \\ T_{M-1} \\ T_M \end{pmatrix}$$

a vector containing M unknowns,

$$A = \frac{\kappa}{\Delta x^2}\begin{pmatrix} -2 & 1 & 0 & & & \cdots & & & 0 \\ 1 & -2 & 1 & & & & & & \\ 0 & 1 & -2 & 1 & & & & & \\ & & 1 & -2 & 1 & & & & \\ \vdots & & & \ddots & \ddots & \ddots & & & \vdots \\ & & & & 1 & -2 & 1 & & \\ & & & & & 1 & -2 & 1 & 0 \\ & & & & & & 1 & -2 & 1 \\ 0 & & & \cdots & & & 0 & 2 & -2 \end{pmatrix} \tag{3.2.8}$$

an $M \times M$ discretization matrix, and

$$\vec{g} = \frac{\kappa}{\Delta x^2}\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

a vector with M elements.

Exercise 3.2.8 Try to be familiarize with the derivation of the above matrix, Eq. (3.2.8).

Exercise 3.2.9 Let consider the heat conduction problem where both ends of the rod are insulated, i.e. there is no heat flux through the ends. Derive the system of ODEs and show that this problem is ill posed.
(*Hint*: show that the resulting discretization matrix is *singular*.)

The next step is time integration of the linear system. Before we continue, we slightly rewrite Eq. (3.2.7c) as follows

$$\frac{d\tilde{T}_M}{dt} = \kappa \frac{T_{M-1} - 2\tilde{T}_M}{\Delta x^2}$$

with

$$\tilde{T}_M = \frac{1}{2} T_M$$

If we redefine the vector with unknowns as

$$\vec{T} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_{M-2} \\ T_{M-1} \\ \tilde{T}_M \end{pmatrix}$$

then the matrix of the system becomes

$$A = \frac{\kappa}{\Delta x^2} \begin{pmatrix} -2 & 1 & 0 & & \cdots & & 0 \\ 1 & -2 & 1 & & & & \\ 0 & 1 & -2 & 1 & & & \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ & & & 1 & -2 & 1 & 0 \\ & & & & 1 & -2 & 1 \\ 0 & & & \cdots & & 1 & -2 \end{pmatrix} \tag{3.2.9}$$

This discretization matrix, Eq. (3.2.9), has the advantage of being **symmetric**, meaning that all the eigenvalues are real. This is exactly what we are looking for since harmonic motion does not show up in the heat equation.

Using the Gerschgorin's theorem we can estimate the eigenvalues of A. For a symmetric matrix A with entries a_{ij} its eigenvalues are lying in the union of intervals

$$a_{ii} - \sum_{\substack{j=1 \\ j\neq i}}^{M} |a_{ij}| \leq \lambda \leq a_{ii} + \sum_{\substack{j=1 \\ j\neq i}}^{M} |a_{ij}|$$

Thus, in our case we have

$$a_{ii} = \frac{-2\kappa}{\Delta x^2}\,, \quad \forall i \in [1, M]$$

$$\sum_{j=2}^{M} |a_{1j}| = \frac{\kappa}{\Delta x^2}\,, \quad \sum_{j=1}^{M-1} |a_{Mj}| = \frac{\kappa}{\Delta x^2}$$

and

$$\sum_{\substack{j=1 \\ j\neq i}}^{M} |a_{ij}| = \frac{2\kappa}{\Delta x^2}\,, \quad \forall i \in [2, M-1]$$

and we conclude that all the eigenvalues satisfy the inequality

$$\frac{-4\kappa}{\Delta x^2} \leq \lambda \leq 0$$

Moreover, matrix A is nonsingular, meaning that all the eigenvalues are nonzero and thus exclusively negative. As a consequence, the considered linear system of ODEs is regarded as being *damped*, a property also shared by the diffusion-like term.

While we are at it, let us also consider the *distribution* of the eigenvalues. For a sufficiently small value of Δx, the distance between the smallest and the largest eigenvalue is relatively large implying that the system can possibly be *stiff*. Although this may ask for a stiffly stable scheme, this is often not the case. The changes in the solution appears not to be grid dependent.

3.2.2 Time integration

The semi discretization of the heat equation leads to a linear system of first order ODEs. This system has been identified as being damped. Thus, a good choice for time integration would be the explicit Euler scheme.

Exercise 3.2.10 Explain why the midpoint rule is not a useful method for the heat equation.

Application of explicit Euler to Eq. (3.2.7b) yields

$$\frac{T_m^{n+1} - T_m^n}{\Delta t} = \kappa \frac{T_{m+1}^n - 2T_m^n + T_{m-1}^n}{\Delta x^2}\,, \quad m = 1, \ldots, M-1\,, \quad n = 0, 1, 2, \ldots$$

with Δt the time step and $T_m^n \approx T(m\Delta x, n\Delta t)$ a discrete function, which represents the wanted numerical solution to the heat equation, Eq. (3.2.1a), at time step n and grid point m. We can re-arrange this equation as follows,

$$T_m^{n+1} = T_m^n + \frac{\kappa \Delta t}{\Delta x^2}\left(T_{m+1}^n - 2T_m^n + T_{m-1}^n\right) \tag{3.2.10}$$

At each *new* time step, the dependent variable T at each interior grid point is computed from values of T at three grid points at the *preceding* time step. This expression defines an **explicit** scheme for solving the heat equation. Hence, the solution can be found at any time through a sequence of time steps. This scheme is known as the **FTCS** scheme, which stands for Forward in Time, Central in Space, since we have used the forward approximation in time with explicit Euler and the central differences in space.

It is common practice to represent finite difference formulas *graphically* by a **stencil** or **molecule**. The stencil indicates which group of unknowns is involved in a numerical scheme, and how they are linked to each other. The stencil of the FTCS scheme is depicted in Figure 3.1.

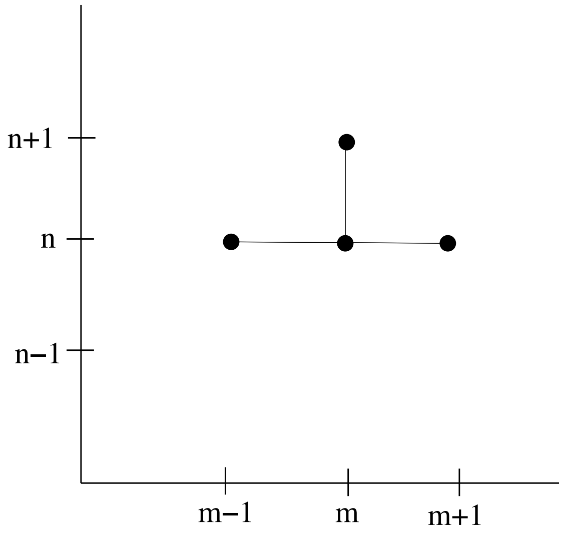


Figure 3.1: The stencil of the FTCS scheme.

We also need to discretize the initial and boundary conditions of the heat equation. In this case, the considered boundary conditions can be approximated by:

$$T_0^{n+1} = 1\,, \quad n = 0, 1, 2, \ldots$$

$$T_M^{n+1} = T_M^n + \frac{2\kappa\Delta t}{\Delta x^2}\left(T_{M-1}^n - T_M^n\right)\,, \quad n = 0, 1, 2, \ldots$$

$$T_m^0 = T^0(x_m)\,, \quad m = 0, \ldots, M$$

Based on the previous considerations, it should not be a surprise that the truncation error of the FTCS scheme is like $\mathcal{O}\left(\Delta t, \Delta x^2\right)$. However, we can derive the truncation error formally, as follows

$$\tau_{\Delta t,\,\Delta x} = \frac{T(x_m, t_{n+1}) - T(x_m, t_n)}{\Delta t} - \kappa \frac{T(x_{m+1}, t_n) - 2T(x_m, t_n) + T(x_{m-1}, t_n)}{\Delta x^2}$$

with $T(x,t)$ a sufficiently smooth, exact solution to the heat equation. So,

$$\begin{aligned}\tau_{\Delta t,\,\Delta x} &= \frac{T(x, t+\Delta t) - T(x,t)}{\Delta t} - \kappa \frac{T(x+\Delta x, t) - 2T(x,t) + T(x-\Delta x, t)}{\Delta x^2} \\ &= \frac{\partial T}{\partial t}(x,t) + \frac{1}{2}\Delta t \frac{\partial^2 T}{\partial t^2}(x,t) - \kappa \frac{\partial^2 T}{\partial x^2}(x,t) - \frac{\kappa \Delta x^2}{12}\frac{\partial^4 T}{\partial x^4}(x,t) + \ldots \\ &= \frac{1}{2}\Delta t \frac{\partial^2 T}{\partial t^2}(x,t) - \frac{\kappa \Delta x^2}{12}\frac{\partial^4 T}{\partial x^4}(x,t) + \ldots\end{aligned}$$

and, thus

$$\tau_{\Delta t,\,\Delta x} = \mathcal{O}\left(\Delta t, \Delta x^2\right)$$

Hence, the FTCS scheme is first order in Δt and second order in Δx. In practice, this means we can discretize the spatial domain quite coarsely but we must discretize the time interval more finely in order to achieve a required accuracy.

Exercise 3.2.11 Compute the truncation error if $\kappa = \frac{1}{6}\Delta x^2/\Delta t$.
(*Hint*: first derive $T_{tt} = \kappa^2 T_{xxxx}$.)

This error analysis shows how well the FTCS scheme approximates the original diffusion equation in the limit of $\Delta x, \Delta t \to 0$. This is called **consistency**. However, it does *not* show how well the numerical solution approximates the exact solution of the heat equation. This is a matter of **convergence**. In the study of convergence of a finite difference method, there are two issues to be considered. Firstly, whether the numerical approximation is *consistent* with the PDE we wish to solve. Secondly, whether the considered numerical recipe is *stable*, i.e. its numerical solution remains bounded in case of endless repeating of this recipe, with fixed Δt and Δx. This topic of **stability** will be investigated in some detail in the next section. Similar to the time integration of ODEs, as discussed in Chapter 2, finite difference methods for PDEs follow the same approach for checking convergence, namely

$$\mathcal{O}(\Delta t^p, \Delta x^q) \text{ truncation error} + \text{stability} \Rightarrow \mathcal{O}(\Delta t^p, \Delta x^q) \text{ global error}$$

where the integers p and q indicate the **order of accuracy** in time and space, respectively. This statement is associated with the fundamental **Lax equivalence theorem** of finite difference methods; see e.g. Richtmyer and Morton (1967, p. 45). It states that for a *consistent* finite difference method for a well posed linear PDE, the method is *convergent* if and only if it is *stable*.

3.2.3 Stability

We may write the FTCS scheme in the matrix-vector notation, as follows

$$\vec{T}^{n+1} = (I + \Delta t A)\,\vec{T}^{n} + \Delta t\,\vec{g}$$

with $\vec{T}^n = (T_1^n, \cdots, T_M^n)^{\mathrm{T}}$ being the vector representing the numerical solution at time step n and A is the discretization matrix as given by Eq. (3.2.9). Matrix I is the identity matrix. In analogy to the time integration of system of ODEs (see Section 2.2.3), we could check stability by considering the eigenvalues of the matrix $I + \Delta t A$. However, the matrices A and I are an $M \times M$ matrix with $M = L/\Delta x$ so that its dimension is growing as $\Delta x \to 0$. So, we might not be able to determine eigenvalues of such a large matrix, because of the cumbersome and laborious algebra.

Instead, stability can be studied by means of the Fourier analysis (Bracewell, 1978). This analysis is based on the Fourier decomposition of numerical errors and was first developed by John Von Neumann. At present this is the most widespread applied method to check the stability of finite difference schemes for PDEs and is known as the **Von Neumann stability analysis**. This method is particular appropriate for *linear PDEs with constant coefficients where the influence of boundary conditions is ignored* [3]. This is the case either for an infinite domain or for periodic boundary conditions on a finite domain. We then obtain *necessary and sufficient* conditions for stability. However, the analysis to find stability conditions might be difficult or even impossible as soon as we have to deal with nonlinear problems or PDEs with non-constant coefficients, since then the principle of superposition is no longer valid. In such cases, we must *linearize* the problem by freezing the nonlinear terms or non-constant coefficients of the PDEs. As a consequence, Von Neumann stability is a *necessary* condition but *not sufficient* as the analysis is incomplete. Hence, satisfying the Von Neumann stability condition does not guarantee a scheme becomes (conditionally) stable, while not satisfying this condition guarantees instability. Fortunately, the Von Neumann condition appears to be often sharp, and therefore the analysis can be regarded in many cases as a good guess to find a potential stability limit.

The Von Neumann method for stability analysis usually considers stability of numerical schemes that is closely associated with the amplification of *numerical errors* due to round-off errors and to discretization errors. A finite difference scheme is stable, in the limit of $\Delta t \to 0$ and $\Delta x \to 0$, if this amplification made at one time step of the calculation does not cause the errors to increase *too large* as the computation is continued. However, in this context, the numerical modes may grow in time for finite values of Δt (known as the Von Neumann condition). In this reader we follow the *practical* definition of stability, introduced in Richtmyer and Morton (1967, p. 270), which is based on the time behaviour of the *solution* itself instead of the error's behaviour. This definition states that the numerical solution remains uniformly bounded (more precisely, all Fourier components of the numerical solution decay in time) as the time stepping is repeated till a fixed end time

[3]Instabilities are usually generated far from boundaries.

while $\Delta t \to 0$ (and, of course, $\Delta x \to 0$). This practical stability criterion simply prevents numerical modes from growing faster than the physical modes.

The considered heat equation, Eq. (3.2.1a), is linear and has constant coefficients. The same holds for the FTCS scheme, Eq. (3.2.10). However, for the application of the Von Neumann method to the FTCS scheme we must also assume the following periodic boundary condition,

$$T(0,t) = T(L,t)\,, \quad t > 0$$

This boundary condition is discretized, as follows

$$T_0^{n+1} = T_M^{n+1}\,, \quad n = 0, 1, 2, \ldots$$

As a consequence, the numerical solution T_m^n can be expanded as a discrete Fourier series in space at each time level n. In other words, we decompose it into the sum of a finite number of harmonics, on the interval $[0, L)$, as

$$T_m^n = \sum_{j=0}^{M-1} \tilde{T}_j^n e^{\mathrm{i}m\phi_j}$$

where $\tilde{T}_j^n$ is the amplitude of the jth harmonic of T_m^n, i is the imaginary unit, and

$$\phi_j \equiv \frac{2\pi j}{M} = k_j\,\Delta x$$

is the phase angle covering the frequency domain $[0, 2\pi)$ in steps of $2\pi/M$, and $k_j = 2\pi j/L$ is the wave number of the jth harmonic (recall $\Delta x = L/M$). (See Appendix B for background details.) The first harmonic $j = 0$ represents a constant function in space (see Figure 3.2). The second harmonic $j = 1$ has the largest wave length of L and the associated wave number is thus $k_1 = 2\pi/L$. The harmonic $j = M/2$ represents the shortest wave resolvable on a grid with spacing Δx, and its wave length is $2\Delta x$. Hence, $k_{M/2} = 2\pi/2\Delta x = \pi M/L$ and $\phi_j = \pi$. Note that not all harmonic waves can be represented correctly on the grid containing a *finite* number of grid points. The reason is that there is no data in between the grid points. This is illustrated in Figure 3.2. In this example, $M = 6$ and so the mesh contains 7 grid points. All the harmonics running from $j = 0$ till $j = 3 = M/2$ are represented correctly. The other waves, i.e. $j = 4, \ldots, M = 6$, show up at lower frequencies! This phenomenon is known as **aliasing**.

To study the stability properties of the FTCS scheme we consider the time evolution of the amplitude of the harmonics. Since this scheme is *linear*, the evolution of each harmonic of the Fourier series is the same as series itself. Hence, it is enough to consider a *single* harmonic and its time evolution,

$$T_m^n = \tilde{T}^n\, e^{\mathrm{i}m\phi} \tag{3.2.11}$$

where the index j has been dropped. To find out how the amplitude $\tilde{T}^n$ varies in steps of time, substitute Eq. (3.2.11) into Eq. (3.2.10), giving

$$\tilde{T}^{n+1}e^{\mathrm{i}m\phi} = \tilde{T}^n e^{\mathrm{i}m\phi} + \frac{\kappa\Delta t}{\Delta x^2}\left(\tilde{T}^n e^{\mathrm{i}(m+1)\phi} - 2\tilde{T}^n e^{\mathrm{i}m\phi} + \tilde{T}^n e^{\mathrm{i}(m-1)\phi}\right)$$

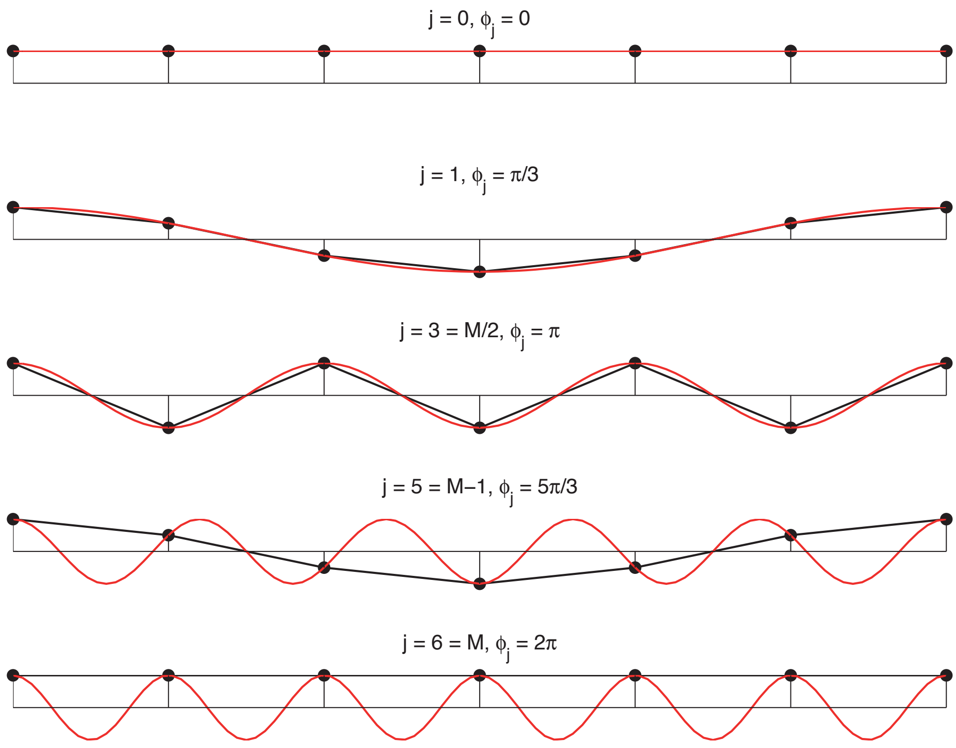


Figure 3.2: This grid contains 6 cells. The harmonics with phase angles $\phi_0 = 0$ and $\phi_6 = 2\pi$, as represented by the red lines, are indistinguishable at this grid, as indicated by the black lines. The same holds for the harmonics with $\phi_1 = \pi/3$ and $\phi_5 = 5\pi/3$. The wave with the highest frequency, $\phi_3 = \pi$, the so-called $\pi-$mode, is the shortest wave resolvable on the grid; its length is $2\Delta x$.

or, dividing by $e^{\mathrm{i}m\phi}$,

$$\tilde{T}^{n+1} = \tilde{T}^{n} + \frac{\kappa\Delta t}{\Delta x^2}\,\tilde{T}^{n}\left(e^{\mathrm{i}\phi} - 2 + e^{-\mathrm{i}\phi}\right)$$

Using the identity

$$\cos\phi = \frac{e^{\mathrm{i}\phi} + e^{-\mathrm{i}\phi}}{2}$$

we obtain

$$\tilde{T}^{n+1} = \left[1 + \frac{2\kappa\Delta t}{\Delta x^2}\,(\cos\phi - 1)\right]\tilde{T}^{n}$$

We recognize this equation as a recurrent relation $\tilde{T}^{n+1} = r\tilde{T}^{n}$ with r the amplification factor as given by

$$r = \frac{\tilde{T}^{n+1}}{\tilde{T}^{n}} = 1 + \frac{2\kappa\Delta t}{\Delta x^2}\,(\cos\phi - 1)$$

For stability we must require that $|r| \le 1$ for all phase angles. Note that the $\phi = 0$ mode (a constant harmonic) makes no contribution to the stability, since it is neutrally stable.

Since $\cos\phi - 1 \leq 0$, we have $r \leq 1$ and the only possibility for instability occurs for $r < -1$. The minimum value of r takes place if $\phi = \pi$, in which case $\cos\phi - 1 = -2$. For this worst case we then have

$$r = 1 - \frac{4\kappa\Delta t}{\Delta x^2}$$

So, requiring $r \geq -1$ yields

$$\boxed{\frac{\kappa\Delta t}{\Delta x^2} \leq \frac{1}{2}} \tag{3.2.12}$$

Eq. (3.2.12) gives the stability requirement for the FTCS scheme as applied to 1D heat equation. It states that for a given Δx, the allowed value of Δt must be small enough to satisfy Eq. (3.2.12). Recall that this stability limit is formally a *necessary* condition. It may not be regarded as *sufficient* since we did not included the impact of the boundary conditions, Eqs. (3.2.1c) and (3.2.1d), on stability. (In practice, however, instabilities often begin far from boundaries and it may require a long time before the boundary conditions can affect these instabilities.) Thus, in order to establish the convergence of the FTCS scheme, it is necessary to use some other methods, which in turn could imply further stability limitations. For instance, we may find another stability condition based on the concept of *positiveness*.

From a physical point of view, the solution to the heat equation, Eqs. (3.2.1a)−(3.2.1d), cannot be negative, i.e. we must have $T(x,t) \geq 0\,, \forall x \in [0, L]\,, \forall t \geq 0$. This implies that our finite difference scheme should share the same property, so that the numerical solution remains non-negative as time progresses. As a consequence, **spurious oscillations** (wiggles) will not occur. It is well known that in the case of diffusion processes, wiggles in the solution will blow up and thus will be unbounded in finite time. To prevent this instability we have to require that our numerical scheme should not exhibit spurious oscillations. Recall Eq. (3.2.10). We can rewrite this equation as follows

$$T_m^{n+1} = \left(1 - \frac{2\kappa\Delta t}{\Delta x^2}\right) T_m^n + \frac{\kappa\Delta t}{\Delta x^2}\left(T_{m+1}^n + T_{m-1}^n\right)$$

Assume that $T_m^n \geq 0$ for all grid points $m = 1, \ldots, M-1$. If

$$1 - \frac{2\kappa\Delta t}{\Delta x^2} \geq 0 \quad \Rightarrow \quad \frac{\kappa\Delta t}{\Delta x^2} \leq \frac{1}{2}$$

then $T_m^{n+1} \geq 0$ for all grid points $m = 1, \ldots, M-1$. Hence, by induction it follows that the numerical solution is non-negative at all times. So, Eq. (3.2.12) appears also to be a *sufficient* stability condition for the FTCS scheme as applied to 1D heat equation.

Exercise 3.2.12 Verify this condition for grid point $m = M$.

3.2.4 Implicit schemes

So far we have considered the explicit FTCS scheme for the numerical solution of 1D heat equation. Instead, we may apply the $\theta-$scheme for the time integration, which yields

$$\frac{T_m^{n+1}-T_m^n}{\Delta t} = \kappa\left(\theta\frac{T_{m+1}^{n+1}-2T_m^{n+1}+T_{m-1}^{n+1}}{\Delta x^2}+(1-\theta)\frac{T_{m+1}^{n}-2T_m^{n}+T_{m-1}^{n}}{\Delta x^2}\right)\,,\quad n=0,1,2,\ldots$$

for all internal points $m = 1, \ldots, M-1$.

Exercise 3.2.13 Apply the $\theta-$scheme at points $m = 0$ and $m = M$.

If $\theta = 0$, the FTCS scheme is recovered. For any other value of $\theta \in (0,1]$ the resulting scheme is implicit.

Exercise 3.2.14 Explain this implicitness.

Common choices are the Crank-Nicolson scheme ($\theta = 1/2$) and implicit Euler ($\theta = 1$). Their stencils are depicted in Figure 3.3.

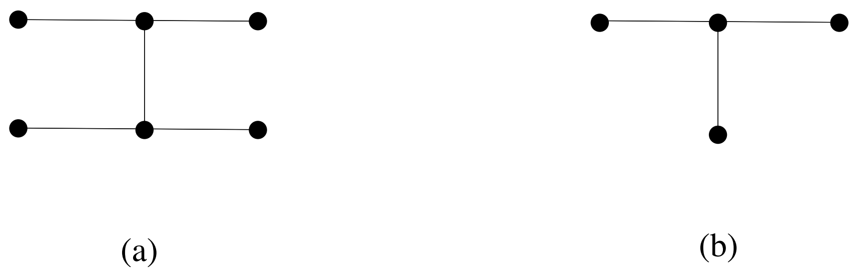


Figure 3.3: The stencils of (a) the Crank-Nicolson scheme and (b) the implicit Euler scheme.

Exercise 3.2.15 Verify the stencils.

Exercise 3.2.16 Compute the truncation error $\tau_{\Delta t,\,\Delta x}$ of both the Crank-Nicolson and implicit Euler schemes.

Exercise 3.2.17 Show by means of the Von Neumann method that the $\theta-$scheme is unconditionally stable if $1/2 \le \theta \le 1$.

A main disadvantage of applying an *implicit* scheme is that the numerical solution is less straightforward to obtain. If we write the $\theta-$scheme in the matrix-vector notation,

$$(I-\theta\Delta t A)\,\vec{T}^{n+1} = (I+(1-\theta)\Delta t A)\,\vec{T}^{n}$$

we see that the solution $\vec{T}^{n+1}$ cannot be computed immediately. Instead, we need to solve a linear system of M equations for the M unknowns at the new time step. For this special case, the matrix $I-\theta\Delta t A$ is a **tri-diagonal** one which is appropriate for the purpose of applying Gaussian elimination. We shall not go into details and refer again to Golub and Van Loan (1989) for details.

Exercise 3.2.18 Verify the above system of equations.

3.2.5 Accuracy and relaxation time

With the truncation error only some *qualitative* trends in the accuracy of solutions can be deduced. For instance, the error of a second order accurate scheme in space is proportional to Δx^2 and hence is much smaller than the error in a first order approximation when Δx is small. Based on this, we can only say that if Δx becomes two times smaller then the truncation error or the global error will be smaller by a factor of four. So, no information on the *actual* error in the numerical solution is given for a certain value of Δx. A way to quantify the accuracy of solutions is by considering how well the *rate of diffusion* is represented numerically. For this, we introduce the notion of **relaxation time**, after which the amplitude of the solution has been decreased by a factor of e^{-1} due to the diffusive process.

Let us consider the exact solution $T(x,t)$ of the heat equation, Eq. (3.2.1a), on the interval $x \in [0, L]$ and $t > 0$. We assume it can be decomposed into an infinite number of Fourier modes with different wave lengths, $L, {}^1\!/\!{}_2L, {}^1\!/\!{}_3L, \ldots$ and one constant term, as follows

$$T(x,t) = \sum_{j=-\infty}^{\infty} \hat{T}_j(t) e^{2\pi \mathrm{i} jx/L}$$

with $\hat{T}_j$ the amplitudes of the exact solution. Since the considered PDE is linear, it is sufficient to restrict to a single Fourier mode,

$$T(x,t) = \hat{T}(t) e^{\mathrm{i}kx}$$

with $k = 2\pi/L$ the wave number.

Exercise 3.2.19 Explain why we can consider just a single Fourier mode in the case of a linear PDE.

Substitution into Eq. (3.2.1a) yields an ordinary differential equation for $\hat{T}(t)$

$$\frac{d\hat{T}}{dt} + \kappa k^2 \hat{T} = 0$$

A solution to this ODE is given by

$$\hat{T}(t) = e^{-\kappa k^2 t}$$

The corresponding relaxation time, denoted as τ, can be determined as follows

$$-\kappa k^2 \tau = -1 \quad \Rightarrow \quad \tau = \frac{1}{\kappa k^2} = \frac{L^2}{4\pi^2 \kappa}$$

Exercise 3.2.20 Let us consider a rod of 1 meter long. A typical thermal diffusivity coefficient is $\kappa = 10^{-4}$ m^2/s. Calculate the corresponding relaxation time.

Exercise 3.2.21 Explain why a larger diffusion rate results in a shorter time scale.

The main interest would be how well the rate of diffusion is represented by a numerical scheme. The amplitude of the numerical solution is governed by the recurrent relation

$$\tilde{T}^n = r^n \tilde{T}^0$$

This numerical amplitude is supposed to be decreased and we may expect that

$$\frac{\tilde{T}^n}{\tilde{T}^0} = r^n = e^{-t_n/\tau'} = e^{-n\Delta t/\tau'}$$

with τ' the *numerical* relaxation time. Hence, after a *single* time step, the amplitude is decreased by a factor of $e^{-\Delta t/\tau'}$, or

$$r = e^{-\Delta t/\tau'} \quad \Rightarrow \quad \tau' = -\frac{\Delta t}{\ln r}$$

For a given numerical scheme, the amplification factor r may depend on the phase angle ϕ and a quantity q as defined by

$$q = \frac{\kappa \Delta t}{\Delta x^2}$$

As an example, the amplification factor of the FTCS scheme is given by

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

Exercise 3.2.22 Verify this amplification factor.

Exercise 3.2.23 Derive the amplification factor for the $\theta-$scheme. Show that it depends on both ϕ and q.
(*Hint*: see Exercise 3.2.17.)

By comparing with the (exact) relaxation time τ, we measure the numerical diffusion rate with respect to the exact one. So, if $\tau > \tau'$, then the diffusion rate as given by the numerical scheme is larger than the true rate of diffusion, while if $\tau < \tau'$, the numerical diffusion rate is less than the exact one. We define the **relative relaxation factor** as τ/τ', and is given by

$$\frac{\tau}{\tau'} = -\frac{L^2 \ln r}{4\pi^2 \kappa \Delta t}$$

To get an idea of the accuracy of the rate of diffusion, we shall consider the relative relaxation factor with respect to the *number of grid points per wave length*. This number is denoted as N. We would then expect that

$$\lim_{N\to\infty} \frac{\tau}{\tau'} = 1$$

So, the more grid points per wave length we choose, the more accurate the numerical representation of the diffusive process will be. However, we must realize that this makes the simulation more time consuming as well. Generally, there is a trade-off between the accuracy of computation and the computational cost.

In our case, L is the wave length for the considered Fourier mode, thus $N = L/\Delta x = M$. The phase angle is then given by

$$\phi = k\Delta x = \frac{2\pi}{M}$$

We obtain a final formula for the relative relaxation factor, as follows

$$\boxed{\frac{\tau}{\tau'} = -\frac{\ln\, r(q,\phi)}{q\phi^2}}$$

Exercise 3.2.24 Verify this formula.

For a given q, we can plot the relative relaxation factor as a function of the number of grid points per wave length N for different numerical schemes. Some plots are depicted in Figure 3.4. From this figure we conclude, for instance, that implicit Euler has generally

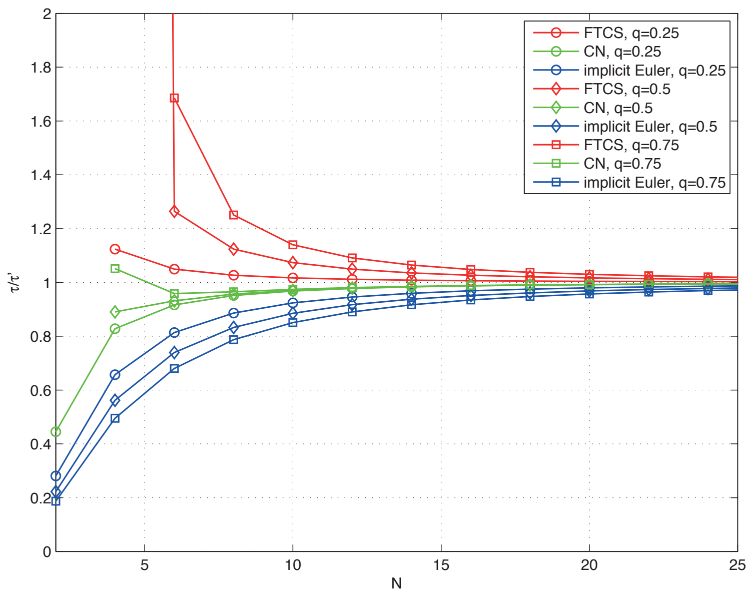


Figure 3.4: Relative relaxation factor as function of the number of grid points per wave length for different values of q and for various numerical schemes.

a lower diffusion rate compared to that of the Crank-Nicolson scheme and the explicit Euler scheme. Also, as expected, for a given level of τ/τ', the Crank-Nicolson scheme needs to use less number of grid points per wave length compared to those of the explicit and implicit Euler schemes. In other words, the Crank-Nicolson scheme is more accurate than both the other schemes. A more important issue related to accuracy that we may conclude from the figure is that apparently at least 20 grid points per wave length seems to be necessary in order to represent a sufficiently accurate diffusive process by any of the considered numerical schemes.

Exercise 3.2.25 Explain why the choice $q = 0.75$ does not make sense for the FTCS scheme.

Exercise 3.2.26 Given a rod of 1 m and $\kappa = 10^{-4}$ m^2/s. We choose the FTCS scheme. What reasonable choices of Δt and Δx would you make if we require a maximum error in the relaxation time of 10%.

3.3 Convection equation

In this section we discuss the numerical solution of the 1D convection equation as given by

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0 \tag{3.3.1}$$

This first order hyperbolic PDE describes the propagation of any disturbance $c(x,t)$ with a constant speed u in an infinite long 1D domain. Given any initial condition $c(x,0) = c^0(x)$ the solution at any *later* time is given by $c(x,t) = c^0(x - ut)$. This can be verified by computing the $t-$ and $x-$derivatives of the solution and substituting into the equation above. The function $c^0(x - ut)$ is a solution for **right-travelling waves**, since the graph of $c^0(x - ut)$ is simply the graph of $c^0(x)$ shifted to the right by ut spatial units. As time increases, the profile $c^0(x)$ moves to the right at speed u. Thus solutions are simply propagated, or **convected**, with constant speed u without *deforming*, i.e. no change in shape. If $u > 0$, propagation is from left to right. If $u < 0$, propagation is from right to left. This phenomenon is also known as **transport** or **advection** in some literature.

Hyperbolic equations can be solved by the **method of characteristics**. Characteristics are curves $x = x_c(t)$ in the space of independent variables (x,t), the so-called $t - x$ plane, on which the solution c is constant. That is $c(x_c(t),t) =$ constant. From the characteristics and the initial condition $c^0(x)$, one can find the solution at any later time t by using the characteristics to *look up* the value of c.

In our specific case, the curves $x_c(t)$ are given by

$$\frac{dx_c}{dt} = u$$

One can verify that c on these curves is constant by differentiating $c(x_c(t),t)$ with respect to t, as follows

$$\frac{dc}{dt} = \frac{\partial c}{\partial x}\frac{dx_c}{dt} + \frac{\partial c}{\partial t} = u\frac{\partial c}{\partial x} + \frac{\partial c}{\partial t} = 0$$

The characteristic is

$$x = x_c(t) = ut + x_0$$

with x_0 the position of the disturbance at $t = 0$. The *shape of the solution* $c(x,t)$ is given by $c^0(x_0)$, or $c(x,t) = c^0(x - ut)$ for any initial condition $c^0(x)$. This means the shape of the initial condition remains unchanged as time proceeds while its position changes. See Figure 3.5 as an example.

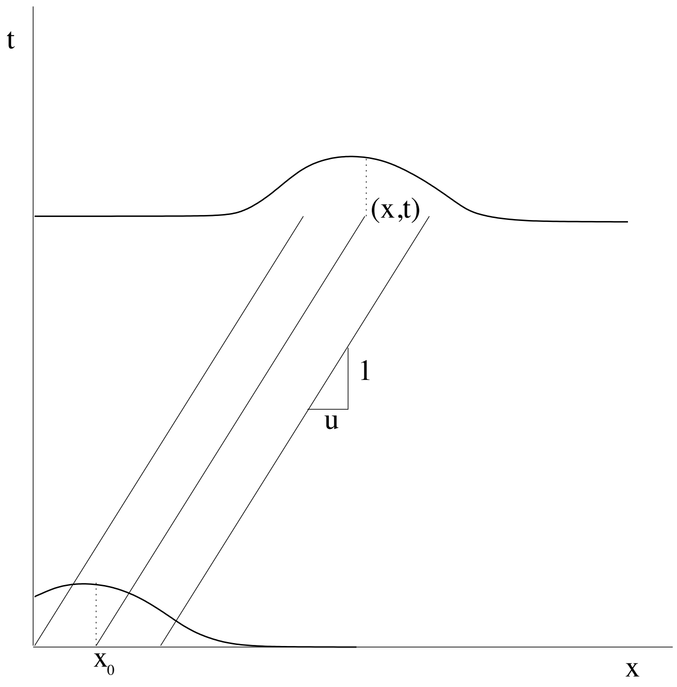


Figure 3.5: An example of the application of the method of characteristics for the convection equation with constant propagation speed u.

So far we only consider an initial value problem. Let us consider a 1D domain with a finite length L. In this case we need to impose some boundary conditions. Using the concept of characteristics, a boundary condition is needed at a certain location where a characteristic *enters* the domain. A basic rule of thumb is the following.

The number of boundary conditions is determined by the number of characteristics entering the domain.

In our case, we can only impose a boundary condition at a boundary with the *ingoing* characteristic. In turn this depends on the propagation velocity. If $u > 0$, the ingoing characteristic is at the left boundary, $x = 0$, while if $u < 0$, the ingoing characteristic is at the right boundary, $x = L$.

Exercise 3.3.1 Consider the convection equation, Eq. (3.3.1), with $u = 1$ m/s, and the following initial and boundary conditions

$$c(x,0) = G(x)\,, \quad 0 \leq x \leq L$$

$$c(0,t) = F(t)\,, \quad t > 0$$

The solution is given by

$$c(x,t) = \begin{cases} G(x-t)\,, & x \geq t \\ \\ F(t-x)\,, & x < t \end{cases}$$

Verify this by means of the method of characteristics.
(*Hint*: Draw the $t - x$ plane and indicate the location of both the initial and boundary conditions.)

3.3.1 Space discretization and time integration

We reconsider the computional grid as discussed in Section 3.2. We apply the MOL approach. A possible candidate for the spatial approximation would be central differences (cf. Eq. (3.2.4)), as follows

$$\frac{dc_m(t)}{dt} + u\frac{c_{m+1}(t) - c_{m-1}(t)}{2\Delta x} = 0\,, \quad m = 1, \ldots, M-1\,, \quad t > 0 \tag{3.3.2}$$

This is a semi discretization of Eq. (3.3.1) for the *internal* grid points of the computational domain.

Exercise 3.3.2 Show that this semi-discretized equation is consistent with the considered convection equation and determine the order of accuracy.

The resulted system of first order ODEs with the unknowns $c_m(t)$ can be written as

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

with $\vec{c} = (c_1, \cdots, c_{M-1})^{\mathrm{T}}$ being the vector representing the unknowns and matrix A is the $(M-1) \times (M-1)$ discretization matrix as given by

$$A = \frac{u}{2\Delta x}\begin{pmatrix} 0 & -1 & 0 & & \cdots & & 0 \\ 1 & 0 & -1 & & & & \\ 0 & 1 & 0 & -1 & & & \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ & & & 1 & 0 & -1 & 0 \\ & & & & 1 & 0 & -1 \\ 0 & & & \cdots & & 1 & 0 \end{pmatrix}$$

This matrix is a real **skew symmetric** one ($A = -A^T$) implying that the nonzero eigenvalues are all purely imaginary. Hence, the system can be regarded as *undamped*. As a consequence, for time integration, the explicit Euler scheme would not be appropriate as it is *unstable* for all time steps Δt.

Exercise 3.3.3 Verify this via the Von Neumann stability analysis.

If we choose the midpoint rule to integrate in time, we get

$$\frac{c_m^{n+1} - c_m^{n-1}}{2\Delta t} + u\frac{c_{m+1}^n - c_{m-1}^n}{2\Delta x} = 0\,, \quad m = 1, \ldots, M-1\,, \quad n = 0, 1, 2, \ldots \tag{3.3.3}$$

This *explicit* scheme is expected to be *conditionally stable*. This can be verified by means of the Von Neumann method. Assume c_m^n of the form

$$c_m^n = c^0 r^n e^{\mathrm{i}m\phi}$$

Substituting this into the scheme, Eq. (3.3.3), gives the corresponding characteristic equation for the amplification factor r,

$$r^2 - 1 + \sigma r \left(e^{\mathrm{i}\phi} - e^{-\mathrm{i}\phi}\right) = 0$$

with

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

known as the **Courant number**. The roots of the characteristic equation are given by

$$r_1 = -\mathrm{i}\sigma \sin\phi + \sqrt{1 - \sigma^2 \sin^2\phi}\,, \quad r_2 = -\mathrm{i}\sigma \sin\phi - \sqrt{1 - \sigma^2 \sin^2\phi}$$

Exercise 3.3.4 Verify these roots.
(*Hint*: use the identity $e^{\mathrm{i}\phi} - e^{-\mathrm{i}\phi} = 2\mathrm{i}\sin\phi$.)

Both roots have an absolute value of 1 for all ϕ if and only if $|\sigma| \le 1$ or

$$\boxed{\frac{|u|\Delta t}{\Delta x} \le 1}$$

This stability condition is known as the **CFL condition**. We shall discuss this important issue in Section 3.3.3.

The above scheme, Eq. (3.3.3), has a special feature in the sense that the coefficient c_m^n is not involved! This scheme is called the **leapfrog** scheme. The corresponding stencil is depicted in Figure 3.6. Another way of looking at this remarkable scheme is that it constitutes two sets of *independent* difference equations. One set is defined at the points $(2m, 2n)$ and $(2m+1, 2n+1)$ while the other set consists of the points $(2m+1, 2n)$ and $(2m, 2n+1)$. By cancelling one of these two sets we gain efficiency by a factor of two without losing accuracy.

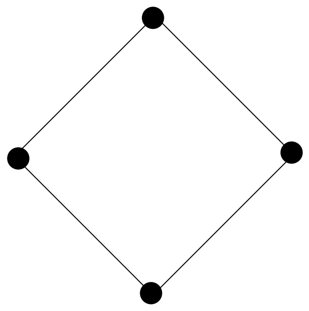


Figure 3.6: The stencil of the leapfrog scheme.

Exercise 3.3.5 Show that the leapfrog scheme is consistent with Eq. (3.3.1) and that its truncation error is $\tau_{\Delta t,\,\Delta x} = \mathcal{O}(\Delta t^2, \Delta x^2)$.

Special consideration must be given to *starting* the leapfrog scheme. For example, when specifying the initial condition, we now need an initial condition for both c_m^0 and c_m^{-1}. These initial conditions must be carefully chosen to be consistent with the PDE.

Another issue related to the leapfrog scheme that must be considered is that central differences cannot be applied at the boundary with *outgoing* characteristics as it introduces a virtual grid point. This grid point cannot be eliminated because there are no boundary conditions at this boundary location. This problem can be circumvented by applying *locally* an upwind method. This will be discussed later.

We could have approximate the time derivative in Eq. (3.3.2) with implicit schemes, e.g. the $\theta-$scheme with $1/2 \leq \theta \leq 1$. They are usually unconditionally stable. However, there is a price to pay for the improved stability of implicit schemes, that is, we must solve a linear system of algebraic equations which might be very time consuming.

3.3.2 The Preissmann scheme

A popular scheme for the convection equation, Eq. (3.3.1), is the so-called **Preissmann** scheme or sometimes called the **box** scheme. This scheme can be constructed by applying the trapezoidal rule in both time and space. The semi-discretized equation is given by

$$\frac{1}{2}\left(\frac{dc_m}{dt} + \frac{dc_{m-1}}{dt}\right) + u\frac{c_m - c_{m-1}}{\Delta x} = 0$$

Next we carried out the time integration, as follows

$$\frac{1}{2}\left(\frac{c_m^{n+1} - c_m^n}{\Delta t} + \frac{c_{m-1}^{n+1} - c_{m-1}^n}{\Delta t}\right) + \frac{1}{2}u\left(\frac{c_m^{n+1} - c_{m-1}^{n+1}}{\Delta x} + \frac{c_m^n - c_{m-1}^n}{\Delta x}\right) = 0$$

This approximation can be rewritten as

$$c_m^{n+1} - c_m^n + c_{m-1}^{n+1} - c_{m-1}^n + \sigma\left(c_m^{n+1} + c_m^n - c_{m-1}^{n+1} - c_{m-1}^n\right) = 0$$

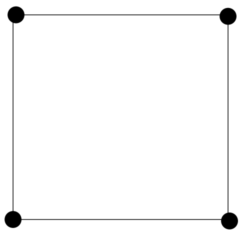


Figure 3.7: The stencil of the Preissmann scheme.

The corresponding stencil is plotted in Figure 3.7. The scheme is **compact** which means that it has the smallest stencil possible while its order of accuracy is two in time and space[4].

Exercise 3.3.6 Show that indeed $\tau_{\Delta t,\,\Delta x} = \mathcal{O}(\Delta t^2, \Delta x^2)$.

Exercise 3.3.7 Show that the Preissmann scheme is implicit and unconditionally stable.

A distinguishing feature of the Preissmann scheme is that the unknown at new time step can be computed *explicitly*! Indeed, by rewriting the box scheme as

$$c_m^{n+1} = c_{m-1}^n + \frac{1-\sigma}{1+\sigma}\left(c_m^n - c_{m-1}^{n+1}\right)$$

we see that c_m^{n+1} can be computed immediately, since c_{m-1}^{n+1} has just been computed already! (It is assumed that the ordering of unknowns through computational grid is from $m = 0$ till $m = M$.)

Another advantage is that the box scheme has no difficulty with the lack of boundary condition at the boundary with outgoing characteristics, since it does not introduce virtual grid points.

Exercise 3.3.8 Verify this.

The box scheme can be generalized with the $\theta-$scheme, with $1/2 \leq \theta \leq 1$, as follows

$$\frac{1}{2}\left(\frac{c_m^{n+1} - c_m^n}{\Delta t} + \frac{c_{m-1}^{n+1} - c_{m-1}^n}{\Delta t}\right) + u\left(\theta\frac{c_m^{n+1} - c_{m-1}^{n+1}}{\Delta x} + (1-\theta)\frac{c_m^n - c_{m-1}^n}{\Delta x}\right) = 0$$

This scheme is called the $\theta-$**box** scheme. The Preissmann scheme is obtained with $\theta = 1/2$.

Exercise 3.3.9 Show by means of the Von Neumann method the *unconditional instability* of this $\theta-$box scheme when $0 \leq \theta < 1/2$.

[4]For many schemes, it is true that the higher order of accuracy requires more coefficients involved and makes thus the stencil larger. These schemes are thus not compact. See Hirsch (1990).

3.3.3 The CFL condition

The CFL condition is a *necessary* condition for the *convergence* of a finite difference scheme to a (non)linear hyperbolic PDE. This condition is named after Courant, Friedrichs and Lewy who published the result in 1928. It is one of the most important results – historically and practically – in numerical solutions of PDEs. For the 1D case, the CFL condition has the following form

$$|\sigma| \equiv \frac{|u|\Delta t}{\Delta x} \leq C$$

where C is a dimensionless constant which depends only on the numerical scheme to be applied. For the leapfrog scheme this is $C = 1$.

The CFL condition is based on the concept of **domain of dependence**. Consider a point P on the computational grid in the $t-x$ plane; see Figure 3.8. Let $\tilde{c}_{\text{P}}$ be the numerical

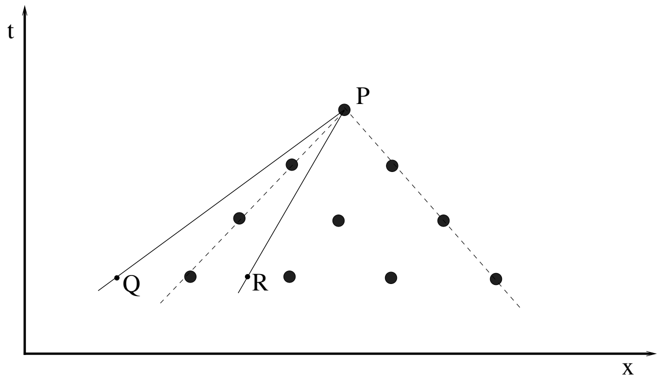


Figure 3.8: Numerical domain of dependence for the leapfrog scheme as bounded by dashed lines. Two possible characteristics are shown in drawn lines. Line PR corresponds larger $1/u$ and the CFL condition is satisfied while PQ corresponds smaller $1/u$ and the CFL condition is violated.

value of c at P. Consider the leapfrog scheme which has the coefficients c_m^{n+1}, c_m^{n-1}, c_{m+1}^n and c_{m-1}^n involved as represented by its stencil. From this follows the *numerical* domain of dependence of P. This contains grid points on which the value of $\tilde{c}_{\text{P}}$ depends.

The PDE itself has a domain of dependence. For Eq. (3.3.1) with constant speed u, the *analytical* domain of dependence is just the line $t = (x - x_0)/u$ which passes through P. The CFL condition states that

a necessary condition of a numerical scheme to converge is that the numerical domain of dependence must contain the analytical domain of dependence.

It is easy to see why this must hold. Let Q be some point in the analytical domain of dependence of P but *not* in the numerical domain of dependence. See Figure 3.8. Let c_{P} be the exact value of the solution to the PDE at point P. The value c_{P} will depend on the initial condition at Q. However, since Q is not in the numerical domain of dependence, the numerical value $\tilde{c}_{\text{P}}$ *cannot* depend on the initial condition at Q. Consider changing the initial condition at Q. As a consequence, c_{P} will change but $\tilde{c}_{\text{P}}$ will not. Hence, there is no possibility of $\tilde{c}_{\text{P}}$ converging to c_{P} as the grid is refined ($\Delta t, \Delta x \to 0$).

Generally, the CFL condition may apply to nonlinear PDEs. However, for linear problems, the Lax equivalence theorem asserts that convergence is equivalent to stability. Therefore the CFL condition can be applied as well to obtain a stability limit, if the underlying PDE is linear.

Consider the leapfrog scheme, its stencil and Figure 3.8. For Eq. (3.3.1) with $u > 0$, the CFL condition is

$$\frac{1}{u} \geq \frac{\Delta t}{\Delta x}$$

In other words, the distance covered during Δt with speed u must be smaller than or equal to Δx: $u\Delta t \leq \Delta x$. Similarly for $u < 0$, the CFL condition is

$$-\frac{1}{u} \leq \frac{\Delta t}{\Delta x}$$

Combining yields

$$\frac{|u|\Delta t}{\Delta x} \leq 1$$

Exercise 3.3.10 Verify this CFL condition.

Note that satisfying this CFL condition does not guarantee the leapfrog scheme converges, while not satisfying this condition guarantees non-convergence (or instability, since Eq. (3.3.1) is linear). Also note that convergence implies CFL condition, which therefore usually refers to an explicit scheme. In fact, according to the CFL condition, there is no explicit, unconditionally stable, finite difference scheme for solving a hyperbolic PDE.

The advantange of the CFL condition is that it is easy to apply. Its weakness, however, is that it provides us with only a *necessary* condition for convergence or stability. This is easy to understand when we consider the unconditionally unstable FTCS scheme for Eq. (3.3.1) (see also Exercise 3.3.3). The CFL condition for this scheme would suggest that $|\sigma| \leq 1$. Hence, though the CFL condition gives us a nontrivial stability condition, it *cannot* be sufficient since the scheme is unstable for all σ. Nevertheless, often, the CFL condition is the same as the stability limit. And so, it is a good place to begin. A final note. The CFL condition is often sharp, i.e. often also sufficient, but it can also be a very bad condition. Therefore it is wise to carry out a more careful analysis, for instance, the Von Neumann stability analysis.

3.3.4 The upwind method

Since we are dealing with information being propagated along a certain direction, it would be better to propose a scheme that gathers information to be propagated in the same direction. For instance, the backward finite difference approximation, Eq. (3.2.3), applied to $\partial c/\partial x$ will gather information from the left to propagate, while the forward finite difference operator, Eq. (3.2.2), gathers from the right. Hence, our knowledge of which direction the wave should propagate indicates the **upwind** direction. If $u > 0$, then propagation of the disturbance is from left to right. So, the upwind direction is *to the left*, i.e. the information is obtained from the **backward** direction. For $u < 0$, the upwind direction is *to the right* and so, the information is from the **forward** direction.

The **upwind difference scheme** is given by

$$\frac{\partial c}{\partial x} = \begin{cases} \dfrac{c_m - c_{m-1}}{\Delta x}\,, & \text{if } u > 0 \\[2ex] \dfrac{c_{m+1} - c_m}{\Delta x}\,, & \text{if } u < 0 \end{cases}$$

and it correctly passes information in the direction of the characteristic of the convection equation. This approximation is called **first order upwinding**.

Exercise 3.3.11 Verify the first order consistency of this scheme to $\partial c/\partial x$.

Application of explicit Euler yields

$$c_m^{n+1} = c_m^n - \sigma \begin{cases} c_m^n - c_{m-1}^n\,, & \text{if } u > 0 \\[2ex] c_{m+1}^n - c_m^n\,, & \text{if } u < 0 \end{cases} \tag{3.3.4}$$

This scheme is called the **FTBS** scheme, which stands for <u>F</u>orward in <u>T</u>ime, <u>B</u>ackward in <u>S</u>pace.

Exercise 3.3.12 Verify Eq. (3.3.4).

The FTBS scheme is first order accurate in both time and space. Depending on the upwind direction, the possible stencils for the FTBS scheme are depicted in Figure 3.9.

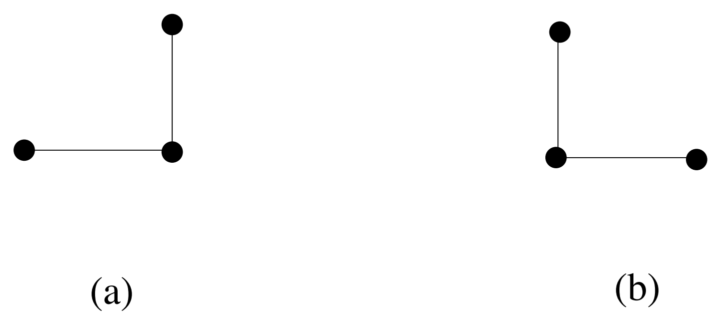


Figure 3.9: The stencil of the FTBS scheme if (a) $u > 0$ and (b) $u < 0$.

Exercise 3.3.13 Verify these stencils.

Exercise 3.3.14 For the FTBS scheme with $u > 0$, verify that

- $\tau_{\Delta t,\,\Delta x} = \mathcal{O}(\Delta t, \Delta x)$, and
- the scheme is stable if $\sigma \leq 1$. Apply both the CFL condition and the Von Neumann stability analysis.

The FTBS scheme does not introduce a virtual grid point at the boundary with the outgoing characteristic.

Exercise 3.3.15 Explain why a first order upwind scheme can be useful at the boundary points while a higher order scheme like leapfrog is applied in the internal points.

One can understand why the FTBS scheme is stable in the following way. Start from this scheme

$$\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_m^n - c_{m-1}^n}{\Delta x} = 0$$

where we have assumed that $u > 0$. Replace the coefficients by their associated exact solutions and apply Taylor series,

$$\frac{\partial c}{\partial t} + \frac{1}{2}\Delta t\frac{\partial^2 c}{\partial t^2} + u\frac{\partial c}{\partial x} - \frac{1}{2}u\Delta x\frac{\partial^2 c}{\partial x^2} + \mathcal{O}(\Delta t^2, \Delta x^2) = 0$$

or, with $c_{tt} = u^2 c_{xx}$,

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} + \frac{1}{2}u\left(u\Delta t - \Delta x\right)\frac{\partial^2 c}{\partial x^2} + \mathcal{O}(\Delta t^2, \Delta x^2) = 0$$

Apparently, the numerical solution produced by the FTBS scheme is consistent up to second order in time and space with the solution to the **convection-diffusion equation**

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} - \kappa\frac{\partial^2 c}{\partial x^2} = 0 \tag{3.3.5}$$

with

$$\kappa = -\frac{1}{2}u(u\Delta t - \Delta x) \tag{3.3.6}$$

The term

$$-\kappa\frac{\partial^2 c}{\partial x^2}$$

adds some diffusion to the equation which is sufficient to stabilize the explicit Euler scheme. This amount of diffusion is *entirely* due to upwinding and is therefore called **numerical diffusion**. The above approach of deriving numerical diffusion for an upwind scheme or any scheme that generates **artificial diffusion** to be consistent up to $\mathcal{O}(\Delta t^2, \Delta x^2)$ with the convection-diffusion equation, Eq. (3.3.5), which is actually the modified one, is called the **modified equation approach**. So, with this approach, we just modify the *original* equation, Eq. (3.3.1), by adding the truncation error of the underlying scheme to obtain Eq. (3.3.5) that may contain some amount of numerical diffusion.

The resulted amount of numerical diffusion, Eq. (3.3.6), can be quite significant so that solutions of the modified equation differ too much from those of the convection equation. Hence, the numerical solution produce by first order upwinding is not sufficiently accurate for practical applications. However, there has been a great deal of research to improve the accuracy of the upwind scheme, by using **higher order upwind** schemes. These higher order schemes aim to obtain at least a second order truncation error. We shall discuss these schemes right after this section.

Recall that the CFL condition $\sigma \leq 1$ is a *necessary* condition for the FTBS scheme. By means of the requirement of positiveness we can deduce a *sufficient* condition. Let us consider the FTBS scheme, Eq. (3.3.4), with $u > 0$, which can be rewritten as

$$c_m^{n+1} = (1-\sigma)c_m^n + \sigma c_{m-1}^n$$

We assume by induction that $c_m^n \geq 0$ and $c_{m-1}^n \geq 0$. Then $c_m^{n+1} \geq 0$ if $1-\sigma \geq 0$. Apparently, $\sigma \leq 1$ is not only necessary for stability but sufficient as well.

Alternatively, we consider the modified equation, Eq. (3.3.5). To prevent a blow up we must require $\kappa \geq 0$. From Eq. (3.3.6), this implies (assuming $u > 0$)

$$-\frac{1}{2}u(u\Delta t - \Delta x) \geq 0 \quad \Rightarrow \quad u\Delta t - \Delta x \leq 0 \quad \Rightarrow \quad \frac{u\Delta t}{\Delta x} \leq 1$$

In other words, we have again obtained the CFL condition. This type of stability analysis based upon the concept of positiveness and the modified equation approach is typically **heuristic** as there is no theoretical basis to proof the correctness of the obtained stability limits. The Von Neumann stability method is a *formal* manner to provide this proof.

As an aside, using the modified equation approach, it is easy to show that the FTCS scheme is indeed unstable. This scheme for Eq. (3.3.1) reads

$$\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_{m+1}^n - c_{m-1}^n}{2\Delta x} = 0$$

We first replace the unknowns with the corresponding exact solutions, after which we apply the Taylor series expansion, and subsequently substitute the time derivative by the space derivative, i.e. $c_{tt} = u^2 c_{xx}$, we then get the following modified equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} + \frac{1}{2}u^2\Delta t\frac{\partial^2 c}{\partial x^2} + \mathcal{O}(\Delta t^2, \Delta x^2) = 0$$

By comparing this equation with Eq. (3.3.5), we conclude that

$$\kappa = -\frac{1}{2}u^2\Delta t < 0$$

which results in a blow up of the numerical solution. Indeed, it is the *forward* Euler scheme that provides the negative diffusion term. Hence, if we simply use a higher order approximation for the convection term with the first order explicit Euler scheme for the time derivative, we shall find that the resulting scheme is unconditionally unstable.

Exercise 3.3.16 Let us consider Eq. (3.3.1). Now we want to approximate this equation using the so-called BTCS scheme, i.e. backward in time and central in space. This scheme combines implicit Euler and central differences. It is expected that this scheme generates some amount of dissipation, otherwise it would not be stable. Concerning this scheme, answer the following questions.

1. Write down this scheme and draw its stencil.

2. Show that $\tau_{\Delta t,\,\Delta x} = \mathcal{O}(\Delta t, \Delta x^2)$.

3. Show its unconditional stability.

4. Compute its numerical diffusion.

3.3.5 Spurious oscillations and monotonicity

Let us go back to the following discretized formulation of the diffusion term $\partial^2 c/\partial x^2$,

$$\frac{c_{m+1} - 2c_m + c_{m-1}}{\Delta x^2}$$

in which central differences have been employed. Next we replace c_m by $c(x_m)$ and we assume that the exact solution $c(x)$ is sufficiently smooth. This enables us to apply Taylor expansions. This gives

$$\frac{\partial^2 c}{\partial x^2} + \underbrace{\frac{1}{12}\Delta x^2 \frac{\partial^4 c}{\partial x^4} + \frac{1}{360}\Delta x^4 \frac{\partial^6 c}{\partial x^6} + \frac{1}{20160}\Delta x^6 \frac{\partial^8 c}{\partial x^8} + \mathcal{O}(\Delta x^8)}_{\text{truncation error}}$$

Apparently, the truncation error only contains *even* derivatives. These derivatives are associated with **dissipation** and they tend to *smear* the solution gradients. Generally, when considering the solution as a sum of harmonics, the higher frequency components being damped more than lower frequency components (see Section 3.3.8). The effect of dissipation therefore is that the solution gradients become less sharper and consequently, the solution becomes more smoother.

Now, let us consider the following central differences applied to the convection term $\partial c/\partial x$,

$$\frac{c_{m+1} - c_{m-1}}{2\Delta x}$$

Again, the unknowns are replaced by their associated exact solutions that are smooth enough. Expansion yields

$$\frac{\partial c}{\partial x} + \underbrace{\frac{1}{6}\Delta x^2 \frac{\partial^3 c}{\partial x^3} + \frac{1}{120}\Delta x^4 \frac{\partial^5 c}{\partial x^5} + \frac{1}{5040}\Delta x^6 \frac{\partial^7 c}{\partial x^7} + \mathcal{O}(\Delta x^8)}_{\text{truncation error}}$$

The truncation error only contains *odd* derivatives. They are associated with **dispersion** and they tend to *sharp* the solution gradients. Dispersion implies that higher frequency components travelling at slower speeds than the lower frequency components (see Section 3.3.8). The effect of dispersion therefore is that often **spurious** oscillations or **wiggles** occur in solutions with sharp gradients, usually with high frequency oscillations trailing the particular effect; see Figure 3.10. These *non-physical* oscillations show solutions where

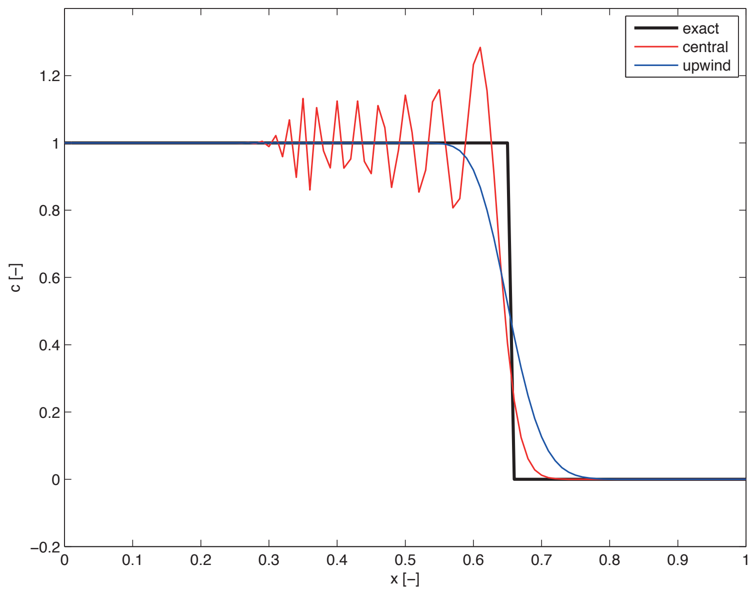


Figure 3.10: Numerical representations of a steep front: central differences and the first order upwind scheme.

wave lengths of two mesh widths can be recognized. Because of these wiggles the computed concentration can be larger than the initial value (here 1). This is called **overshoot**. Also, in some cases, the numerical solution can be negative, which is called **undershoot**.

If, instead of central differences, the upwind method is applied, assuming $u > 0$,

$$\frac{c_m - c_{m-1}}{\Delta x}$$

then its Taylor expansion is given by

$$\frac{\partial c}{\partial x} - \frac{1}{2}\Delta x \frac{\partial^2 c}{\partial x^2} + \frac{1}{6}\Delta x^2 \frac{\partial^3 c}{\partial x^3} - \frac{1}{24}\Delta x^3 \frac{\partial^4 c}{\partial x^4} + \frac{1}{120}\Delta x^4 \frac{\partial^5 c}{\partial x^5} + \mathcal{O}(\Delta x^5)$$

Apparently, the upwind scheme generates both dissipation and dispersion effects. Hence, the steep front is not only propagated but also it becomes less steeper due to numerical diffusion; see Figure 3.10. In this case, the first order upwind scheme creates much dissipation so that it damps out any numerical oscillation created by dispersion.

Summarizing, central differences exhibit either even derivatives in the case of a diffusive process or odd derivatives in the case of a convective process. This is due to its symmetry; it is **symmetric** around its point of consideration ($= x_m$). Because of this property, it cancels out the odd derivatives when diffusion is involved and even derivatives in the case of convection. On the other hand, the first order upwind scheme is not symmetric. This scheme is said to be **asymmetric**. As a consequence, none of the higher derivatives will be cancel out in the Taylor expansion of the truncation error.

From the above considerations, one can understand why finite differences constructed to approximate the convection term $\partial c/\partial x$ often do not reproduce the convection process correctly. For instance, due to the absence of dissipation, central differences are prone to generate wiggles in the numerical solution, in particular near sharp gradients, which can render the solution physically meaningless. This is the case when we consider, for example, the physics of a dissolved substance of which its concentration can never be negative. On the other hand, the first order upwind scheme does not display wiggles but creates too much numerical diffusion leading to an inaccurate numerical solution. The question arises whether there are schemes exhibiting a better balance between dispersion and dissipation such that spurious oscillations are diminished.

Higher order upwind schemes seem to be the answer. It is obvious that any upwind scheme is asymmetric because of its preferred direction, thus exposing both dispersion and dissipation effects. In the literature, there are many higher order upwind schemes proposed. See, for instance, Hirsch (1990) and Wesseling (2001) and the references quoted there. In this reader, we shall consider two examples. The first example is the well-known second order BDF (backward difference) upwind scheme for the space discretization of $\partial c/\partial x$, assuming $u > 0$,

$$\frac{3c_m - 4c_{m-1} + c_{m-2}}{2\Delta x}$$

Exercise 3.3.17 Show that the corresponding Taylor expansion is given by

$$\frac{\partial c}{\partial x} - \frac{1}{3}\Delta x^2 \frac{\partial^3 c}{\partial x^3} + \frac{1}{4}\Delta x^3 \frac{\partial^4 c}{\partial x^4} - \frac{7}{60}\Delta x^4 \frac{\partial^5 c}{\partial x^5} + \mathcal{O}(\Delta x^5)$$

This scheme obviously has a smaller amount of numerical dissipation compared to the first order upwind scheme. Unfortunately, it generates wiggles, though these are smaller than in the case of central differences; see Figure 3.11. Note that here the BDF scheme creates undershoot.

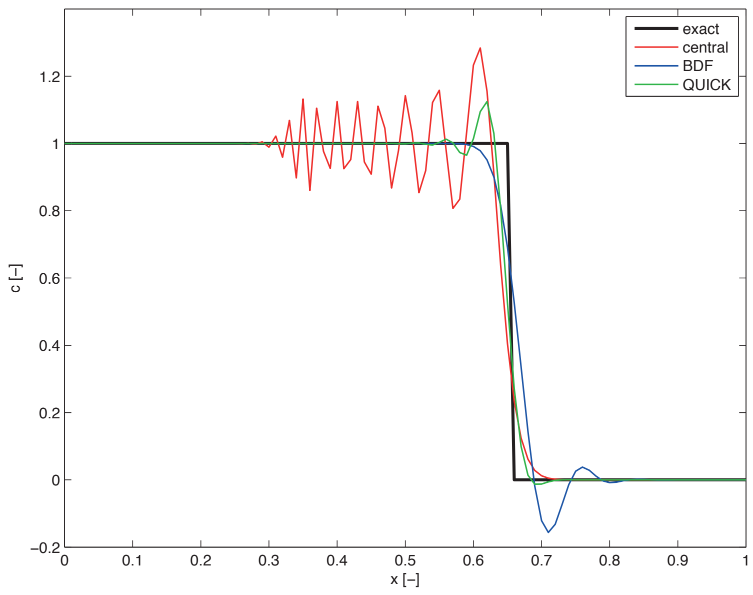


Figure 3.11: Numerical representations of a steep front: central differences, the BDF scheme and the QUICK scheme.

The second example is the well-known second order QUICK scheme as given by, assuming $u > 0$,

$$\frac{3c_{m+1} + 3c_m - 7c_{m-1} + c_{m-2}}{8\Delta x}$$

Exercise 3.3.18 Show that the corresponding Taylor expansion is given by

$$\frac{\partial c}{\partial x} + \frac{1}{24}\Delta x^2 \frac{\partial^3 c}{\partial x^3} + \frac{1}{16}\Delta x^3 \frac{\partial^4 c}{\partial x^4} + \mathcal{O}(\Delta x^4)$$

By comparison, the QUICK scheme is more accurate than the BDF scheme, though they are of the same order of accuracy. See also Figure 3.11. Nevertheless, this scheme also creates non-physical oscillations, both under- and overshoots.

When suitable mesh refinement is applied around steep gradients or discontinuities, the amplitude of the numerical oscillations is often small enough to tolerate these wiggles. Nevertheless, it is desirable to construct a scheme that does not produce these unwanted wiggles. In this context, we discuss the requirement of positiveness that we found useful for the diffusion equation. This requirement basically means that if the initial condition is non-negative, i.e. $c(x, 0) \geq 0$ for all x, then at any subsequent time t, $c(x, t) \geq 0$ is also satisfied for all x. An example is the FTBS scheme which is positive when it is stable as shown in the previous section.

However, positiveness by itself is not very useful for discretizations of the convection term. A more useful concept is that of **monotonicity**. This says that if the initial condition is a monotonically increasing function then the solution at any subsequent time is also monotonically increasing. (The same holds for a monotonically decreasing function.) A monotone scheme cannot create *new* extrema and guarantees that the numerical solution does not exhibit spurious oscillations. As a consequence, there will be no under- and overshoots. A monotone scheme *must* produce some numerical dissipation to counteract wiggles. In other words, monotone schemes are dissipative.

Exercise 3.3.19 Explain that symmetric schemes can never be monotone.

Exercise 3.3.20 Explain why dissipative schemes are not necessarily monotone.

We have seen that higher order upwind schemes generate unwanted oscillations around sharp gradients. This is somehow disappointing since it is the hope that the introduction of some dissipation in the upwind approximation will prevent the generation of wiggles in the numerical solutions. However, this is not the case. In fact, it was Godunov who introduced in 1959 the following theorem called the **Godunov's order barrier theorem**.

A monotone, linear scheme is at most first order accurate.

As a consequence, to construct a higher order, monotone scheme in order to capture steep gradients without oscillations, it must be *nonlinear* even when the PDE itself is linear.

Exercise 3.3.21 Explain why the BDF and QUICK schemes are not monotone.

A popular framework for constructing a higher order, monotone, nonlinear, upwind scheme is the **flux limiting** technique. In this context, we reformulate the convection equation in the **flux conservative form**,

$$\frac{\partial c}{\partial t} + \frac{\partial f(c)}{\partial x} = 0 \tag{3.3.7}$$

where $f(c)$ is the **conserved flux** of c. In our case, $f(c) = uc$. This equation typically express **conservation** of some quantity c. Conservation means that the time variation of the solution (mass, momentum, energy, etc.) inside a volume (region of interest, computational grid, etc.) is *only* due to the boundary fluxes. Hence, apart from these fluxes, there are no other sources of generating and dissipating mass or energy. So, the total mass or energy remains *constant* inside that volume. Let us integrate Eq. (3.3.7) along the interval $[0, L]$,

$$\int\limits_{x=0}^{L} \frac{\partial c}{\partial t}\,\mathrm{d}x + \int\limits_{x=0}^{L} \frac{\partial f(c)}{\partial x}\,\mathrm{d}x = 0 \quad \Rightarrow \quad L\frac{\partial c}{\partial t} = f(c)|_{x=0} - f(c)|_{x=L}$$

Indeed, the time variation of (space-averaged) c is only determined by the fluxes at the boundaries, $f(c(0,t))$ and $f(c(L,t))$. We may recognized this as a balance principle "*rate-of-change equals input minus output*" and it can be regarded as a **conservation law**.

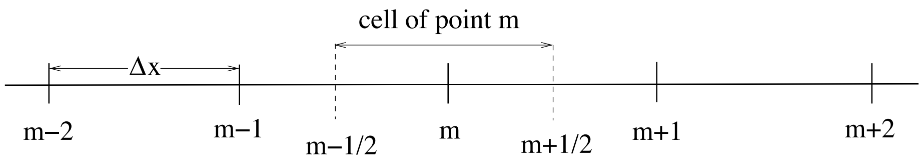


Figure 3.12: A part of a 1D computational grid.

The property of conservation should be met by any numerical scheme as well. Consider a part of a computational grid in 1D space as depicted in Figure 3.12. Within this grid we shall work with **grid cells** instead of grid points. We have **cell centers** located at x_m and **cell interfaces** at $x_{m+1/2}$. The grid cells can be regarded as volumes containing conserved quantity c_m. This quantity is assumed to be constant throughout the cell. Note that this is *just* an assumption because we have not more information about the state of the quantity in the cell than just the cell-center state, i.e. the discrete value of c_m at cell center m. This is called a **subgrid** model. It is a model of what the state looks like at spatial scales smaller than the grid spacing.

The desired semi-discretized equation must reflect the conservation property in the sense that c_m can only be reduced by moving some of it to neighbouring cells. Or its amount can be enlarged by moving something from neighbouring cells into cell m. Hence, the change in c_m can be seen as in- and outflow through the cell interfaces. This is expressed in terms of a (face) **flux** $f_{m+1/2} = f(c_{m+1/2}) \approx f(c(x_{m+1/2}, t))$ and naturally, we get the next balance principle

$$\Delta x \frac{dc_m}{dt} = f_{m-1/2} - f_{m+1/2}$$

which yields the following semi-discretized formulation

$$\frac{dc_m}{dt} + \frac{f_{m+1/2} - f_{m-1/2}}{\Delta x} = 0 \tag{3.3.8}$$

The same discretization applied at cells $m-1$ and $m+1$ is given by

$$\frac{dc_{m-1}}{dt} + \frac{f_{m-1/2} - f_{m-3/2}}{\Delta x} = 0$$

and

$$\frac{dc_{m+1}}{dt} + \frac{f_{m+3/2} - f_{m+1/2}}{\Delta x} = 0$$

respectively. Hence, the flux $f_{m+1/2}$ is an *outgoing* (or ingoing) flux for the cell of point m and an *ingoing* (or outgoing) one for the cell of point $m+1$. Also, flux $f_{m-1/2}$ is an ingoing flux of cell of point m and an outgoing one of cell of point $m-1$. Thus, a flux that is outgoing in one cell is supposed to be ingoing in the adjacent cell. If we summed up the discretized equations over all cells, the fluxes through cells will *cancel.* The resulting approximation will *only* contains the fluxes at the boundaries of the computational domain. If, for some reason, the resulting approximation still contains flux contributions from *inside* the domain, the discretization is said to be **non-conservative**. So, the sum of the discretized equations over cells $m-1$, m and $m+1$ is given by

$$\frac{d}{dt}\left(c_{m-1}+c_m+c_{m+1}\right)+\frac{f_{m+3/2}-f_{m-3/2}}{\Delta x}=0$$

where the fluxes $f_{m-1/2}$ and $f_{m+1/2}$ have been cancelled out. Multiplication with a factor $1/3$ gives

$$\frac{d}{dt}\left(\frac{c_{m-1}+c_m+c_{m+1}}{3}\right)+\frac{f_{m+3/2}-f_{m-3/2}}{3\Delta x}=0$$

which is consistent with Eq. (3.3.7) at the interval $[m-3/2, m+3/2]$, i.e. on a *coarser* mesh. Since the state at the grid center is identical to that in the whole cell, the average value of the concentration over this interval represents the state within the cell $[m-3/2, m+3/2]$ located at m. In other words, the resulted scheme and its conservation property does not only hold for one cell but also for a chain of cells.

As an example, we consider the following nonlinear 1D convection equation that typically arises in the Burger's equation, the Navier-Stokes equations and the shallow water equations

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}=0$$

with u the flow velocity. This equation is written in the **non-conservative form**. However, it can be *rewritten* in the following **conservative form**

$$\frac{\partial u}{\partial t}+\frac{\partial \frac{1}{2}u^2}{\partial x}=0$$

Next we consider a second order semi-discretized approximation of the non-conservative convection equation

$$\frac{du_m}{dt}+u_m\frac{u_{m+1}-u_{m-1}}{2\Delta x}=0$$

and a second order semi-discretized one of the conservative convection equation

$$\frac{du_m}{dt}+\frac{\frac{1}{2}u_{m+1}^2-\frac{1}{2}u_{m-1}^2}{2\Delta x}=0$$

Obviously, these conservative and non-conservative forms of the finite difference formulation of the convection term are *not* equivalent. Also, there is no way that the non-conservative approximation can be rewritten into a conservative one. Since, any conservative form of finite differences reflect the desired conservation property, it is necessary to express the PDE in conservation form.

Let us go back to our convection equation in the conservative form, Eq. (3.3.7) with $f(c)=uc$. This equation is discretized in space, as given by Eq. (3.3.8). The flux at cell interface $m+1/2$ is given by

$$f_{m+1/2}=u_{m+1/2}\,\hat{c}_{m+1/2}$$

where $\hat{c}_{m+1/2}$ is some estimate of the concentration at the interface. Here, we shall assume that the velocity u is given at the interfaces, i.e. the *cell-face* values $u_{m+1/2}$ are known. The question remains how to determine the cell-face values of the concentration. This is the central issue being addressed in the **finite volume method**. It is a matter of interpolation: the cell-face value is expressed in terms of the cell-centered values. A choice would be the arithmetic mean, as follows

$$\hat{c}_{m+1/2} = \frac{1}{2}(c_{m+1} + c_m)$$

so that

$$f_{m+1/2} = \frac{1}{2}u_{m+1/2}\,(c_{m+1} + c_m)$$

This flux approximation appears to be very similar to central differences. The difference is the space dependency of the velocity. Suppose that the velocity is constant everywhere, i.e. $u_{m+1/2} = u$ for $m = 0, \ldots, M$. Substitution in Eq. (3.3.8) yields

$$\frac{dc_m}{dt} + u\frac{c_{m+1} - c_{m-1}}{2\Delta x} = 0$$

which is exactly Eq. (3.3.2). Hence, the arithmetic mean does not provide a monotone scheme and thus, tend to produce oscillations near steep gradients. Obviously, there are many ways to estimate the cell-face value of the conserved quantity at the cell interface using the surrounding cell-centered values. Examples are

First order upwind

$$f_{m+1/2} = \begin{cases} u_{m+1/2}\,c_m\,, & \text{if } u_{m+1/2} > 0 \\ \\ u_{m+1/2}\,c_{m+1}\,, & \text{if } u_{m+1/2} < 0 \end{cases}$$

BDF

$$f_{m+1/2} = \begin{cases} \frac{1}{2}u_{m+1/2}\,(3c_m - c_{m-1})\,, & \text{if } u_{m+1/2} > 0 \\ \\ \frac{1}{2}u_{m+1/2}\,(3c_{m+1} - c_{m+2})\,, & \text{if } u_{m+1/2} < 0 \end{cases}$$

QUICK

$$f_{m+1/2} = \begin{cases} \frac{1}{8}u_{m+1/2}\,(3c_{m+1} + 6c_m - c_{m-1})\,, & \text{if } u_{m+1/2} > 0 \\ \\ \frac{1}{8}u_{m+1/2}\,(3c_m + 6c_{m+1} - c_{m+2})\,, & \text{if } u_{m+1/2} < 0 \end{cases}$$

Exercise 3.3.22 Suppose that the velocity u is constant everywhere. Compare these semi-discretized flux formulations with the previous discussed upwind schemes and explain their names.

The aforementioned flux formulations yield linear schemes. Except for the first order upwind scheme, they tend to generate wiggles near steep fronts. This can be understood in terms of dispersion, dissipation and their disbalance thereof. The advantage of the flux formulation, however, lies in the easy extension to *nonlinear* cases. This is the key issue of the flux limiting method. The basic idea of this method is to write the flux $f_{m+1/2}$ as the sum of a (diffusive) first order upwind part and an **anti-diffusive** part. So, the considered flux contains less diffusion compared to the first order upwind scheme. It seems logical then to formulate the anti-diffusive part as a higher order, nonlinear flux satisfying the monotonicity condition, since without it we would have the first order upwind scheme. To this end, we assume $u_{m+1/2} > 0$ and we write the flux-limiting scheme as follows

$$f_{m+1/2} = \underbrace{u_{m+1/2}\, c_m}_{\text{first order upwind}} + \underbrace{\frac{1}{2} u_{m+1/2}\, \Psi(r_{m+1/2})\, (c_m - c_{m-1})}_{\text{anti-diffusive part}} \tag{3.3.9}$$

where $\Psi(r)$ is the so-called **flux limiter**, which is generally a nonlinear function of the upwind **ratio of consecutive gradients** of the solution (assuming $u_{m+1/2} > 0$)

$$r_{m+1/2} = \frac{c_{m+1} - c_m}{c_m - c_{m-1}}$$

Exercise 3.3.23 Find the formula's for $r_{m+1/2}$ and $f_{m+1/2}$ for the case $u_{m+1/2} < 0$.

The aim of the flux limiter Ψ is to adaptively reduce the influence of the higher order terms near steep fronts or discontinuities so that the scheme is satisfying monotonicity. Furthermore, a negative value of r identifies an extremum in the solution. In such a case the scheme is switch to fully first order in order to avoid wiggles. For a smoothly changing solution, however, the flux limiter will not operate and the derivatives can be represented by higher order approximations without introducing wiggles.

Exercise 3.3.24 Derive the flux limiters for the first order upwind, central differences, BDF and QUICK schemes and show that they are linear.
(*Hint*: compare each of these schemes, as indicated previously, with Eq. (3.3.9).)

The flux-limiting scheme, Eq. (3.3.9), is monotone[5] if the graph of Ψ lies within the shaded area shown in Figure 3.13. Moreover, one can proof that if $\Psi(1) = 1$ then the flux-limiting scheme is second order accurate in space.

Exercise 3.3.25 Show that central differences, BDF and QUICK schemes are indeed second order accurate and not monotone by drawing their associated flux limiters in the $r - \Psi$ plane.

[5]The precise definition of monotonicity based on the concept of total variation diminishing (TVD) can be found in Hirsch (1990).

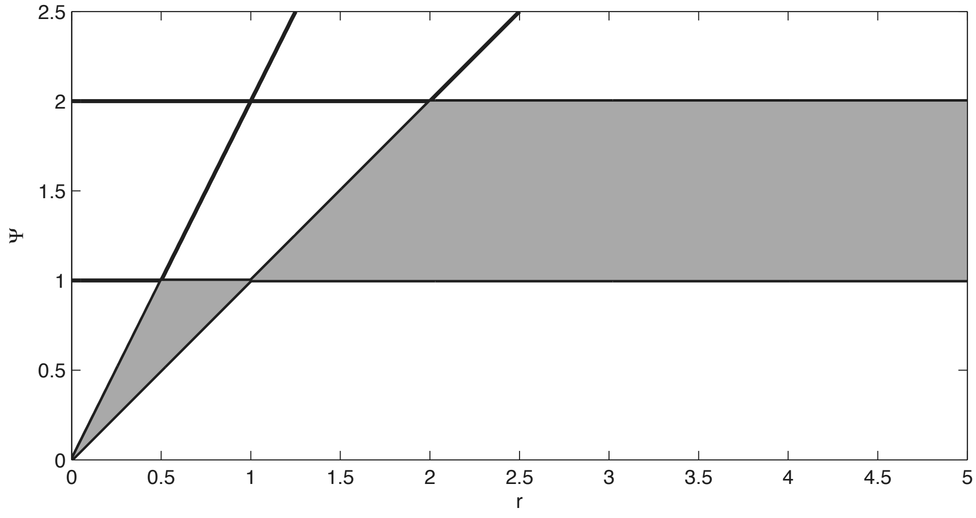


Figure 3.13: Monotonicity region for flux limiters.

Of course, an infinite number of limiters can be devised that lie in the monotonicity region. A few examples of the (nonlinear) flux limiters satisfying the monotonicity condition are

Minmod limiter

$$\Psi(r) = \max(0, \min(1, r))$$

Superbee limiter

$$\Psi(r) = \max(0, \min(2r, 1), \min(r, 2))$$

Van Leer limiter

$$\Psi(r) = \frac{r + |r|}{1 + r}$$

and they are plotted in Figure 3.14. Figure 3.15 illustrates the monotonicity obtained with the considered flux limiters for approximating the steep front. Compared to the first order upwind scheme, these flux limiters are much less dissipative. In the order of less diffusive to most diffusive (apart from first order upwinding) are the Superbee limiter, the Van Leer limiter and the Minmod limiter. This is a consequence of that fact that we have added some negative diffusion. The constraints on the limiter functions shown in Figure 3.13 basically tell us how much of it is acceptable. At the lower limit, the Minmod limiter adds just enough so that the scheme is actually second order while the Superbee limiter adds the maximum amount that still keeps the scheme from producing wiggles. Thus, the Minmod

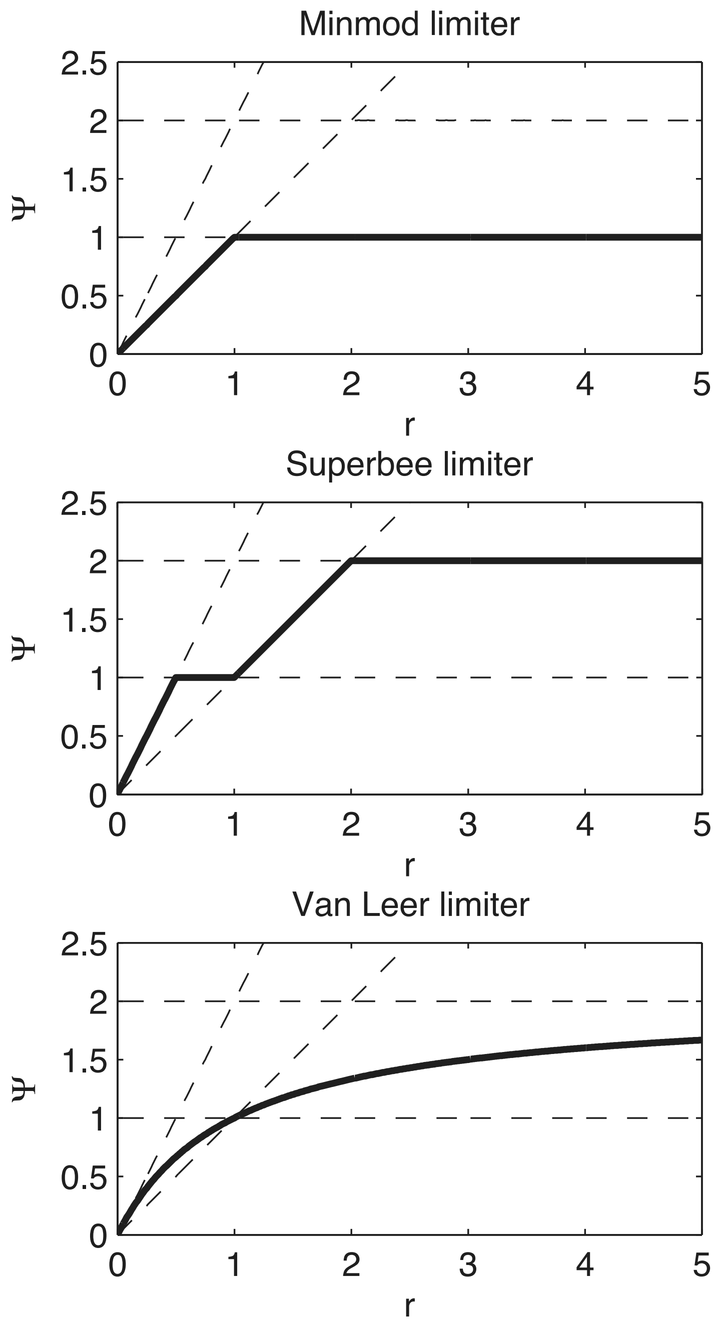


Figure 3.14: Some well-known flux limiters.

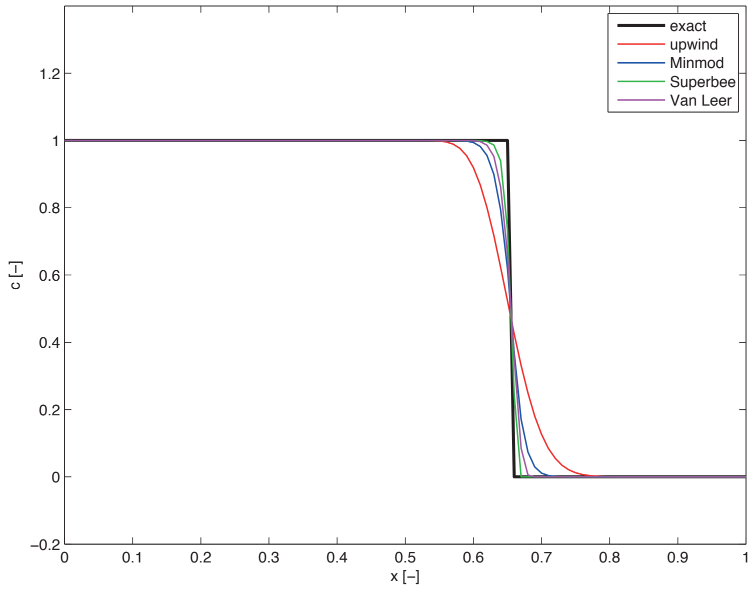


Figure 3.15: Numerical representations of a steep front: the first order upwind scheme, the Minmod limiter, the Superbee limiter and the Van Leer limiter.

limiter ends up being somewhat diffusive and smoothens out the gradients while the negative diffusion implied by the Superbee limiter has the opposite effect of sharpening even what should have been smooth profiles. Hence, it is not always wise to unconditionally choose the Superbee limiter as it might *oversteep* the shape of smooth profiles − the maxima are more flattened and the gradients are made too steep. In practice, a limiter that lies somewhere in between usually offers a good compromise for general applications.

Many other flux limiters have been proposed in the literature, see e.g. Hirsch (1990) and Wesseling (2001). For the most part, however, they do not radically affect the numerical solution. One is a little better here, another is a little better there, but on average all of the flux limiters are roughly the same. A main disadvantage is their nonlinearity, which is difficult to implement for implicit time integration like Crank-Nicolson and the $\theta-$scheme.

3.3.6 The Lax-Wendroff method

So far we only have considered the MOL approach in which the choice for time integration is independent from that of space discretization. In Section 3.3.4, it was found that any higher order finite difference scheme is unstable when used with explicit Euler time stepping.

Historically, higher order explicit schemes were formulated by introducing additional terms that compensated for the instability caused by the higher order spatial discretizations. It is possible to formulate a general procedure whereby any of the second order central or upwind schemes discussed in the previous section can be used to create stable schemes that are second order accurate in both time and space. Many schemes have been derived over the years by researchers using such ideas.

In this section we shall discuss a very popular method called the **Lax-Wendroff** method. This method is an explicit, finite difference method suited to time-marching solutions and has played a historic role. Typically, this method results in time-step dependence in the steady state solution. This occurs because the spatial approximation appears to be tied to the temporal discretization. Nowadays, the Lax-Wendroff method is generally not used and the MOL approach is usually preferred. For higher order spatial discretizations this requires the use of either a multi-step method or an implicit time integration such as the $\theta-$method. However, these resulting methods are more difficult to implement.

The derivation that is referred to as the Lax-Wendroff method for hyperbolic problems is the following. It is predicated on a Taylor expansion over one time step,

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

In this series the carrying of the second derivative, $\partial^2 c / \partial t^2$, is necessary to obtain second order accuracy in time. It is assumed that $c(x, t)$ is known from the solution at time t. If we find expressions for the time derivatives $\partial c / \partial t$ and $\partial^2 c / \partial t^2$, then the solution at the next time step, $c(x, t + \Delta t)$, can be computed explicitly. Now, use the PDE, in our case the convection equation, Eq. (3.3.1), to replace time derivatives by space derivatives,

$$\frac{\partial c}{\partial t} = -u \frac{\partial c}{\partial x}$$

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

Substituting for $\partial c / \partial t$ and $\partial^2 c / \partial t^2$ gives

$$c(x, t + \Delta t) = c(x, t) - u \Delta t \frac{\partial c}{\partial x}(x, t) + \frac{1}{2} u^2 \Delta t^2 \frac{\partial^2 c}{\partial x^2}(x, t) + \mathcal{O}\left(\Delta t^3\right)$$

If we approximate the space derivatives with central differences, we shall have a second order scheme in time and space,

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

This scheme is known as the **Lax-Wendroff** scheme for the approximation of the convection equation, Eq. (3.3.1). It is explicit and stable for $|\sigma| \leq 1$.

Exercise 3.3.26 Explore this Lax-Wendroff scheme by analyzing its order of accuracy and stability limit. Draw the stencil.

A distinctive property of the Lax-Wendroff scheme is the addition of the artificial diffusion term (compared to the FTCS scheme), though it is relatively weak. It appears to be the *least* amount of artificial dissipation allowed by the stability condition. This is can be verified by means of the modified equation approach.

Exercise 3.3.27 Verify that the total amount of numerical diffusion of the Lax-Wendroff scheme is exactly zero.

An alternative to the Lax-Wendroff scheme is the **Beam-Warming** scheme and it used to be one of the most popular methods in CFD. Instead of using second order central differences, we now employ the second order one-sided differences from the direction of upwind similar to the BDF scheme. The Beam-Warming scheme is given by ($u > 0$)

$$c_m^{n+1} = c_m^n - \frac{1}{2}\sigma\left(3c_m^n - 4c_{m-1}^n + c_{m-2}^n\right) + \frac{1}{2}\sigma^2\left(c_m^n - 2c_{m-1}^n + c_{m-2}^n\right)$$

and can be considered as an upwind version of Lax-Wendroff.

Exercise 3.3.28 Find the Beam-Warming scheme for the case $u < 0$.

Like the Lax-Wendroff scheme, the Beam-Warming scheme is second order accurate in time and space. It is also explicit, however, it is stable for $|\sigma| \leq 2$.

Exercise 3.3.29 Explore the Beam-Warming scheme by analyzing its order of accuracy and stability limit. Draw the stencil.

A main disadvantage of the Lax-Wendroff method is the algebra associated with the second derivative in time, which can be very tedious in the case of more general hyperbolic PDEs, for instance, the shallow water equations. Moreover, mixed derivatives with respect to time and space due to the differentiation of the continuity and momentum equations to obtain the second derivatives in time may be involved, which include a lot of lengthly algebra as well. The next section discuss a method that does not require the evaluation of such derivatives.

3.3.7 The MacCormack method

The **MacCormack** method is a variant of the Lax-Wendroff approach but is much simpler in its application. Like the Lax-Wendroff method, the MacCormack method is an explicit, finite difference technique which is second order accurate in both time and space. The essence of the MacCormack method is that it proceeds in two steps: a **predictor step** which is followed by a **corrector step**.

Predictor step

In this step, a *provisional* value of $c(x, t + \Delta t)$, denoted as $\overline{c}(x, t + \Delta t)$, is obtained by considering the first two terms of a Taylor series

$$\overline{c}(x, t + \Delta t) = c(x, t) + \Delta t \frac{\partial c}{\partial t}(x, t) + \mathcal{O}\left(\Delta t^2\right)$$

The expression for $\partial c/\partial t$ is provided by the PDE, while the space derivative is replaced by *forward* differences (regardless the sign of u!), as follows

$$\frac{\partial c}{\partial t}(m\Delta x, n\Delta t) = -u\frac{\partial c}{\partial x}(m\Delta x, n\Delta t) \approx -u\frac{c^n_{m+1} - c^n_m}{\Delta x}$$

Substitution yields

$$\overline{c}^{n+1}_m = c^n_m - \frac{u\Delta t}{\Delta x}\left(c^n_{m+1} - c^n_m\right)$$

The value of $\overline{c}^{n+1}_m$ is only a *predicted* value as it is first order accurate in both time and space. This will be corrected in the next step.

Corrector step

In the corrector step, we consider another intermediate step

$$\overline{\overline{c}}(x, t + \Delta t) = c(x, t) + \Delta t \overline{\frac{\partial c}{\partial t}}(x, t + \Delta t)$$

where the *predicted* value of the time derivative at time $t+\Delta t$, $\overline{\partial c/\partial t}(x, t+\Delta t)$, is calculated using the PDE again and the predicted value of c, while replacing the space derivative with *backward* differences

$$\overline{\frac{\partial c}{\partial t}}(m\Delta x, (n+1)\Delta t) \approx -u\frac{\overline{c}^{n+1}_m - \overline{c}^{n+1}_{m-1}}{\Delta x}$$

Finally, by averaging the two steps we obtain the corrected value of c at time $t + \Delta t$, as follows

$$c(x, t + \Delta t) = \frac{1}{2}\left[\overline{c}(x, t + \Delta t) + \overline{\overline{c}}(x, t + \Delta t)\right]$$

Combining all these together yields the following **MacCormack** scheme for the convection equation, Eq. (3.3.1),

$$\begin{aligned}
c^{n+1}_m &= \frac{1}{2}\left(c^n_m + \overline{c}^{n+1}_m\right) - \frac{1}{2}\sigma\left(\overline{c}^{n+1}_m - \overline{c}^{n+1}_{m-1}\right) \\[2em]
&= c^n_m - \frac{1}{2}\sigma\left(c^n_{m+1} - c^n_m + \overline{c}^{n+1}_m - \overline{c}^{n+1}_{m-1}\right) \\[2em]
&= c^n_m - \frac{1}{2}\sigma\left(c^n_{m+1} - c^n_{m-1}\right) + \frac{1}{2}\sigma^2\left(c^n_{m+1} - 2c^n_m + c^n_{m-1}\right)
\end{aligned}$$

Exercise 3.3.30 Verify this.

Apparently, the MacCormack scheme is equivalent to the Lax-Wendroff scheme. This is, however, only the case for *linear* equations.

With the MacCormack method a two-step predictor-corrector sequence is employed with forward differences on the predictor and backward differences on the corrector. The order of differencing can be reversed for the time step, i.e. backward followed by forward differences. By *alternating* the finite difference sequence in time, the method is made second order accurate in both time and space.

Interestingly enough, the MacCormack scheme can also be interpreted as a first order upwind scheme augmented with a second order correction. Indeed, the above scheme can be rewritten as

$$c_m^{n+1} = \underbrace{c_m^n - \sigma\left(c_{m+1}^n - c_m^n\right)}_{\text{first order upwind}} - \underbrace{\frac{1}{2}\sigma\left(\overline{c}_m^{n+1} - c_{m+1}^n + c_m^n - \overline{c}_{m-1}^{n+1}\right)}_{\text{second order correction}}$$

which can be applied for the case $u < 0$. When using backward differences on the predictor and forward differences on the corrector, we obtain a scheme for $u > 0$

$$c_m^{n+1} = c_m^n - \sigma\left(c_m^n - c_{m-1}^n\right) - \frac{1}{2}\sigma\left(\overline{c}_{m+1}^{n+1} - c_m^n + c_{m-1}^n - \overline{c}_m^{n+1}\right)$$

Exercise 3.3.31 Verify this.

3.3.8 Phase– and amplitude–error analysis

In this section we analyze the accuracy of some numerical schemes by considering the accuracy of propagation and the associated comparison between the exact and numerical solution.

We consider the linear convection equation, Eq. (3.3.1), and we assume periodic boundary conditions, $c(0, t) = c(L, t)$, for all time t. In effect, this problem can be considered as an initial value problem with $-\infty < x < \infty$. The exact solution is the sum of an infinite number of Fourier modes of which we consider a single one,

$$c(x, t) = \hat{c}(t)e^{\mathrm{i}kx}$$

with $\hat{c}(t)$ the amplitude of the exact solution and $k = 2\pi/L$ the wave number. Substitution into Eq. (3.3.1) gives

$$\frac{d\hat{c}}{dt} + \mathrm{i}uk\hat{c} = 0$$

which is an ODE for $\hat{c}(t)$. A solution is given by

$$\hat{c}(t) = e^{-\mathrm{i}ukt}$$

Hence, a solution to Eq. (3.3.1) is given by

$$c(x,t) = e^{\mathrm{i}k(x-ut)}$$

So, *any* Fourier mode travels with speed u. See also Appendix B for details.

Due to space discretization of $\partial c/\partial x$, it is expected that the numerical solution travels with some other speed $\tilde{u}$ differing from u depending on the *number of grid points per wave length*. This will be the central issue. We shall discuss this issue for the following semi discretizations: central differences, the Preissmann scheme and the upwind method.

Consider the following central differences of the considered convection equation

$$\frac{dc_m}{dt} + u\frac{c_{m+1} - c_{m-1}}{2\Delta x} = 0\,, \quad m = 1, \ldots, M-1\,, \quad t > 0 \tag{3.3.10}$$

Because of the periodicity in the boundary conditions, the numerical solution to Eq. (3.3.10) can be expanded as a discrete Fourier series in space for all time t, as follows

$$c_m(t) = \sum_{j=0}^{M-1} \tilde{c}_j(t) e^{im\phi_j}$$

where $\tilde{c}_j(t)$ is the amplitude of the jth harmonic, $\phi_j = k_j \Delta x \in [0, 2\pi)$ is the phase angle, and $k_j = 2\pi j/L$ is the associated wave number (see Appendix B). Since, the semi-discretized equation is linear, we consider a single harmonic

$$c_m(t) = \tilde{c}(t) e^{\mathrm{i}m\phi}$$

with $\phi = k\Delta x = 2\pi\Delta x/L$. Substitution into Eq. (3.3.10) and subsequently division of $e^{\mathrm{i}m\phi}$ yields an ODE for $\tilde{c}(t)$

$$\frac{d\tilde{c}}{dt} + \frac{u}{2\Delta x}\left(e^{\mathrm{i}\phi} - e^{-\mathrm{i}\phi}\right)\tilde{c} = 0 \quad \Rightarrow \quad \frac{d\tilde{c}}{dt} + \frac{\mathrm{i}u}{\Delta x}\sin\phi\,\tilde{c} = 0$$

Exercise 3.3.32 Verify this ODE.

A solution to this ODE is given by

$$\tilde{c}(t) = e^{-\mathrm{i}\frac{u}{\Delta x}\sin\phi t}$$

Hence,

$$c_m(t) = e^{\mathrm{i}m\phi - \frac{\mathrm{i}u}{\Delta x}\sin\phi t} \overset{\overset{\phi=k\Delta x}{\downarrow}}{=} e^{\mathrm{i}k(m\Delta x - u\frac{\sin\phi}{\phi}t)}$$

This numerical Fourier mode travels with speed

$$\tilde{u} = u\frac{\sin\phi}{\phi}$$

The **relative wave speed** is defined as $\tilde{u}/u$ and here, it is given by

$$\frac{\tilde{u}}{u} = \frac{\sin\phi}{\phi}$$

The **phase error** is given by

$$\frac{\tilde{u}}{u} - 1 = \frac{\sin\phi}{\phi} - 1$$

The number of grid points per wave length is

$$N = \frac{L}{\Delta x}$$

so that

$$\phi = k\Delta x = \frac{2\pi}{L}\Delta x = \frac{2\pi}{N}$$

Hence,

$$\Delta x \to 0 \quad \Rightarrow \quad N \to \infty \quad \Rightarrow \quad \phi \to 0$$

As a consequence,

$$\lim_{\phi\to 0} \frac{\sin\phi}{\phi} = 1 \quad \Rightarrow \quad \tilde{u} \to u$$

or

$$\lim_{N\to\infty} \tilde{u} = u$$

So, the more grid points per wave length we choose, the more accurate the numerical representation of the propagation will be.

Since, $\tilde{u}$ is a function of ϕ, each harmonic has a different speed. Generally, a shorter component propagates slower than a longer component. This phenomenon is call **dispersion**. Moreover, since

$$\frac{\sin\phi}{\phi} < 1\,, \quad \forall\phi \in [0,\pi]$$

the numerical solution propagates slower compared to the exact solution. This is called **lagging** and apparently, central differences introduces a **lagging phase error**.

Exercise 3.3.33 Explain why this is the case.

Next we consider the phase error induced by the Preissmann scheme. An associated semi discretization is given by

$$\frac{1}{2}\frac{dc_m}{dt} + \frac{1}{2}\frac{dc_{m-1}}{dt} + u\frac{c_m - c_{m-1}}{\Delta x} = 0$$

Again, we assume a single Fourier mode

$$c_m(t) = \tilde{c}(t)e^{im\phi}$$

Substitution and subsequently division of $e^{\mathrm{i}m\phi}$ gives

$$\frac{d\tilde{c}}{dt} + 2\frac{u}{\Delta x}\frac{1-e^{-\mathrm{i}\phi}}{1+e^{-\mathrm{i}\phi}}\,\tilde{c} = 0$$

or

$$\frac{d\tilde{c}}{dt} + 2\mathrm{i}\frac{u}{\Delta x}\tan\frac{1}{2}\phi\,\tilde{c} = 0$$

Exercise 3.3.34 Verify this latter implication.

A corresponding solution is given by

$$\tilde{c}(t) = e^{-2\mathrm{i}\frac{u}{\Delta x}\tan {}^1\!/\!_2\phi t}$$

Hence,

$$c_m(t) = e^{\mathrm{i}k(m\Delta x - 2u\frac{\tan {}^1\!/\!_2\phi}{\phi}t)}$$

Thus, the relative wave speed is

$$\frac{\tilde{u}}{u} = \frac{\tan\tilde{\phi}}{\tilde{\phi}}$$

with

$$\tilde{\phi} = \frac{1}{2}\phi = \frac{\pi}{N}$$

As a consequence,

$$\lim_{\substack{N\to\infty\\ \tilde{\phi}\to 0}} \frac{\tilde{u}}{u} = \frac{\sin\tilde{\phi}}{\tilde{\phi}}\frac{1}{\cos\tilde{\phi}} = 1$$

However, some Fourier modes travel with an infinite speed. These Fourier modes are $\tilde{\phi} = {}^1\!/\!_2 p\pi$, $p = 1, 2, \ldots$. The phase error of the Preissmann scheme will therefore be a leading one; the Preissmann scheme is expected to show a **leading phase error**. Figure 3.16 illustrates the behaviour of the phase error due to the space discretization. Clearly, the numerical solution obtained with the Preissmann scheme propagates faster than the exact solution, while that of central differences the opposite is the case. Apparently, by taking at least 20 grid points per wave length, the phase error is negligble small.

Finally, we consider the following upwind discretization with $u > 0$

$$\frac{dc_m}{dt} + u\frac{c_m - c_{m-1}}{\Delta x} = 0$$

and its associated ODE for $\tilde{c}$ is given by

$$\frac{d\tilde{c}}{dt} + \frac{u}{\Delta x}\left(1 - e^{-\mathrm{i}\phi}\right)\tilde{c} = 0$$

Exercise 3.3.35 Verify this ODE.

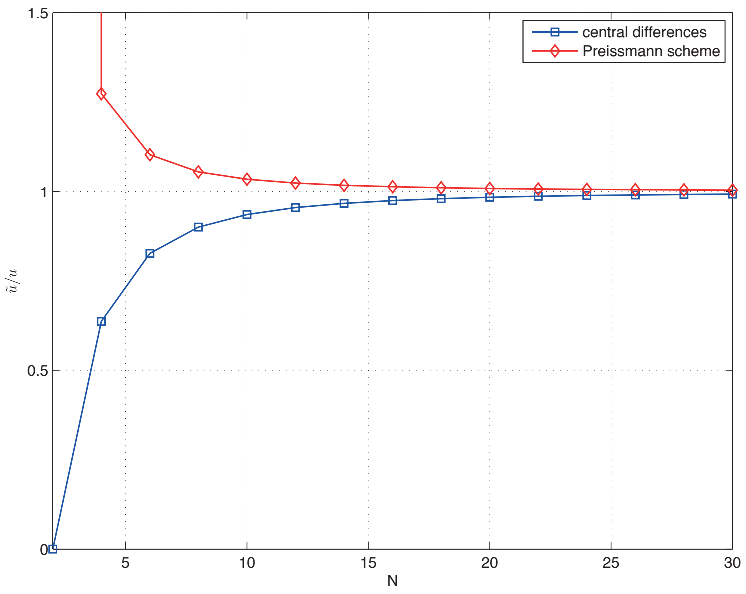


Figure 3.16: Phase error as function of the number of grid points per wave length due to the space discretization.

A corresponding solution is given by

$$\tilde{c}(t) = e^{-\frac{u}{\Delta x}(1-\cos\phi)t}\, e^{-\mathrm{i}\frac{u}{\Delta x}\sin\phi t}$$

Thus, a solution to the above semi-discretized equation is

$$c_m(t) = \overbrace{e^{-\frac{u}{\Delta x}(1-\cos\phi)t}}^{\text{real}}\ \overbrace{e^{\mathrm{i}k(m\Delta x - u\frac{\sin\phi}{\phi}t)}}^{\text{imaginary}}$$

This solution represents a travelling wave (as indicated by the *imaginary* exponent). The wave propagates with a speed $\tilde{u} = u\sin\phi/\phi$ and, just like central differences, first order upwinding induced a lagging phase error. In addition, however, the amplitude of this wave is decreasing as time evolves (as indicated by the *real* exponent). (For an explanation, see Appendix B.1. Recall that the shape of the exact solution of the considered convection equation never changes.) This is due to the numerical dissipation that is generated by the upwind method. This method is said to be **dissipative**. Any dissipative scheme displays an **amplitude error**. One can proof that symmetric schemes never have amplitude errors, while asymmetric schemes do have amplitude errors (see also Section 3.3.5).

The **amplitude factor** G is defined as the relative decay of the amplitude of the solution *after* travelling over one wave period of L/u. So, in the case of first order upwinding, this factor is given by

$$G = e^{-\frac{u}{\Delta x}(1-\cos\phi)\frac{L}{u}} = e^{N(\cos\phi - 1)}$$

Generally speaking, a longer Fourier component will be less dissipated compared to a shorter component. Furthermore, one can show that

$$\lim_{N\to\infty} G = 1$$

Exercise 3.3.36 Verify this.

Thus, by taking a larger number of grid points per wave length, the amplitude error becomes smaller.

So far we did not take into account the influence of the *time integration* on the phase error. We start with the usual ansatz

$$c_m^n = c^0 r^n e^{\mathrm{i}m\phi}$$

with c^0 an initial condition. Generally, the amplification factor r is a complex number

$$r = |r|e^{\mathrm{i}\arg(r)}$$

The numerical solution can be written as

$$c_m^n = c^0 |r|^n e^{\mathrm{i}n\arg(r)+\mathrm{i}m\phi} = c^0 |r|^n e^{\mathrm{i}k(m\Delta x+\frac{n}{k}\arg(r))}$$

The numerical speed is then

$$-\tilde{u}t = \frac{n}{k}\arg(r) \quad \Rightarrow \quad \tilde{u} = -\frac{n}{kt}\arg(r)$$

and thus, the relative speed equals (recall $n = t/\Delta t$)

$$\frac{\tilde{u}}{u} = -\frac{\arg(r)}{u\,k\Delta t}$$

In addition, we have

$$u\,k\Delta t = \frac{u\Delta t}{\Delta x}\,k\Delta x = \sigma\phi$$

So, the general formula for the relative speed due to both time integration and space discretization is given by

$$\boxed{\frac{\tilde{u}}{u} = -\frac{\arg(r)}{\sigma\phi}}$$

The amplitude factor is given by

$$G = |r|^{L/(u\Delta t)} = |r|^{2\pi/(\sigma\phi)}$$

so that quantity $\sigma\phi$ can be expressed in terms of number of time steps per wave period.

Exercise 3.3.37 Verify this amplitude factor.

As an example, we consider the FTBS scheme with $u > 0$. Its amplification factor is given by

$$r = 1 - \sigma\left(1 - e^{-\mathrm{i}\phi}\right) = 1 - \sigma(1 - \cos\phi) - \mathrm{i}\sigma\sin\phi$$

Exercise 3.3.38 Verify this.

Hence,

$$\arg(r) = \mathrm{atan}\left(\frac{-\sigma\sin\phi}{1 - \sigma(1 - \cos\phi)}\right)$$

The phase error is then given by

$$\frac{\tilde{u}}{u} - 1 = -1 - \frac{\arg(r)}{\sigma\phi} = -1 - \frac{1}{\sigma\phi}\mathrm{atan}\left(\frac{-\sigma\sin\phi}{1 - \sigma(1 - \cos\phi)}\right)$$

Exercise 3.3.39 Determine the amplitude factor of the FTBS scheme.

Let us consider the situation when the Courant number equals one, i.e. $\sigma = 1$. The phase error of the FTBS scheme becomes

$$\frac{\tilde{u}}{u} - 1 = -1 - \frac{1}{\phi}\mathrm{atan}\left(\frac{-\sin\phi}{\cos\phi}\right) = 0$$

Apparently, when $\sigma = 1$, the FTBS scheme does not show any phase error in the numerical solution. Also, there is no amplitude error, i.e. $G = 1$, if $\sigma = 1$. The numerical solution is thus exact along the characteristic $dx/dt = u = \Delta x/\Delta t$, i.e. $c_m^{n+1} = c_{m-1}^n$; see Figure 3.17. This property is called **point-to-point transfer**. An easy way to find this property is the

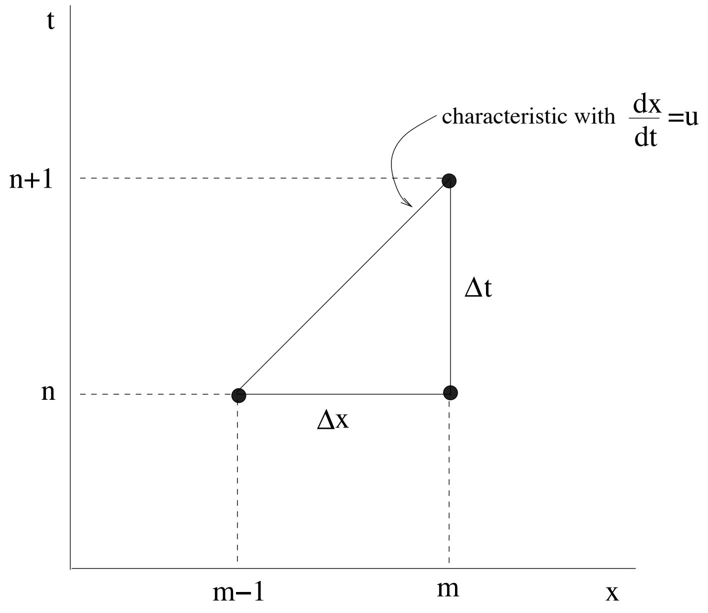


Figure 3.17: Point-to-point transfer.

following. The FTBS scheme reads (recall $u > 0$)

$$c_m^{n+1} = c_m^n - \sigma(c_m^n - c_{m-1}^n)$$

If $\sigma = 1$ then this equation reduces to $c_m^{n+1} = c_{m-1}^n$ which exactly represent point to point along the characteristic.

The behaviour of both the phase error and amplitude error for various schemes is depicted in Figure 3.18. These errors are directly related to the number of grid points per wave length. (They can also be expressed in terms of number of time steps per wave period, although it is less striking.) Also, the dependence of the Courant number on errors is clearly shown. Obviously, the FTBS scheme, the Lax-Wendroff scheme and the Preissmann scheme have a point-to-point transfer when $\sigma = 1$. Furthermore, both the FTBS scheme and Lax-Wendroff scheme have an amplitude error. Note also that the FTBS scheme needs a lot of grid points per wave length to obtain an acceptable amplitude error.

Above considerations are quite helpful in understanding how actually the numerical scheme works and what it does to the wave. Below are some exercises to do to explore some numerical properties of different schemes.

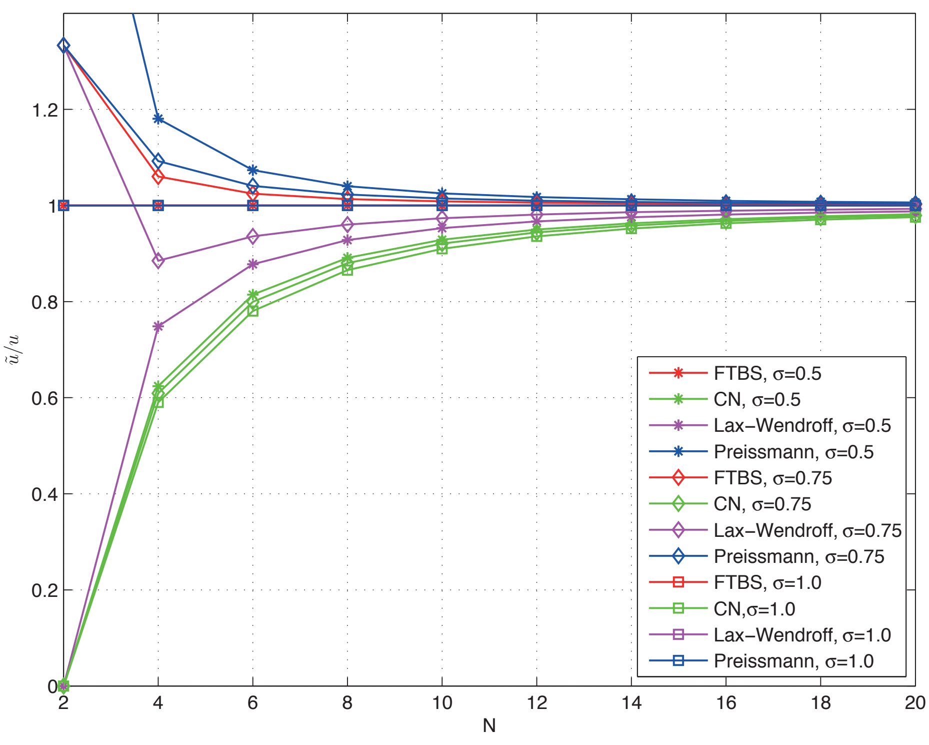
Exercise 3.3.40 Consider the scheme that combines Crank-Nicolson for time integration and central differences for space discretization. Show that this scheme does not have an amplitude error and does not have a point-to-point transfer.

Exercise 3.3.41 Consider the Lax-Wendroff scheme. Show that this scheme does both have an amplitude error and a point-to-point transfer.

Exercise 3.3.42 Consider the Beam-Warming scheme. Compute the amplitude error and the phase error. Verify whether this scheme has a point-to-point transfer.

Exercise 3.3.43 Consider the leapfrog scheme. Answer the following questions.

1. Does this scheme have a phase error? If so, give the corresponding formula.
2. Does this scheme have an amplitude error? If so, determine the amplitude factor.
3. Does this scheme have a point-to-point transfer?



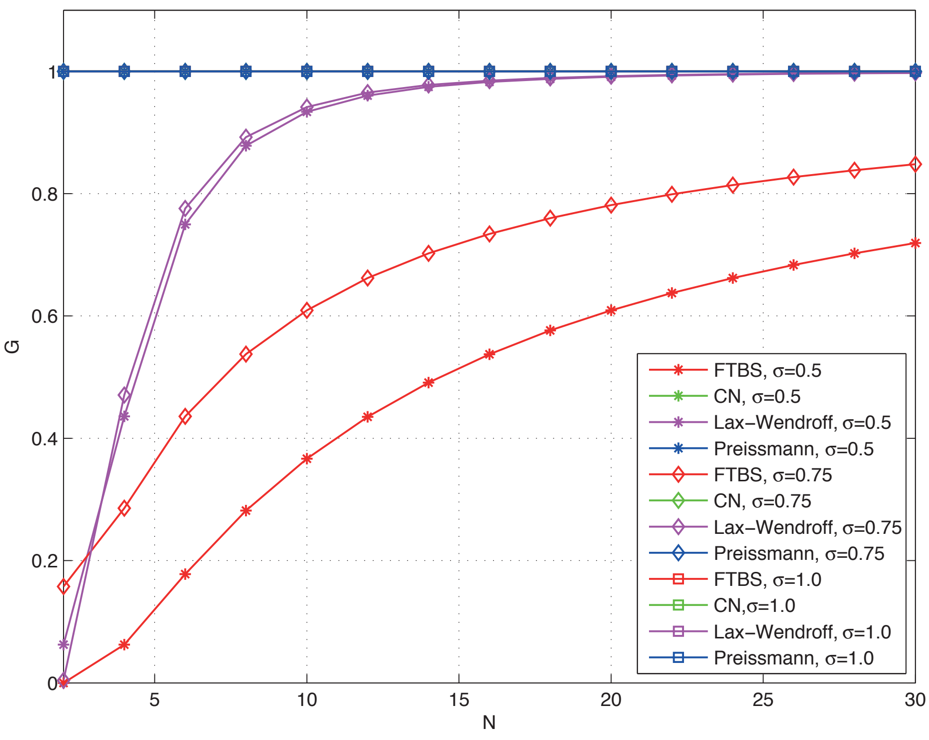


Figure 3.18: Phase error (top panel) and amplitude error (bottom panel) as function of the number of grid points per wave length for different values of Courant number and for various numerical schemes.

3.4 Convection-diffusion equation

In Sections 3.2 and 3.3 we have treated the numerical solution of the diffusion equation and the convection equation, respectively. By combining these processses we obtain the following 1D **convection-diffusion** equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} - \kappa\frac{\partial^2 c}{\partial x^2} = 0$$

This **instationary** convection-diffusion equation describes the time evolution of a **constituent** c that is transported with flow velocity u and at the same time diffuses with a diffusion coefficient κ. This equation is classified as a *parabolic* one. So, the method of characteristics is not an appropriate method to find a solution. In some cases, however, we may be interested in the steady state only. In this respect, the following **stationary** equation will be considered

$$u\frac{dc}{dx} - \kappa\frac{d^2 c}{dx^2} = 0$$

Exercise 3.4.1 To which class belongs this stationary convection-diffusion equation?

The numerical solution of the stationary convection-diffusion equation will be dealt with in Section 3.4.1, while Section 3.4.2 will consider the numerical solution of the instationary convection-diffusion equation.

3.4.1 Stationary convection-diffusion equation

The following 1D boundary value problem is considered

$$u\frac{dc}{dx} - \kappa\frac{d^2 c}{dx^2} = 0\,, \quad 0 < x < 1 \tag{3.4.1a}$$

$$c(0) = 0 \tag{3.4.1b}$$

$$c(1) = 1 \tag{3.4.1c}$$

The solution to this problem is the space distribution of the constituent c along the interval $0 \le x \le 1$ as depicted in Figure 3.19. Generally, this constituent $c(x)$ is assumed to be smooth enough. Clearly, the solution displays a steep gradient in the boundary layer. The thickness of this layer is mainly determined by the amount of diffusion involved and is due to the fact that the solution has to match to all the given boundary conditions. The boundary layer thickness is generally proportional to the square root of κ. To accurately solve this boundary layer, a local mesh refinement is often necessary.

Exercise 3.4.2 Indicate which type of boundary conditions have been imposed.

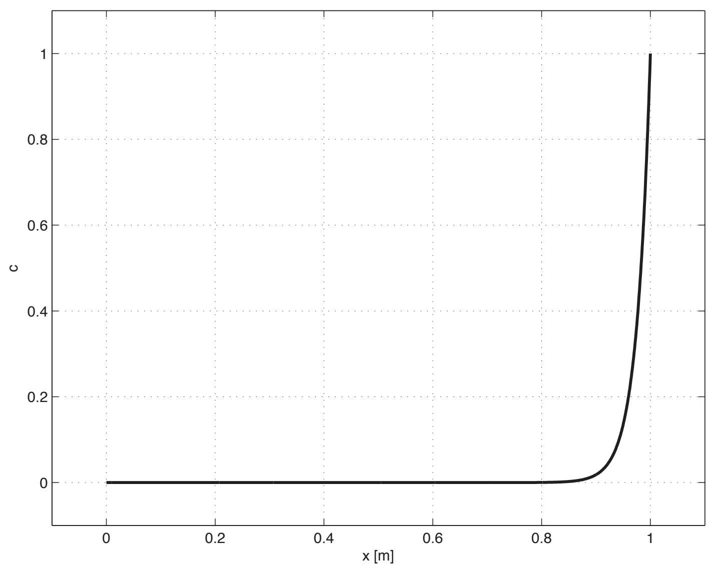


Figure 3.19: Exact solution to Eqs. (3.4.1a)−(3.4.1c) with $\kappa = 0.025$ m^2/s and $u = 1$ m/s.

Based on accuracy reasons, we choose central differences for both convective and diffusive processes, as follows

$$u\frac{c_{m+1} - c_{m-1}}{2\Delta x} - \kappa\frac{c_{m+1} - 2c_m + c_{m-1}}{\Delta x^2} = 0\,, \quad m = 1, \ldots, M-1 \tag{3.4.2}$$

Exercise 3.4.3 Verify this and show that $\tau_{\Delta x} = \mathcal{O}(\Delta x^2)$.

The boundary conditions are simply discretized as follows

$$c_0 = 0\,, \quad c_M = 1$$

Since convection is involved, the numerical solution may display wiggles or negative values, especially near steep gradients, unless there is sufficient diffusion. The question arises whether the amount of physical diffusion, as indicated by κ, is considered to be enough to prevent oscillations. We rewrite Eq. (3.4.2) as follows

$$\left(\frac{1}{2}P_\Delta - 1\right)c_{m+1} + 2c_m - \left(\frac{1}{2}P_\Delta + 1\right)c_{m-1} = 0$$

with

$$P_\Delta \equiv \frac{u\Delta x}{\kappa}$$

the so-called **mesh Péclet number**.

Exercise 3.4.4 Verify this equation.

Note that because of consistency, the sum of the coefficients is zero. Hence, with $p = {}^1\!/_2 P_\Delta - 1$ and $q = -{}^1\!/_2 P_\Delta - 1$, we have $p + q + 2 = 0$. This equation represents a recurrent relation and its general solution is of the form

$$c_m = \alpha r_1^m + \beta r_2^m$$

where r_1 and r_2 are the roots of the following characteristic equation

$$r^2 - \left(1 + \frac{q}{p}\right) r + \frac{q}{p} = 0$$

These roots are given by

$$r_1 = 1\,, \quad r_2 = \frac{q}{p} = \frac{2 + P_\Delta}{2 - P_\Delta}$$

Exercise 3.4.5 Verify the characteristic equation and its roots.

Exercise 3.4.6 How can the constants α and β be determined?

To prevent wiggles, we must require $r_2 \geq 0$. Hence, the restriction on the mesh Péclet number is

$$\boxed{|P_\Delta| = \frac{|u|\Delta x}{\kappa} \leq 2}$$

Exercise 3.4.7 Verify this.

Let us consider our example with $\kappa = 0.025$ m^2/s and $u = 1$ m/s and we choose $\Delta x = 0.1$ m, so that $P_\Delta = 4$. Figure 3.20 depicts the obtained numerical solution that clearly shows wiggles (or undershoot).

Unless κ is considerably large, the above restriction can be a severe one. A simple remedy would be to decrease the mesh width Δx such that the considered restriction is fulfilled. From an efficieny point of view, however, this remedy is not an appropriate one. Another remedy would be to apply an upwind scheme. In this way, sufficient amount of *numerical* diffusion is added so that wiggles are prevented. However, precautions should be taken so that the numerical diffusion will not dominate the physical one. This way of applying an upwind scheme is a very common practice.

To demonstrate this, we apply a first order upwind scheme. Let us assume $u > 0$. Hence, we have the following discretized equation for internal points

$$u \frac{c_m - c_{m-1}}{\Delta x} - \kappa \frac{c_{m+1} - 2c_m + c_{m-1}}{\Delta x^2} = 0 \tag{3.4.3}$$

The corresponding characteristic equation is given by

$$r^2 - (P_\Delta + 2) r + P_\Delta + 1 = 0$$

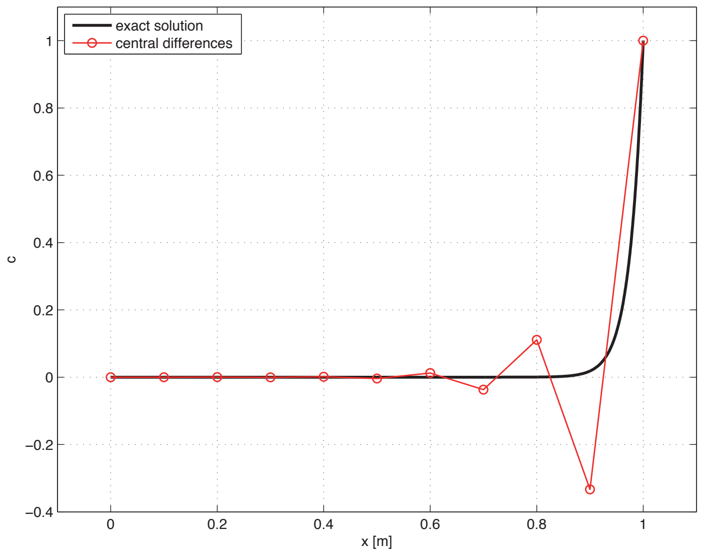


Figure 3.20: Solution obtained with central differences and $\Delta x = 0.1$ m.

and its roots are

$$r_1 = 1\,, \quad r_2 = 1 + P_\vartriangle$$

Hence, both roots are always non-negative and the numerical solution will not display any oscillations, irrespective of the value of $P_\vartriangle$. Apparently, the addition of the numerical diffusion due to first order upwinding appears to be sufficient.

Another way to see this is by considering the modified equation. This equation is obtained by adding and substracting ${}^1\!/\!{}_2 u(c_{m-1}+c_{m+1}-2c_m)/\Delta x$ to the left-hand side of Eq. (3.4.3) and subsequently re-ordering terms

$$u\frac{c_{m+1}-c_{m-1}}{2\Delta x} - (\kappa + \kappa_a)\frac{c_{m+1}-2c_m+c_{m-1}}{\Delta x^2} = 0$$

where $\kappa_a = {}^1\!/\!{}_2 u\Delta x > 0$ is the artificial or numerical diffusion coefficient due to first order upwinding. This scheme would result from the application of central differences to Eq. (3.4.1a) with a diffusion coefficient $\kappa + \kappa_a$. This scheme will generate non-negative solutions if

$$\frac{u\Delta x}{\kappa + \kappa_a} \leq 2 \quad \Rightarrow \quad u\Delta x \leq 2\kappa + u\Delta x$$

This is true for any Δx and thus, the application of first order upwinding will prevent wiggles.

However, this first order upwind scheme is often considered to be too dissipative. See Figure 3.21 where the corresponding numerical solution is found to be rather inaccurate, in particular when the physical diffusion is smaller than the numerical one, $\kappa < \kappa_a$. A higher order upwind scheme would be more appropriate.

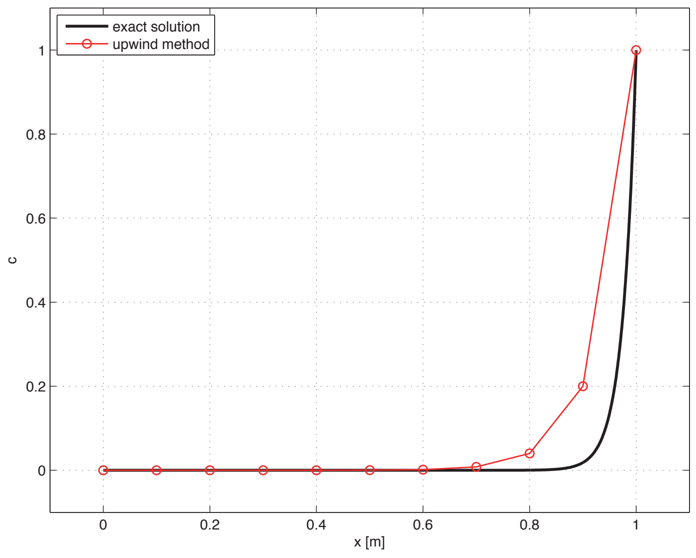


Figure 3.21: Solution obtained with the first order upwind scheme and $\Delta x = 0.1$ m so that $\kappa_a = 0.05$ m^2/s.

Exercise 3.4.8 Explain why the boundary layer thickness of the numerical solution is larger than that of the exact one.

Exercise 3.4.9 Explain that a higher order upwind scheme must be supplemented with a flux limiter in order to obtain non-negative solutions.

3.4.2 Instationary convection-diffusion equation

We consider the following 1D (initial) boundary value problem

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} - \kappa\frac{\partial^2 c}{\partial x^2} = 0\,, \quad 0 < x < 1\,, \quad t > 0 \tag{3.4.4a}$$

$$c(x,0) = 0\,, \quad 0 \le x \le 1 \tag{3.4.4b}$$

$$c(0,t) = 0\,, \quad t > 0 \tag{3.4.4c}$$

$$c(1,t) = 1\,, \quad t > 0 \tag{3.4.4d}$$

Again, we assume that the constituent $c(x,t)$ is sufficiently smooth. The solution to this problem is called **transient** and in the limit, $t \to \infty$, a steady state solution may be obtained. This is the case when the boundary conditions are time independent. This is also the solution to the stationary problem, Eqs. (3.4.1a)−(3.4.1c).

As usual, we employ the MOL approach. Again, we may choose central differences. The semi-discretized equation reads

$$\frac{dc_m}{dt} + u\frac{c_{m+1} - c_{m-1}}{2\Delta x} - \kappa\frac{c_{m+1} - 2c_m + c_{m-1}}{\Delta x^2} = 0\,, \quad m = 1, \ldots, M-1\,, \quad t > 0$$

This system of ODEs is considered to be a damped one. A commonly time integration method would be the explicit Euler method. Thus, we obtain the following discretized equation for the instationary convection-diffusion equation, Eq. (3.4.4a),

$$\frac{c_m^{n+1} - c_m^n}{\Delta t} + u\frac{c_{m+1}^n - c_{m-1}^n}{2\Delta x} - \kappa\frac{c_{m+1}^n - 2c_m^n + c_{m-1}^n}{\Delta x^2} = 0\,, \quad m = 1, \ldots, M-1\,, \quad n = 0, 1, 2, \ldots$$

This scheme is explicit and thus conditionally stable. By requiring non-negative solutions we may find some stability conditions. We rewrite the discretized equation as

$$c_m^{n+1} = (q - \frac{1}{2}\sigma)c_{m+1}^n + (1 - 2q)c_m^n + (q + \frac{1}{2}\sigma)c_{m-1}^n$$

and by induction, we assume that $c_m^n \geq 0$ for $m = 0, \ldots, M$. To have $c_m^{n+1} \geq 0$, the following conditions must be met

$$\boxed{q \leq \frac{1}{2}\,, \quad |P_\Delta| \leq 2}$$

Exercise 3.4.10 Verify these conditions.

These conditions are, however, sufficient for stability. By means of the Von Neumann method we shall found exactly the same conditions as the necessary ones for stability.

Exercise 3.4.11 Verify this.

The first condition is recognized as the stability condition for the heat equation while the second one is the condition that we have found for the stationary convection-diffusion equation.

Instead of explicit Euler we may choose the implicit $\theta-$scheme and regarding the convection term, we may choose a higher order upwind scheme instead of central differences.

Exercise 3.4.12 Let us consider the convection-diffusion equation, Eq. (3.4.4a). We apply the Crank-Nicolson scheme to integrate in time, the BDF scheme for the convection term and central differences for the diffusion term. Write down the resulted scheme, draw its stencil and show its unconditional stability. What is the order of consistency of this scheme? Is this scheme monotone?

3.5 Summary and concluding remarks

In this chapter we have discussed many topics with respect to the numerical approximation of a partial differential equation (PDE) in an introductory manner. We have restricted ourselves to the prototypes of the hyperbolic (wave-like propagation) and parabolic (diffusive transport) equations (cf. Eqs. (3.2.1a) and (3.3.1)). In this section we will summarize these topics and we will also outline some other subjects that we have not dealt with. The considered topics in this chapter describe the numerical process which essentially consists of two steps: 1) the construction step, and 2) the analysis step.

The construction step

The usually method is the method of lines which means that first the space derivatives of the PDE are discretized based on a finite difference method. The space derivatives are replaced by finite differences or algebraic expressions on the computational grid based upon the Taylor series expansion. It is important to note that it is assumed that the solution is continuously differentiable and that the Taylor series is a convergent one. A solution of the finite difference scheme will introduce numerical errors. These errors can be identified as discretisation errors and round-off errors.

For a given PDE there are a number of different ways in which its finite difference representations can be expressed. These included numerical schemes which solve forward, backward or centrally in space. The order of accuracy of a finite difference scheme describes the order of the leading term in the Taylor series, which represents the error obtained by substituting the exact solution into the finite difference formula. This error is called the truncation error of the discretization. The order of accuracy indicates the scaling of the truncation error with grid refinement. Increased order of accuracy is not necessarily desirable because it also increases the size of the computational stencil and makes the solution more difficult. (Exceptions are the box-like schemes, e.g. the Preissmann scheme.) For instance, it requires more careful treatment on the boundaries, in particular, when boundary conditions are not available. Moreover, increased accuracy of the discretization need not produce a more accurate solution unless the grid is sufficiently fine. Finally, the non-uniformity, non-orthogonality and skewness of the computational grid also may affect the discretization error. Hence, it is difficult to determine a priori the discretization error in the solution. Systematic grid refinement studies need to be carried out in order to determine the overall accuracy of the numerical scheme.

Finite difference methods are certainly not the only ones. Other well-known methods for space discretization are the finite volume method and the finite element method. Both these methods can deal with flexible meshes and are mainly useful for complicated domains. Starting point in these methods is the weak formulation, or the integral form, of the underlying PDE. This makes sense since many PDEs in physics are derived from conservation laws. In addition, the weak formulation can handle discontinuities in the solution. Rather than pointwise approximations on a grid, the finite volume method approximates the average integral value on a so-called control volume. Flux interpolations and quadrature rules (e.g. the midpoint rule) are the key techniques to carry out the finite volume approximations. Within the context of the finite element method, the PDEs are integrated over an element after having been multiplied by a so-called weight function. The unknowns are represented on the element by a shape function, which has the same form as the weight function. Many types of elements can be chosen, like triangular (2D) and tetrahedral (3D) elements. Finite volume methods are always dealing with face fluxes of which the physical properties are very clear, while finite element methods have strong mathematical foundation but less physical significance. Details on these methods may be found in Hirsch (1990) and Wesseling (2001).

Due to space discretization the PDE is replaced by a system of ODEs. This system of ODEs is integrated in time by means of a linear multistep method. (This is the final step of the method of lines.) In addition, we have time marching explicit solutions, where the unknown value can be calculated directly from known values without solving a system of equations, and implicit solutions, where the unknown is represented in terms of other neighbouring unknowns. Explicit schemes have, in general, more restrictive time step contraints than implicit schemes.

For accuracy time steps must be small compared to relevant time scales. Explicit schemes are fine in that case, since they are usually cheaper (require less computing time) per time step than implicit schemes. On the other hand, for steady state problems, implicit schemes are a better choice, since they allow larger time steps and thus more efficient.

Dealing with a stiff problem means many diverse time scales are involved. Very small time scales are due to the transient solution, while the steady state solution or the solution influenced by boundary conditions is usually represented by the larger time scales. Generally, the solution of problems that are stiff are completely impractical with methods intended for non-stiff problems, such as explicit schemes. Because, when applying an explicit scheme, the time step is limited more severely by the stability than by the accuracy of the scheme. As a consequence, implicit schemes, or even better, stiffly-stable schemes are the most appropriate ones; see e.g. Gear (1971). Examples are the first order implicit Euler scheme and the second order BDF scheme.

Alternatives to the linear multistep method are the Runge-Kutta methods and the fractional step methods (e.g. alternating direction implicit (ADI) schemes and operator splitting schemes). Runge-Kutta methods are nonlinear, one-step, multistage time integration formulas that are easier to implement, since starting values are not required, but more difficult to analyze. Fractional step methods are mainly used for advection-diffusion type equations which contain stiff source terms (e.g. turbulence model equations, reaction-diffusion-advection equations to model concentration of chemical species). In this case different time stepping techniques are applied to different parts of the equations. Stiff source terms call for a stiffly-stable scheme, while advection terms may be integrated in time explicitly. Details on these methods can be found in textbooks and on the internet.

For advection-dominated flow problems the Lax-Wendroff or MacCormack methods might be good alternatives to the method of lines. Such methods can be derived by considering the Δt^2 terms in the Taylor expansion of the time derivative and make some subsequent approximations using the underlying PDEs. Typically, these methods take into account the mathematical and physical properties of the PDEs while the method of lines is using only a minimum amount of information of the equations; see e.g. Hirsch (1990).

The analysis step

In order to obtain a correct numerical solution, it is necessary to analyze the numerical properties associated with the finite difference scheme. This includes consistency and stability; both these properties generally imply convergence. The finite difference equation must be consistent with the PDE so that both equations describe the same physical phenomena. The finite difference scheme must be stable so that the cumulative effects of all the numerical errors at any stage of the simulation are negligible, and that the computed solution only differs marginally from the exact solution of the finite difference equation. The convergence condition relates the computed solution of the discretized equation to the exact solution of the PDE. These numerical concepts can be formally presented below.

Consistency, stability and convergence

A well posed, linear, one dimensional PDE is given by

$$L\,(\,c(x,t)\,) = 0$$

with $c(x,t)$ the exact solution at location x and time t, and L is the linear (differential) operator representing the PDE. Well posedness implies that this operator is bounded, i.e. $\|L\| \leq K$ with K a constant being independent of t, and $\|\cdot\|$ a suitable norm, which prevents the solution from being unstable. (This statement stems from functional analysis; see also Richtmyer and Morton (1967, p. 41)). The considered PDE is approximated by means of a finite difference scheme given by

$$L_d\,(\,c_m^n\,) = 0$$

where c_m^n is the numerical solution at location $m\Delta x$ and time $n\Delta t$ with Δx and Δt the mesh width and time step, respectively, and L_d is the linear (difference) operator symbolizing the numerical scheme. This operator is generally a function of Δx and Δt. Just like the differential operator, the difference operator is uniformly bounded in Δx and Δt, i.e. $\|L_d\| \leq K$ with K a constant, independent of Δt (and thereby independent of Δx). The implication of this boundedness is that the numerical solution remains stable. In fact, the stability condition establishes a relation between the computed solution, afflicted with a round-off error, and the exact solution of the discretized equation.

If we replace the numerical solution by the exact one, we get the following

$$\tau_m^n = L_d\,(\,c(m\Delta x, n\Delta t)\,) = L_d\,(\,c(x_m, t_n)\,)$$

with τ_m^n the truncation error. Now, we can present this equation in two ways. One way is

$$\tau_m^n = L_d\,(\,c(x_m,t_n)\,) - L\,(\,c(x_m,t_n)\,) = (L_d - L)\,(\,c(x_m,t_n)\,)$$

This equation tells us about consistency which defines a relation between the differential equation and its discrete formulation. In fact, in the limit $\Delta x \to 0$ and $\Delta t \to 0$, the difference operator L_d approaches the differential operator L, implying the truncation error vanishes, i.e. $\tau_m^n \to 0$.

Another way is

$$\tau_m^n = L_d\,(\,c(x_m,t_n)\,) - L_d\,(\,c_m^n\,) = L_d\,(\,c(x_m,t_n) - c_m^n\,) = L_d\,(\,e_m^n\,)$$

with $e_m^n = c(x_m,t_n) - c_m^n$ the (global) error in the solution at location x_m and time t_n. This implies the following equation

$$e_m^n = L_d^{-1}\,(\,\tau_m^n\,)$$

with L_d^{-1} the inverse operator of L_d. This equation tells us about convergence which connects the computed solution of the discretized equation to the exact solution of the PDE. If there is consistency between the scheme and the PDE, then in the limit $\Delta x \to 0$ and $\Delta t \to 0$, the error in the numerical solution vanishes if and only if the operator L_d^{-1} is uniformly bounded in Δx and Δt. This makes sense because if this operator is not bounded, then there is no guarantee that during the limit process, i.e. when time step and mesh width tend to zero, the global error will vanish, even when $\tau_m^n \to 0$.

Clearly, the concepts of consistency, stability and convergence are inter-related. This is expressed by the Lax equivalence theorem (Richtmyer and Morton, 1967, p. 45) which states that for a well posed, linear PDE with a consistent discretization, stability is the necessary and sufficient condition for convergence of the numerical scheme. The key importance of this fundamental theorem is that while convergence is hard to proof, both consistency and stability are easy to verify. For nonlinear problems, however, there is no such general theorem. Here, local stability analysis may be performed by linearizing the PDE in a small solution domain. The resulted stability condition is then merely a necessary one for convergence.

Stability analysis

Various methods have been developed for the analysis of stability, nearly all of them are limited to linear problems. For a linear problem with constant coefficients, the problem of stability is now well understood when the influence of boundaries can be neglected or removed. This is the case either for an infinite domain or for periodic boundary conditions on a finite domain. For such problems, the Von Neumann method is the classical one and has been recognized as a formal stability analysis procedure, in spite of its limitations.

The effect of (non-periodic) boundary conditions, such as Dirichlet and Neumann ones, both of which are practically used in many cases, on stability can be investigated systematically using the so-called matrix method. The basic idea is to express the stability condition in terms of the eigenvalues of the amplification matrix or its matrix norm. This amplification matrix is a result of the discretization of the PDE including the boundary conditions. Incorporating different boundary conditions may imply different eigenvalues, and the boundary conditions may therefore influence the stability. However, this is rather a cumbersome and tedious method; for details, see Richtmyer and Morton (1967).

In this chapter we have considered only linear equations. However, the governing equations describing fluid motion, water movement and transport of dissolved substances are generally nonlinear. This non-linearity may arise from genuine non-linearity as in the momentum advection term $u\partial u/\partial x$, like in the shallow water equations (see Chapter 4), or from non-constant coefficients, such as the spatial variation of the thermal diffusivity or dispersion coefficient in the diffusion term, like

$$\frac{\partial}{\partial x}\left(\kappa(x)\frac{\partial c}{\partial x}\right)$$

In these cases, the Von Neumann stability method cannot be applied as the principle of superposition is no longer valid and therefore, a single wave component cannot be analyzed in isolation. Another typical feature of nonlinear problems is that stability may depend on the amplitude of the discretization error, perturbation or disturbance. An nonlinear equation may be stable for small errors but may become unstable for finite amplitudes. Also, the interaction of different Fourier modes can give rise to high frequency oscillations. Sometimes these high frequency components need extra numerical damping in the high frequency range in order to maintain stability.

For linear problems with non-constant coefficients, it can be shown that a local Von Neumann stability analysis will provide a necessary condition for stability (Richtmyer and Morton, 1967, p. 91). The nonlinear term needs to be linearized about a point to obtain a contant coefficient in the underlying PDE. Several ways can be employed to perform this linearization. The simplest and the most often used is lagging the coefficients so that these coefficients are evaluated at the previous time step. Another method is a Newton-Raphson type of procedure to linearize the nonlinear term about a known point. For details, see Appendix A.3.

Other ways to analyze stability properties are the CFL condition and the heuristic approach. The CFL condition provides a necessary condition for the convergence of a finite difference scheme to a hyperbolic equation. This CFL condition is related to the requirement of the analytical domain of dependence being contained in the numerical domain of dependence. Heuristic stability methods often have no theoretical basis to proof the correctness of the obtained results. Well-known examples are the modified equation approach and the positivity requirement. These methods typically yield a sufficient stability condition.

Further properties of numerical schemes

In addition to consistency, stability and convergence, a number of other properties is useful, but not strictly necessary, in an overall evaluation of the numerical scheme. The conservative property of discretization has been discussed in Section 3.3.5. It is shown that if a consistent evaluation of the fluxes is made then it is possible to ensure that the total flux leaving a cell through a particular face is equal to the total flux entering through the same face into the adjacent cell which share the face. This would enable the conservation law to be upheld not only at the differential level but also at discrete level. An example is mass conservation in the modelling of water quality. If a non-conserving approximation is employed, then there may be unphysical changes of concentrations.

Monotonicity is another property which should be incorporated into numerical schemes if the physical problem is asked for. By monotonicity of a solution, one means that this solution remains within bounds imposed by the physics. In addition, the minimum bound is non-decreasing and the maximum bound is non-increasing. In many cases both initial and boundary conditions expose this property. As a consequence, no new local extrema can be created within the computational domain as time proceeds, unless there are sources and sinks in the governing equations. For example, some variables such as mass, concentration, temperature and (turbulent or wave) energy are always non-negative and the computed solution should not become negative. Some of these variables such as mass fraction and porosity also have a physical upper limit of unity.

Lack of monotonicity may result in poor convergence and even failure of the numerical scheme (e.g. square root of a negative quantity). So, monotone schemes are attractive because they do not produce non-physical solutions. Usually dissipative schemes such as first order upwind ones for the advective term lead to a monotone though inaccurate solution, while higher order linear schemes tend to give rise to oscillations where discontinuities or shocks arise. This is in line with the Godunov's order barrier theorem which proves that monotone, linear schemes are, at most, first order accurate. Flux-limiting schemes are the typical higher order schemes appropriate for monotonicity preservation (Hirsch, 1990). However, these schemes are nonlinear and are therefore more difficult to implement (especially when implicit time marching is involved).

When propagation of (tidal) waves or dissolved substances is important, then propagation errors / phase errors / dispersion and damping errors / amplitude errors / dissipation must be analyzed as well. Generally, these errors are a function of the number of grid points per wave length and the number of time steps per wave period. For many numerical schemes graphs and/or expressions are available for these relations (see this lecture notes).

Chapter 4

Numerical treatment of shallow water equations

4.1 The shallow water equations

The shallow water equations describe the flow below a free surface in a fluid. These equations are derived from integrating the **Navier-Stokes equations** over the depth. The Navier-Stokes equations, in turn, are derived from mass and momentum conservation, and consist of the **continuity equation** and the **momentum equations**. We make the general assumption that flow is essentially depth averaged, observing that ocean and coastal seas, estuaries, lakes, rivers and channels are much longer than they are deep. Under this **shallowness** condition, conservation of mass implies that the vertical velocity of the fluid is small. Vertically integrating allows the vertical velocity to be removed from the equations. Furthermore, it can be shown from the vertical momentum equation that the vertical acceleration is also small, implying that the vertical pressure gradient is **hydrostatic** so that pressure is proportional to the depth of water above any point.

We restrict ourselves to a one dimensional basin (or waterway) with length L in the $x-$space. Figure 4.1 depicts a part of this basin. The most important dependent quantities that we need to calculate are $u(x,t)$, the depth-averaged **flow velocity**, and $\zeta(x,t)$, the elevation of the free surface, or the **water level**. They are governed by the continuity equation

$$\frac{\partial \zeta}{\partial t} + \frac{\partial hu}{\partial x} = 0$$

with $h(x,t) = \zeta(x,t) + d(x)$ the water depth, $d(x)$ the bottom level, and the momentum equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial \zeta}{\partial x} + c_f\frac{u|u|}{h} = 0$$

with c_f the dimensionless friction coefficient due to bottom roughness, and $g = 9.81$ ms^{-2} the gravitational acceleration. Note that the water level ζ and the bottom level d are

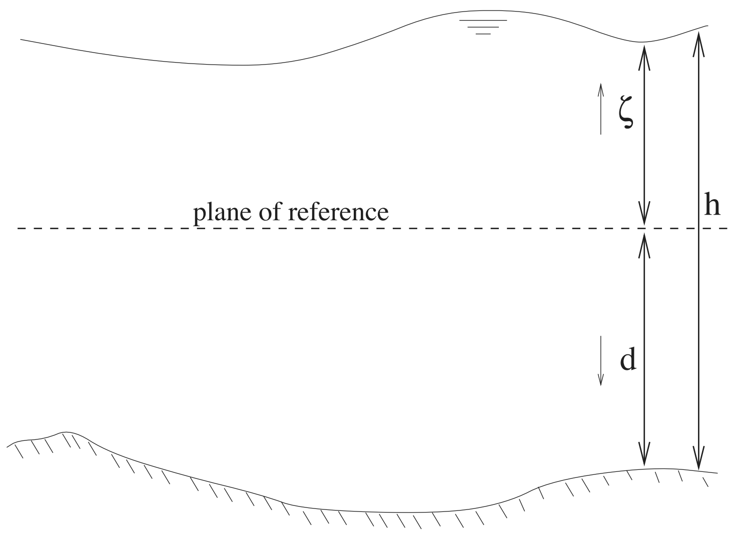


Figure 4.1: A section of basin with free surface and bottom (not to scale).

measured positively upwards and positively downwards, respectively, with respect to a **plane of reference**.

Exercise 4.1.1 Verify that c_f is indeed dimensionless.

The terms in the momentum equation are

- $\partial u/\partial t$ – comes from the rate of change of momentum in the basin with time
- $u\partial u/\partial x$ – comes from the variation of momentum flux due to fluid velocity along the basin
- $g\partial\zeta/\partial x$ – comes from the pressure forces in the water
- $c_f u|u|/h$ – comes from the resistance to motion in the basin as being caused by a bottom shear force due to bottom roughness. This term is not fundamentally exact and must be accomplished with some empirical formulas.

Writing the equations again, we have the set of PDEs

$$\frac{\partial\zeta}{\partial t} + \frac{\partial hu}{\partial x} = 0 \tag{4.1.1a}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial\zeta}{\partial x} + c_f\frac{u|u|}{h} = 0 \tag{4.1.1b}$$

These are the **shallow water equations**, sometimes called the **Saint-Venant equations** or **long wave equations**. They are the basis for most applications in hydraulics. In order for these equations to be valid, the typical wave length has to be much larger than the typical depth of the basin. As such, we can only deal with long waves like **tidal waves**, **flood waves**, **seiches**, **tsunami waves**, etc. but not wind waves.

The Eqs. (4.1.1a)−(4.1.1b) are a *coupled* system. Using the similarity transformation an uncoupled system can be derived with which an in-depth analysis can be carried out. For instance, it can be shown that the system is **hyperbolic** implying that solutions are of a wave-like nature. A more common interpretation is to write the system in terms of **characteristics**. Here we present a derivation as it is given in many textbooks. The idea is to bring the Eqs. (4.1.1a)−(4.1.1b) into a convection equation like Eq. (3.3.1) by taking a proper linear combination of these equations. In this respect, the water depth h will be considered as an unknown instead of the water level ζ. Since the bottom level $d(x)$ is time independent we can formulate the continuity equation as follows

$$\frac{\partial h}{\partial t} + \frac{\partial hu}{\partial x} = \frac{\partial h}{\partial t} + h\frac{\partial u}{\partial x} + u\frac{\partial h}{\partial x} = 0$$

while the momentum equation is rewritten as

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial h}{\partial x} - g\frac{\partial d}{\partial x} + c_f\frac{u|u|}{h} = 0$$

To obtain a linear combination we multiply the continuity equation with a factor α and add the momentum equation to it

$$\frac{\partial u}{\partial t} + (u+\alpha h)\frac{\partial u}{\partial x} + \alpha\frac{\partial h}{\partial t} + (g+\alpha u)\frac{\partial h}{\partial x} - g\frac{\partial d}{\partial x} + c_f\frac{u|u|}{h} = 0$$

We want to write the latter equation as a convection equation with an inhomogeneous part

$$\frac{\partial R}{\partial t} + c\frac{\partial R}{\partial x} = g\frac{\partial d}{\partial x} - c_f\frac{u|u|}{h}$$

with R known as the **Riemann invariant** and c the propagation speed. The inhomogeneous part is due to bottom slope $\partial d/\partial x$ and resistance. The Riemann invariant R appears to be a combination of h and u and remains constant along a characteristic $x_c(t)$ with speed $dx_c/dt = c$. To get the Riemann invariant and its speed we need to solve the following relations

$$c\alpha = g + \alpha u\,, \quad c = u + \alpha h$$

Exercise 4.1.2 Verify this.
(*Hint*: both h and u must be differentiated in the same direction in the $t-x$ plane.)

Elimination of α results in a quadratic equation in c

$$(c-u)^2 = gh$$

Apparently, there are two characteristics with the corresponding propagation speeds

$$c^{\pm} = u \pm \sqrt{gh}$$

Generally, the characteristics are *curved* since the inhomogeneous part is involved. One may recognize the quantity $\sqrt{gh}$ as the **long wave speed**. There are two contributions of $\pm\sqrt{gh}$, corresponding to both right-travelling and left-travelling waves. Their respective Riemann invariants are given by

$$R^+ = u + 2\sqrt{gh}\,, \quad R^- = u - 2\sqrt{gh}$$

Exercise 4.1.3 Verify this.

Assume constant bottom level and neglect bottom friction, we then find the simplified characteristic equations for the Riemann invariants

$$\frac{\partial R^+}{\partial t} + (u + \sqrt{gh})\frac{\partial R^+}{\partial x} = 0$$

$$\frac{\partial R^-}{\partial t} + (u - \sqrt{gh})\frac{\partial R^-}{\partial x} = 0$$

and so $R^\pm$=constant along the characteristic travelling with a speed $dx/dt = u \pm \sqrt{gh}$. Waves propagating at the speed $\pm\sqrt{gh}$ relative to the flowing water are caused by gravity only. Assume a right-travelling wave of the form (the conclusions to be drawn hold for the left-travelling wave as well)

$$R^+ = \tilde{R}e^{\mathrm{i}(\omega t - kx)}$$

with $\tilde{R} \neq 0$ a constant amplitude, ω the angular frequency and k the wave number. This is a solution of the corresponding characteristic equation if

$$(\mathrm{i}\omega - \mathrm{i}ck)\,\tilde{R} = 0$$

or

$$\omega = ck$$

Exercise 4.1.4 Verify this.

Generally, the angular frequency is a function of the wave number, i.e. $\omega = \omega(k)$. This is called the **dispersion relation**. Next we compute the **phase velocity** as defined by

$$c_{\mathrm{p}} = \frac{\omega}{k}$$

which gives

$$c_{\mathrm{p}} = c = u + \sqrt{gh}$$

Apparently, the phase velocity does not depend on the wave number k. Gravity waves are therefore called **nondispersive**.

For subcritical flow ($\mathrm{Fr} = |u|/\sqrt{gh} < 1$) with Fr the **Froude number**, the two propagation speeds, $u+\sqrt{gh}$ and $u-\sqrt{gh}$, have different signs. Hence, two characteristics will meet at

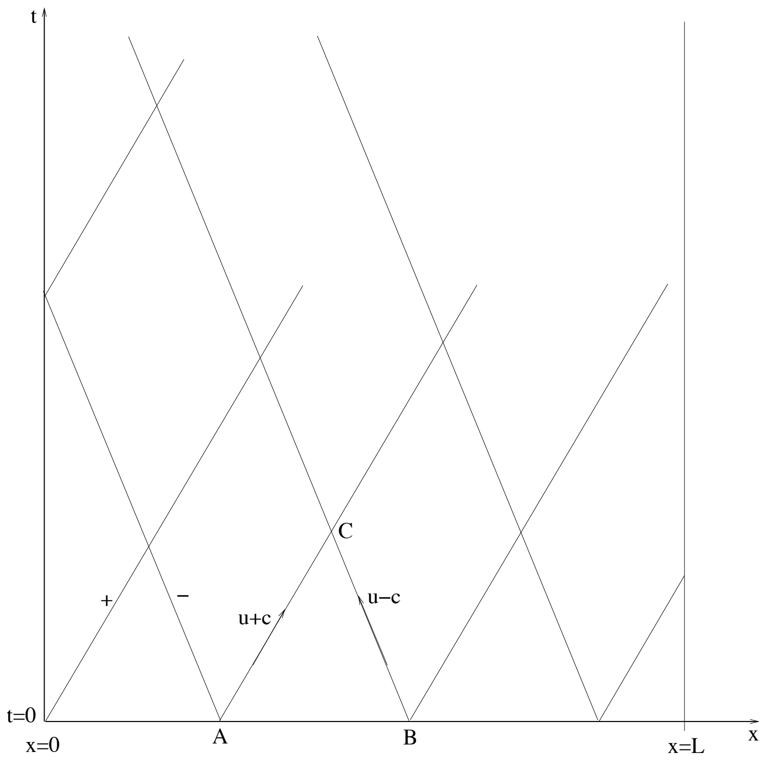


Figure 4.2: $t - x$ plane with characteristics showing information arriving from upstream and downstream at velocities $u \pm c$.

a point as shown in Figure 4.2. The downstream or "+" characteristic has a velocity at any point of $u + \sqrt{gh}$ along which R^+ is constant. In the usual case where u is positive, both parts are positive and the speed is relatively large. The upstream or "−" characteristic has a velocity $u - \sqrt{gh}$, which is negative and smaller in magnitude than the other. So, upstream-propagating disturbance R^- travels more slowly.

Suppose that h and u are given at the segment AB as depicted in Figure 4.2. The Riemann invariants are thus known at AB. Along their characteristics they are constant. As a consequence, in the region ABC both R^+ and R^-, and thus also h and u, are known and uniquely determined by h and u on AB. This implies that any changes in h or u along AB will influence the solution in point C. The region ABC is called the **domain of dependence** of point C.

Obviously, initial and boundary conditions are needed that lead to a well posed problem for the shallow water equations, Eqs. (4.1.1a)−(4.1.1b). Based on the rule concerning the relation between boundary conditions needed and the number of *ingoing* characteristics we conclude that for subcritical flow one boundary condition at each end of the domain is needed, i.e. $x = 0$ and $x = L$. Also, there are two initial conditions required, one for the water level ζ and one for the flow velocity u.

Exercise 4.1.5 Consider a supercritical flow ($\mathrm{Fr} = |u|/\sqrt{gh} > 1$) and $u > 0$. How many boundary conditions need to be imposed and at which side(s) of the domain?

The solution is more complicated when the bottom friction or bed slope is involved. It appears that waves travel at speeds which depend on their wave length. Such waves are called **dispersive** and they may *deform* while travelling. An example are flood waves generated in rivers by rainfall (see Section 4.3).

4.2 Open boundaries and boundary conditions

Often a part of the domain boundary is **artificial** and is introduced *merely* to restrict the domain of computation, for example, to separate the North Sea from the Atlantic Ocean; see Figure 4.3. Another example is depicted in Figure 4.4 in which a river is truncated where the discharge can be described. The artificial boundary is actually a water-water

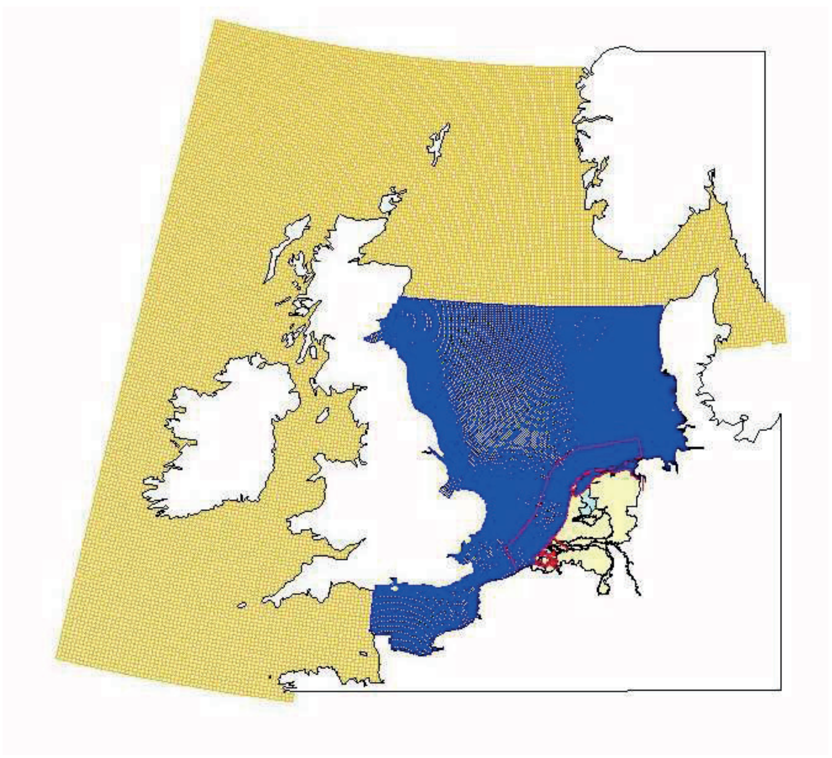


Figure 4.3: Computational grid schematizations. The yellow domain represents the Continental Shelf Model (CSM) and the blue one represents the south part of the North Sea. (Courtesy of Rijkswaterstaat, The Netherlands.)

boundary and is therefore called the **open** boundary. Another type of boundary is a water-land boundary which is genuine. This is called the **closed** boundary. At boundaries we must impose proper boundary conditions. In the case of open boundaries, these conditions must represent the *outside world* accurately.

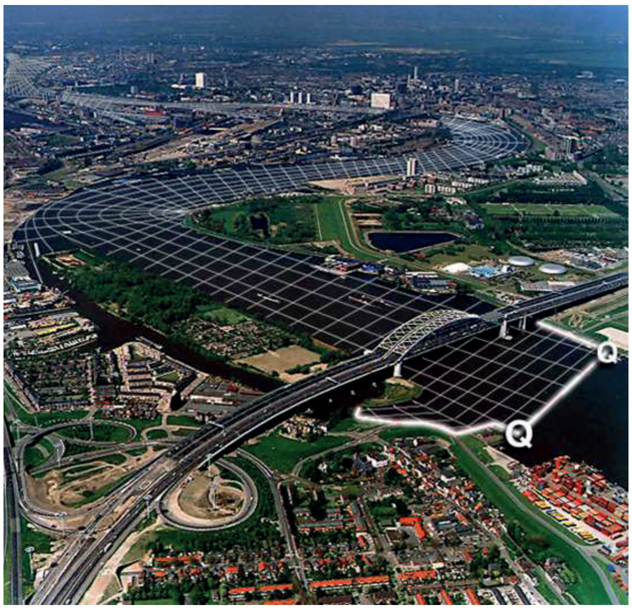


Figure 4.4: A computational grid lay out into the river and the description of the discharge at the boundary.

We consider the following shallow water basins

- tidal basin (continental shelf),
- estuary or tidal river,
- (lower) river, and
- lake.

Typical grid schematizations for these basins are shown in Figure 4.5. In this reader, we restrict ourselves to subcritical flow in one dimensional space with x corresponding to distance along the basin. The length of the considered domain is L. The tidal basin has one open boundary at the sea side, $x = 0$, and one closed boundary at the land side, $x = L$. Both the estuary and river have two open boundaries at both sides. The lake has closed boundaries only. We need to impose two boundary conditions, one at each side of the domain.

Exercise 4.2.1 Explain why we need to impose one boundary condition at each side of the domain.

The boundary conditions to be considered are

- water level ζ,
- velocity u,

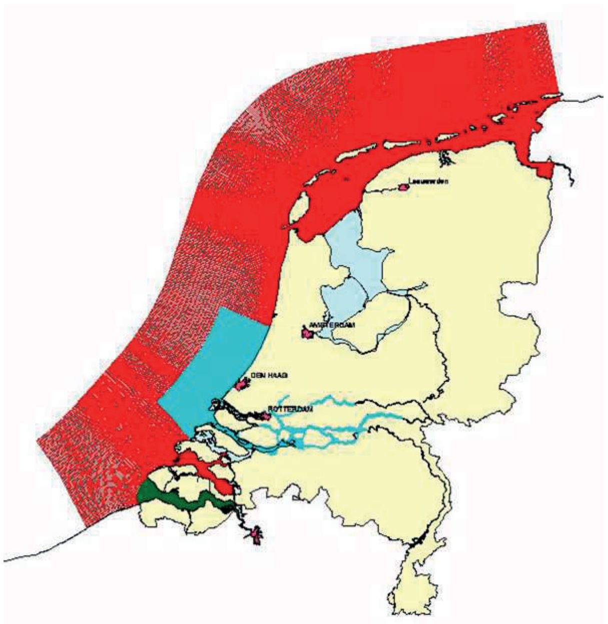


Figure 4.5: Computational grid schematizations. The red domain represents a 2D tidal basin, the green domain an estuary, the blue part in the middle consists of an estuary (western part) and lower rivers (eastern part), and the light blue one represents a lake. (Courtesy of Rijkswaterstaat, The Netherlands.)

- discharge $q \equiv uh$,
- incoming Riemann invariant R^+ or R^-, and
- Sommerfeld radiation condition.

The latter two types of boundary conditions will be explain later.

For each considered basin we may obtain a well posed problem, as follows

basin	left boundary $(x = 0)$	right boundary $(x = L)$
tidal basin	$\zeta(0, t) = f(t)$	$u(L, t) = 0$
estuary	$\zeta(0, t) = f(t)$	$q(L, t) = g(t)$
river	$q(0, t) = f(t)$	$\zeta(L, t) = g(t)$
lake	$u(0, t) = 0$	$u(L, t) = 0$

with $f(t)$ and $g(t)$ the user-prescribed boundary conditions. Note that certain combination of boundary conditions may yield an ill posed problem. An example is imposing discharges or velocities at both sides of the domain, i.e. $q(0,t) = f(t)$ and $q(L,t) = g(t)$, as the in- and outflow are ever increasing or ever decreasing. The solution is then not unique.

The tidal basin, estuary and river can be regarded as **half-closed basins** in which **standing waves** may occur. The lake is an example of a **closed basin**. A standing wave is a phenomenon in which two **progressive waves** are propagating in *opposed* direction with the same amplitude and the same speed though with different sign. For example, consider two waves $\phi_1(x,t) = ae^{\mathrm{i}(\omega t - kx)}$ and $\phi_2(x,t) = ae^{\mathrm{i}(\omega t + kx)}$, where ϕ_1 moves to the right and ϕ_2 moves to the left. A general formulation for the standing wave is then given by

$$\phi(x,t) = \phi_1(x,t) + \phi_2(x,t) = a\left[e^{\mathrm{i}(\omega t - kx)} + e^{\mathrm{i}(\omega t + kx)}\right] = 2ae^{\mathrm{i}\omega t}\cos kx$$

Exercise 4.2.2 Verify this.

A standing wave in a basin has at least a **node** and an **anti-node**. A node indicates zero displacement while an anti-node denotes maximum displacement. Their location depend on the choice of the boundary conditions. For instance, a half-closed basin has one open boundary with imposed water level, the so-called **water level boundary**, and one closed boundary ($u = 0$). The water level is represented as a node at the water level boundary but is an anti-node at the closed boundary. For the velocity the opposite is true. See Figure 4.6a. The longest standing wave has a wave length of four times the length of the basin. Thus, the highest eigen period is $4L/\sqrt{gh}$ (Merian formula). A closed basin has

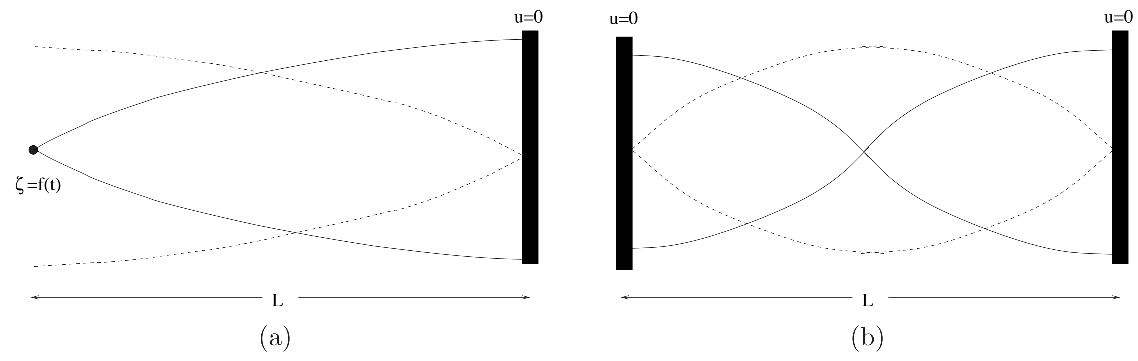


Figure 4.6: Basins with open and closed boundaries: (a) half-closed basin with water level boundary and (b) closed basin. Solid line represents the first harmonic of water level and dashed line represents the first harmonic of velocity.

two closed boundaries with anti-nodes in the case of water level and nodes in the case of velocity. See Figure 4.6b. The eigen period of the first harmonic is $2L/\sqrt{gh}$.

A commonly effect of standing waves in an enclosed or partially enclosed basin, like lakes, bays, harbors and seas, is **resonance** that has been induced by a number of factors, in particular wind and atmospheric pressure variations. These waves are called **seiches** and their period is associated with the eigen period of the basin.

A numerical model for the solution of the shallow water equations may also produce standing waves or seiches that have *no* physical grounds. To see this, we consider a half-closed basin with a water level boundary at $x = 0$ and a closed boundary at $x = L$. The boundary conditions are

$$\zeta(0,t) = ae^{\mathrm{i}\omega t}\,, \quad u(L,t) = 0$$

Moreover, the initial conditions are given by

$$\zeta(x,0) = u(x,0) = 0\,, \quad 0 \leq x \leq L$$

Next, to make the analysis simple to carry out, we simplify the shallow water equations by assuming a very small amplitude, $a \ll d$, a constant depth and no bottom friction,

$$\frac{\partial \zeta}{\partial t} + h\frac{\partial u}{\partial x} = 0 \tag{4.2.1a}$$

$$\frac{\partial u}{\partial t} + g\frac{\partial \zeta}{\partial x} = 0 \tag{4.2.1b}$$

Generally, the solution to these *linearized* equations can be formed as the sum of the homogeneous and particular solutions,

$$\begin{pmatrix} \zeta \\ u \end{pmatrix} = \begin{pmatrix} \zeta \\ u \end{pmatrix}_{\mathrm{H}} + \begin{pmatrix} \zeta \\ u \end{pmatrix}_{\mathrm{P}}$$

The homogeneous part, or the *transient*, represents the eigen modes of the basin and depends on the initial conditions. These conditions are required to start a time-dependent simulation. It takes some time before the influence of the initial state on the solution vanish. Such duration is called the **spin-up time**. Usually the accuracy with which the initial conditions are determined is not relevant unless the simulation period is shorter than the spin-up time. In order to start a model run, certain model parameters need to be specified. This includes the initialization of both the water level and flow velocity. A **cold start** occurs when a simulation is first initialized and then needs to be spin up. A **hort start** is a restart of a model from the saved results of a previous simulation, which can be used to eliminate or reduce the spin-up time.

After the spin-up time the model has reached a state of equilibrium under the forcing of the boundary conditions. Boundary conditions are driving forces for model simulations and as such, we are interested in the influence of the boundary conditions on the flow behaviour inside the basin. Since, the transient is tend to zero as time proceeds, i.e.

$$\lim_{t\to\infty} \begin{pmatrix} \zeta \\ u \end{pmatrix}_{\mathrm{H}} = 0$$

the particular part of the solution *only* depends on the boundary conditions. So, we are looking for a *steady state* solution, i.e.

$$\lim_{t\to\infty}\begin{pmatrix}\zeta\\ u\end{pmatrix}=\begin{pmatrix}\zeta\\ u\end{pmatrix}_{\text{P}}$$

This implies that the simulation end time must be chosen such that the steady state is reached. Hence, it is important to have an idea how long it does take for the simulation to reach equilibrium. We shall explore this later.

Recall that the linear equations has two set of characteristics, one corresponding to right-travelling waves and the other to left-travelling waves so that a solution has the general form

$$\zeta(x,t)=\mu e^{\mathrm{i}(\omega t-kx)}+\nu e^{\mathrm{i}(\omega t+kx)}$$

$$u(x,t)=\alpha e^{\mathrm{i}(\omega t-kx)}+\beta e^{\mathrm{i}(\omega t+kx)}$$

Substitution into Eqs. (4.2.1a)−(4.2.1b) gives the following relations

$$\omega\mu=hk\alpha\,,\quad \omega\nu=-hk\beta$$

$$\omega\alpha=gk\mu\,,\quad \omega\beta=-gk\nu$$

With $c=\omega/k=\sqrt{gh}$, these four relations can be reduced to the following two relations

$$\mu=\alpha\sqrt{\frac{h}{g}}\,,\quad \nu=-\beta\sqrt{\frac{h}{g}}$$

Hence, a solution is given by

$$\begin{pmatrix}\zeta(x,t)\\ u(x,t)\end{pmatrix}=\alpha\begin{pmatrix}\sqrt{\frac{h}{g}}\\ 1\end{pmatrix}e^{\mathrm{i}(\omega t-kx)}+\beta\begin{pmatrix}-\sqrt{\frac{h}{g}}\\ 1\end{pmatrix}e^{\mathrm{i}(\omega t+kx)}$$

where the constants α and β can be determined by means of the boundary conditions.

Exercise 4.2.3 Verify that this is indeed a solution to Eqs. (4.2.1a)−(4.2.1b) and find the constants α and β.

However, this solution with $\alpha\neq 0$ and $\beta\neq 0$ does not fulfill the initial conditions. Indeed, the solution found is the steady state solution. To obtain the complete solution we need to find a transient which is given by[1]

$$\begin{pmatrix}\zeta(x,t)\\ u(x,t)\end{pmatrix}_{\text{H}}=\sum_{j=0}^{\infty}e^{\mathrm{i}k_jct}\left[\begin{pmatrix}\mu_j\\ \nu_j\end{pmatrix}e^{-\mathrm{i}k_jx}+\begin{pmatrix}-\mu_j\\ \nu_j\end{pmatrix}e^{\mathrm{i}k_jx}\right]$$

[1]The method to find this homogeneous solution is quite complicated. So we end up with a closed form of the solution.

with $k_j = 2\pi/L_j$ and $L_j = 4L/(2j+1)$. The constants of this series, μ_j and ν_j, can be found using the initial conditions. Clearly, the eigen modes of the basin are the standing waves.

Exercise 4.2.4 Explain why this series represents standing waves.

However, the transient will not disappear since, in this specific case,

$$\lim_{t\to\infty} \begin{pmatrix} \zeta \\ u \end{pmatrix}_{\text{H}} \neq 0$$

What are the practical consequences of this? The answer is probably best illustrated by doing some numerical experiments. For this purpose, we choose a tidal basin with a length of $L = 20$ km and a uniform depth of $d = 25$ m. At $x = L$ the basin is closed and at $x = 0$ we impose $\zeta(0, t) = \sin(2\pi t/T)$ with period $T = 40,000$ s. Figure 4.7a shows the obtained time series of the water level. Clearly, the irregular behaviour of the solution is due to the eigen modes of which the largest eigen period is $4L/\sqrt{gh} \approx 5,100$ s. Also, these small

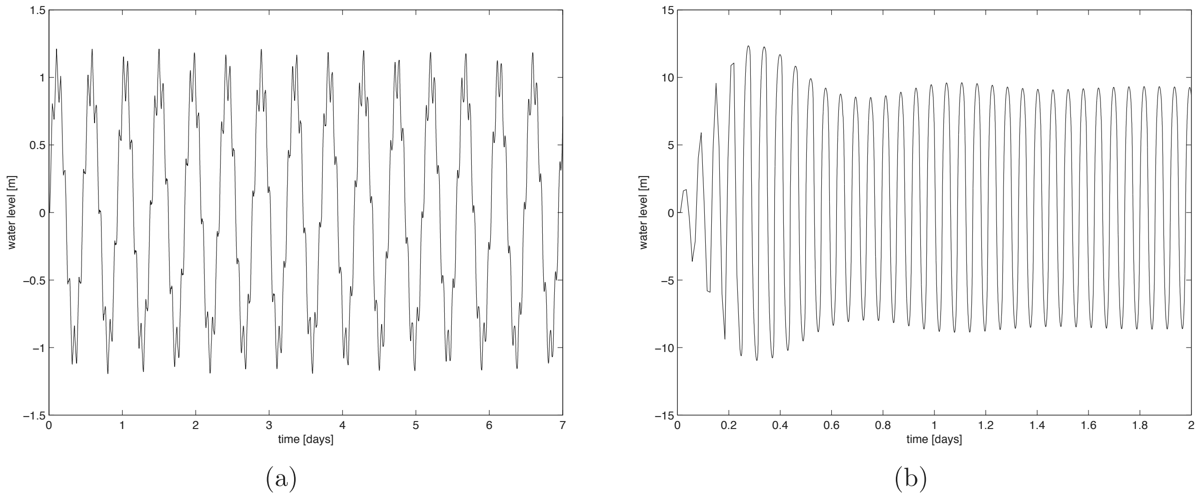


Figure 4.7: Time series of computed water level obtained with forcing period of (a) 40,000 s and (b) 5,100 s.

oscillations persist. Hence, the spin-up time is infinite. Let us choose a period of external forcing as $T = 5,100$ s. It is expected that the solution becomes very inaccurate because of resonance. This is indeed the case as depicted in Figure 4.7b. The eigen modes now act as seiches.

So, the central question will be: how to get rid of the eigen modes?

Recall the spring-mass system as discussed in Section 2.2.1. The steady state solution of this system is determined by the external forcing. The transient solution consists of eigen modes and is dictated by the initial data. When the eigenvalues are complex, the transient behaves like a harmonic motion. This harmonic motion is undamped if the eigenvalues are purely imaginary. In that case this motion persists for all time but remains stable (see Figure 2.9). On the other hand, the harmonic motion is damped if the real part of the eigenvalues is negative. Hence, this motion decays to zero as $t \to \infty$ and the effect of the initial data dies out (see Figures 2.11 and 2.12). So, to let the eigen modes disappear in the end, we must have some physical and/or numerical damping in the system.

A natural candidate for the physical damping is bottom friction as we almost always deal with it in the coastal areas. This physical process is represented by the term $c_f u|u|/h$ in Eq. (4.1.1b). A typical time scale of this process is

$$\tau_b \sim \frac{h}{|u|c_f}$$

Exercise 4.2.5 Verify this.

After a while the bottom friction works at full strength and the eigen modes disappear. This is called underdamping (cf. Figure 2.11). In other words, if $t > \tau_b$ then all the effects of initial data have died out. The time scale τ_b represents the *order of magnitude* of the spin-up time. Let us rerun the abovementioned simulation, this time with bottom friction added. We choose $c_f = 0.0025$. Figure 4.8 shows the numerical result. Clearly, after 6 days the eigen modes are completely disappear. In the same simulation it appears that the

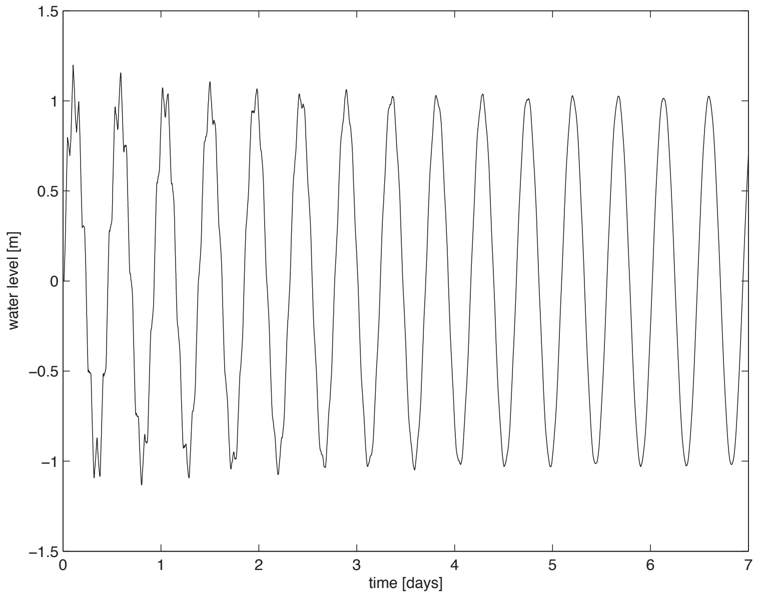


Figure 4.8: Time series of computed water level obtained with bottom friction.

maximum flow velocity is 0.015 m/s. Hence, $\tau_b \approx 8$ days. This is clearly a conservative estimate of the spin-up time. The spin-up time can be made shorter if we also introduce some numerical damping in the numerical scheme. Examples are implicit Euler for time integration and an upwind scheme for approximation of advection.

Next, suppose we consider the Riemann invariant as a boundary condition instead of the water level. The corresponding open boundary is called the **Riemann boundary**. The expression for the Riemann invariants of Eqs. (4.2.1a)−(4.2.1b) can be found using the method of characteristics. Eq. (4.2.1a) is multiplied with $\sqrt{g/h}$ and subsequently added to Eq. (4.2.1b), yielding

$$\frac{\partial}{\partial t}\left(u+\sqrt{\frac{g}{h}}\zeta\right)+\sqrt{gh}\frac{\partial}{\partial x}\left(u+\sqrt{\frac{g}{h}}\zeta\right)=0$$

Exercise 4.2.6 Verify this.
(*Hint*: $g=\sqrt{gh}\sqrt{g/h}$.)

The Riemann invariant propagating along a characteristic starting at $x=0$ with speed $+\sqrt{gh}$ is $R^+=u+\zeta\sqrt{g/h}$. The other Riemann invariant propagating along a characteristic starting at $x=L$ with speed $-\sqrt{gh}$ is given by $R^-=u-\zeta\sqrt{g/h}$.

Exercise 4.2.7 Verify this and explain why this system has two ingoing characteristics.

We impose the following boundary conditions

$$u(0,t)+\sqrt{\frac{g}{h}}\zeta(0,t)=f(t)\,,\quad u(L,t)-\sqrt{\frac{g}{h}}\zeta(L,t)=g(t)$$

with $f(t)$ and $g(t)$ the user-prescribed functions and we have the following initial conditions

$$u(x,0)+\sqrt{\frac{g}{h}}\zeta(x,0)=p(x)\,,\quad u(x,0)-\sqrt{\frac{g}{h}}\zeta(x,0)=q(x)$$

with $p(x)$ and $q(x)$ the user-prescribed functions. Then the solution is given by

$$u(x,t)+\sqrt{\frac{g}{h}}\zeta(x,t)=\begin{cases}f(t-\frac{x}{c})\,, & x-ct<0\\ \\ p(x-ct)\,, & x-ct\geq 0\end{cases}$$

and

$$u(x,t)-\sqrt{\frac{g}{h}}\zeta(x,t)=\begin{cases}g(t+\dfrac{x-L}{c})\,, & x+ct>L\\ \\ q(x+ct)\,, & x+ct\leq L\end{cases}$$

Let us choose $f(t)=ae^{\mathrm{i}\omega t}$, and $g(t)=p(x)=q(x)=0$. The solution is given by

$$u(x,t)+\sqrt{\frac{g}{h}}\zeta(x,t)=\begin{cases}0\,, & x\geq ct\\ \\ ae^{\mathrm{i}\omega\left(t-\frac{x}{c}\right)}\,, & x<ct\end{cases}$$

and

$$u(x,t) = \sqrt{\frac{g}{h}}\,\zeta(x,t)$$

The influence of the initial conditions will disappear as soon as $t > L/\sqrt{gh}$. This spin-up time is of the order of magnitude of the eigen period of the basin. This is similar to critical damping (cf. Figure 2.13). Due to the imposition of a Riemann boundary, the transient solution does not act as a standing wave but runs out of the basin very fast.

The implications of applying a Riemann boundary instead of water level boundary will be studied in the next numerical experiment. The boundary conditions are thus

$$u(0,t) + \sqrt{\frac{g}{h}}\,\zeta(0,t) = \sin(\frac{2\pi t}{T})\,, \quad u(L,t) = 0$$

with T = 40,000 s. There is no bottom friction, so we reset $c_f = 0$. The obtained time series of the water level is displayed in Figure 4.9. Obviously, the effect of the initial data

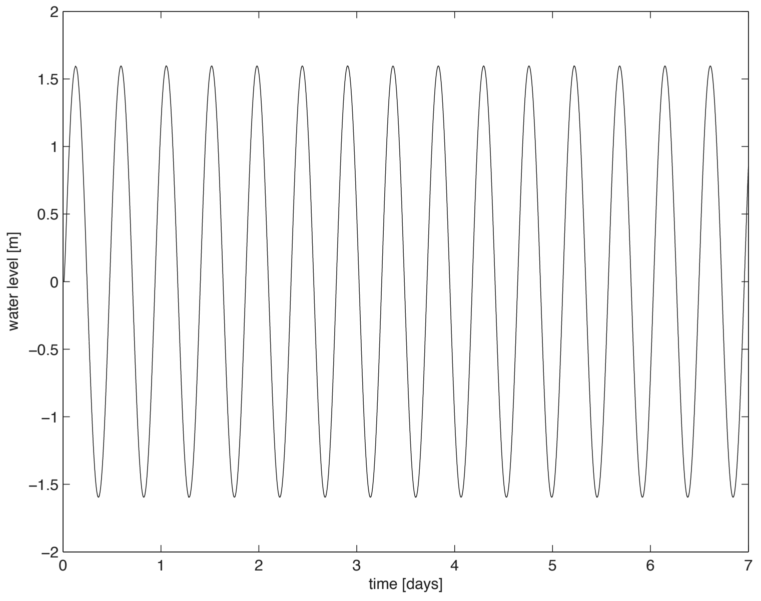


Figure 4.9: Time series of computed water level obtained with Riemann invariant at open boundary (no friction).

is nil. The expected spin-up time is $L/\sqrt{gh} \approx 1,300$ s, just one percent of a day.

Summarizing, the transient consists of the eigen modes of the basin and is steered by the initial conditions. Depending on the *choice* of the boundary conditions these eigen modes may result in standing waves or not. In any case, they can be removed by adding some damping so that we end up with the steady state. The model has been spun up.

Exercise 4.2.8 Mention at least four ways to diminish the effect of the initial data.

In the last experiment there is a full reflection at the closed boundary, $x = L$. Indeed, $u = 0$ implies $R^- = -R^+$ since $R^+ + R^- = 2u = 0$. The steady state solution is therefore a standing wave. To see this, we generalize the Riemann boundary condition as follows

$$R^+(0,t) = f(t) = e^{\mathrm{i}\omega t}$$

The solution for R^+ is then given by

$$R^+(x,t) = f(t - \frac{x}{c}) = e^{\mathrm{i}\omega\left(t-\frac{x}{c}\right)}$$

Due to reflection at the closed boundary, we have

$$R^-(L,t) = g(t) = -R^+(L,t) = -f(t - \frac{L}{c}) = -e^{\mathrm{i}\omega\left(t-\frac{L}{c}\right)}$$

and so,

$$R^-(x,t) = g(t + \frac{x-L}{c}) = -f(t + \frac{x-2L}{c}) = -e^{\mathrm{i}\omega\left(t+\frac{x-2L}{c}\right)}$$

By simply adding and substracting we find the solution for the flow velocity and water level, respectively, as follows

$$\begin{aligned} u(x,t) &= \frac{1}{2}e^{\mathrm{i}\omega\left(t-\frac{L}{c}\right)}\left[e^{-\mathrm{i}\omega\left(\frac{x-L}{c}\right)} - e^{\mathrm{i}\omega\left(\frac{x-L}{c}\right)}\right] = \mathrm{i}\, e^{\mathrm{i}\omega\left(t-\frac{L}{c}\right)} \sin\left(\omega\frac{x-L}{c}\right) \\ &= \mathrm{i}\, e^{\mathrm{i}\omega t} \sin kx \end{aligned}$$

and

$$\zeta(x,t) = \frac{1}{2}\sqrt{\frac{h}{g}}\, e^{\mathrm{i}\omega\left(t-\frac{L}{c}\right)}\left[e^{-\mathrm{i}\omega\left(\frac{x-L}{c}\right)} + e^{\mathrm{i}\omega\left(\frac{x-L}{c}\right)}\right] = \sqrt{\frac{h}{g}}\, e^{\mathrm{i}\omega t} \cos kx$$

Exercise 4.2.9 Verify this and compare the solution with Figure 4.9.

In some cases we are dealing with an **open basin** in the sense that no boundary condition is specified at infinity. However, to limit the extent of computation, it is necessary to introduce an artificial boundary. To guarantee a well posed problem we must impose boundary conditions at this artificial boundary. These boundary conditions should be carefully chosen in order to simulate the unbounded surroundings and to minimize the influence of the artificial boundary. For instance, one must prevent outgoing waves from reflecting from the artificial boundary. This is achieved by using **absorbing boundary conditions** to simulate the undisturbed outgoing waves.

We consider a 1D basin with two open boundaries. At the left boundary, $x = 0$, we impose an ingoing wave,

$$R^+(0,t) = e^{\mathrm{i}\omega t}$$

At the right boundary, $x = L$, this progressive wave must travelling out of the domain without reflection. Obviously, a water level or velocity boundary is strongly reflective as there is still an ingoing characteristic at the boundary. Indeed, at the water level boundary, we have $R^-(L,t) = R^+(L,t) - 2\sqrt{g/h}\,\zeta(L,t)$ and at the velocity boundary, we obtain $R^-(L,t) = 2u(L,t) - R^+(L,t)$. Instead, we must require $R^-(L,t) = 0$, or

$$u(L,t) = \sqrt{\frac{g}{h}}\,\zeta(L,t) \tag{4.2.2}$$

Exercise 4.2.10 Show that the solution is given by

$$\zeta(x,t) = \frac{1}{2}\sqrt{\frac{h}{g}}\,e^{\mathrm{i}(\omega t - kx)}\,, \quad u(x,t) = \frac{1}{2}e^{\mathrm{i}(\omega t - kx)}$$

and verify that it represents a progressive wave.

From Eq. (4.2.2) it follows that

$$\frac{\partial u}{\partial x} = \sqrt{\frac{g}{h}}\frac{\partial \zeta}{\partial x} \quad \text{at } x = L$$

Substitution into Eq. (4.2.1a) yields

$$\frac{\partial \zeta}{\partial t} + \sqrt{gh}\frac{\partial \zeta}{\partial x} = 0 \quad \text{at } x = L$$

This is the well-known **Sommerfeld radiation condition** for nondispersive waves and is an example of an absorbing boundary condition. An alternative to this condition is the following

$$\frac{\partial u}{\partial t} + \sqrt{gh}\frac{\partial u}{\partial x} = 0 \quad \text{at } x = L$$

Exercise 4.2.11 Derive this alternative Sommerfeld radiation condition.

A more general form of the Sommerfeld radiation condition is

$$\frac{\partial \phi}{\partial t} + \vec{c}\cdot\nabla\phi = 0 \tag{4.2.3}$$

with ϕ a wave quantity such as surface elevation, $\vec{c}$ the local phase speed of the wave and ∇ the gradient operator. In two spatial dimensions, they are given by $\vec{c} = (c_x, c_y)^{\mathrm{T}}$ and $\nabla = (\partial/\partial x, \partial/\partial y)^{\mathrm{T}}$. Eq. (4.2.3) is recognized as a 2D convection equation for wave propagation without deformation.

In the final numerical experiment we repeat the last run, this time with the right boundary as an open one where the Sommerfeld radiation condition is imposed. Figure 4.10 depicts the obtained result.

Exercise 4.2.12 Compare Figure 4.10 with Figure 4.9 and explain the difference.

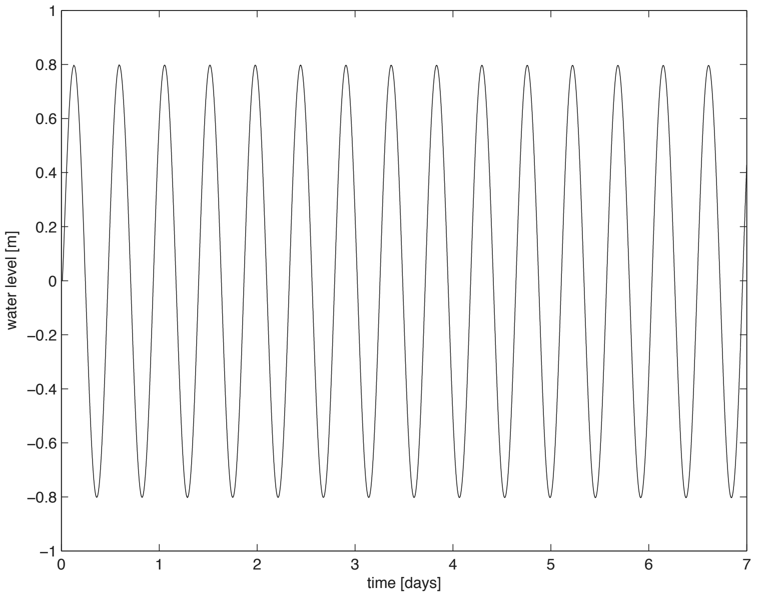


Figure 4.10: Time series of computed water level obtained with Riemann invariant at left boundary and Sommerfeld radiation at right boundary.

Often the requirement of a perfect wave absorption cannot be achieved. For instance, Eq. (4.2.3) only works in an ideal case of a nondispersive, monochromatic wave in an frictionless fluid in a basin with constant depth. In addition, this Sommerfeld radiation condition enables the boundary to freely propagate incident or oblique waves out of the basin as long as the phase speed and the wave direction at the boundary are chosen properly. However, for dispersive waves, like flood waves, these local properties are often not known *a priori* and must be determined during the course of the computation, if possible, such that the reflected outgoing waves will be minimized as much as possible. Sommerfeld radiation conditions are therefore referred to as a **weakly reflective boundary condition**.

4.3 Flood propagation in rivers

A common problem in river engineering is how far upstream water levels might be changed, and hence flooding possibly enhanced, due to downstream works such as the installation of a bridge or other obstacles. In subcritical flow, the effects of any control can propagate back up the river. On the other hand, in supercritical flow, all disturbances are swept downstream, so that the effects of a control cannot be felt upstream.

It is usually the upstream boundary condition that drives the whole model describing a subcritical flow in a river, where a flood wave enters, via the specification of the time variation of discharge q at the boundary. This is usually given in the form of a discharge hydrograph. The location of the upstream boundary condition is usually well defined. If there is a structure downstream that restricts or controls the flow, the choice of location and the nature of the control are both made easy. The downstream boundary condition is usually the water level ζ described. Sometimes the computational domain might be truncated without the presence of a control, such as just below a town, for which the danger of flooding is to be investigated, but where there is no control structure on the stream. In this case an absorbing boundary condition might be appropriate.

The flow in a river is assumed to be steady and gradually varied. Far away from the obstacle or control, the flow is uniform and the depth is said to be equilibrium. This **equilibrium depth** h_e is determined by the balance between bottom slope and friction losses. In general the depth at a control is not equal to the equilibrium depth. Figure 4.11 shows a typical subcritical flow in a river, where the depth at a control is larger than the equilibrium depth. The surface decays somewhat curved to equilibrium depth upstream from a downstream

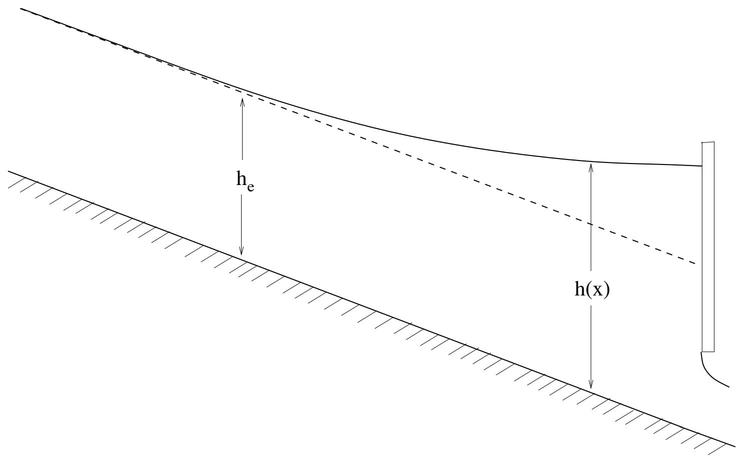


Figure 4.11: Subcritical flow retarded by a gate, showing typical behaviour of the free surface and decaying upstream to equilibrium depth h_e.

control. This phenomenon is called **backwater**. The transition between conditions at the control, and where there is uniform flow is described by the shallow water equations.

In case the flow is steady, $\partial\zeta/\partial t = 0$, and uniform, $\partial u/\partial x = 0$, from Eq. (4.1.1a), it follows

$$\frac{\partial h}{\partial x} = 0 \quad \Rightarrow \quad \frac{\partial \zeta}{\partial x} = -\frac{\partial d}{\partial x} \equiv -i_b$$

where i_b is the given bottom slope. Eq. (4.1.1b) reduces to

$$-gi_b + c_f\frac{u|u|}{h} = 0$$

Exercise 4.3.1 Verify this.

Obviously, friction dominates over transport, which is typical of flood waves in rivers. With $u > 0$, we have

$$u = \sqrt{\frac{ghi_b}{c_f}}$$

The friction coefficient c_f is often not known at all. However, for steady uniform flow, this coefficient can be expressed in terms of the Chézy coefficient C,

$$c_f = \frac{g}{C^2}$$

so that

$$u = C\sqrt{hi_b}$$

The equilibrium depth h_e can be derived, as follows

$$h_e = \frac{u^2}{C^2 i_b}$$

With $q = uhb$ and b the **effective river width**, we have

$$h_e = \left(\frac{q}{bC\sqrt{i_b}}\right)^{2/3}$$

It also follows that

$$\frac{\partial u}{\partial x} = \frac{1}{2}C\sqrt{\frac{i_b}{h}}\frac{\partial h}{\partial x}$$

Substitution into Eq. (4.1.1a) gives (recall $\partial d/\partial t = 0$)

$$\frac{\partial h}{\partial t} + u\frac{\partial h}{\partial x} + \frac{1}{2}C\sqrt{hi_b}\frac{\partial h}{\partial x} = \boxed{\frac{\partial h}{\partial t} + \frac{3}{2}u\frac{\partial h}{\partial x} = 0} \tag{4.3.1}$$

This is a convection equation for flood waves as represented by the water depth h, and is called the **kinematic wave equation**. This equation seems deceptively simple. However, it is nonlinear, and the dependence on h is complicated, such that analytical solution is often not possible. For instance, the characteristics along which h is constant are not parallel and thus, the flood wave is *deformed* during propagation. Moreover, due to bottom friction this wave is damped. One boundary condition must be supplied to obtain the solution. In subcritical flow, this is the depth at a downstream control; in supercritical flow it is the depth at an upstream control.

The flood wave is a progressive wave which propagate at a speed of $^3/_2u$. Assume a flood wave of the form

$$\tilde{h}e^{\mathrm{i}(\omega t - kx)}$$

with $\tilde{h}$ the amplitude. The phase velocity is given by

$$c_\mathrm{p} = \frac{3}{2}u = \frac{3}{2}C\sqrt{hi_b}$$

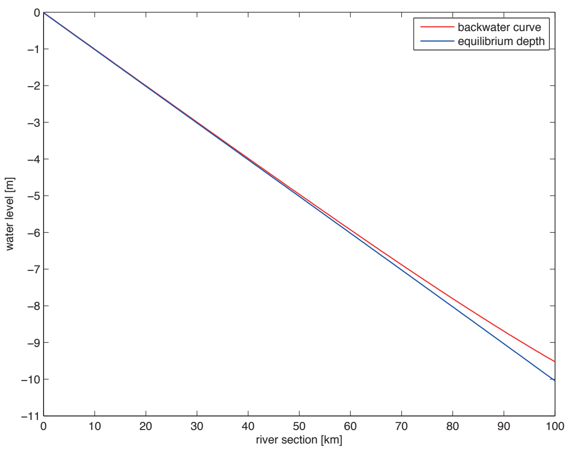
Clearly, the phase velocity depends on the wave length and so, the flood wave is dispersive.

Exercise 4.3.2 Verify this.

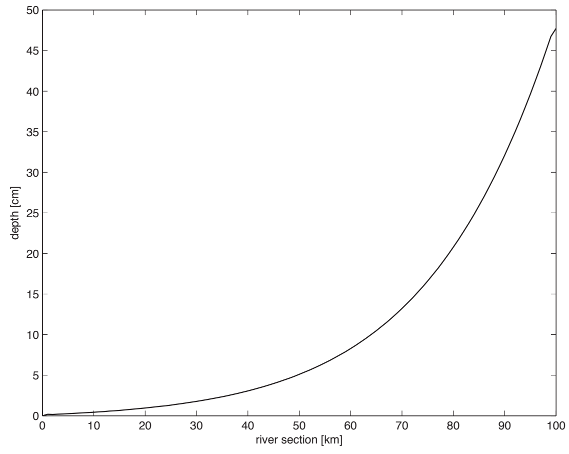
With the numerical solution of the shallow water equations for river problems we may observed some unwanted effects. For instance, initially, we have to choose a water depth that may not match with the equilibrium depth. Thus, the initial volume of water is incorrect and its influence on the model results is thus spin up. However, this effect can propagate easily out of the domain as it is a progressive wave. The corresponding spin-up time scale is

$$\tau \sim \frac{2}{3}\frac{L}{u} = \frac{2}{3}\frac{L}{C\sqrt{hi_b}}$$

Another issue is the *artificial* backwater due to the choice of the boundary conditions − downstream water level control and upstream discharge. The water level downstream can be such that the water depth at the control is larger than the equilibrium depth. This artificial backwater can be noticable in a substantial part of the domain. We illustrate this with the following example. A channel with a length of 100 km and a width of 1 m has a bottom slope of 10^{-4}. Upstream we impose a discharge of 5 m^2/s, while downstream a depth of 7 m is described. The Chézy coefficient is 30 $\sqrt{\mathrm{m}}/\mathrm{s}$. This implies an equilibrium depth of $h_e = 6.5248$ m. Figure 4.12a shows the equilibrium depth and the artificial backwater. The difference between these two curves is shown in Figure 4.12b. Halfway the



(a)



(b)

Figure 4.12: Computed depth along the river section: (a) backwater and equilibrium depth and (b) equilibrium depth substracted from backwater.

river section there is already a difference of 5 cm, while at the downstream end the water level is approximately 50 cm too high.

An alternative would be to impose a water level upstream and a discharge downstream. However, this will lead to an ill posed problem with an arbitrary backwater. A better alternative is the use of so-called **rating curves**. These curves give a relationship between the discharge and (equilibrium) depth. Rating curves are often available as measured data. Based on the given downstream discharge the water level downstream can then be described. A more simple but effective method is to impose a zero gradient of the water depth at the downstream boundary

$$\frac{\partial h}{\partial x}(L,t) = 0$$

This boundary condition can be combined with the rating curves that can be used to calibrate the model by adapting the bottom friction coefficient such that a correct equilibrium depth can be obtained.

4.4 Discretization methods

We consider the linearized shallow water equations, Eqs. (4.2.1a)−(4.2.1b). Let us apply the MOL approach. Hence, we choose time integration independent from space discretization. We start by considering the space discretization. We define a computational grid in which both the unknowns ζ and u are located in the same grid point. See Figure 4.13. This is called a **colocated grid arrangement**. This grid consists of $M + 1$ grid points

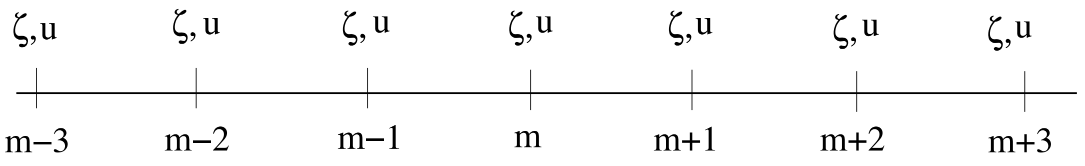


Figure 4.13: Colocated grid arrangement.

equidistantly distributed and the mesh width is Δx. We discretize the equations by second order central differences in space in the following way

$$\frac{d\zeta_m}{dt} + h\frac{u_{m+1} - u_{m-1}}{2\Delta x} = 0\,, \quad m = 1, \dots, M-1$$

$$\frac{du_m}{dt} + g\frac{\zeta_{m+1} - \zeta_{m-1}}{2\Delta x} = 0\,, \quad m = 1, \dots, M-1$$

This semi-discretized system has to be completed with appropriate boundary conditions. For subcritical flow, one boundary condition must be available at the boundary point $m = 0$, and one at $m = M$. We shall not go into details as there are many ways to implement the boundary conditions.

We observe that the above semi-discretized equations have two *independent* sets of unknowns

$$\{\zeta_{2j}, u_{2j+1}\} \quad \text{and} \quad \{\zeta_{2j+1}, u_{2j}\}\,, \quad j = 0, \dots, (M-1)/2$$

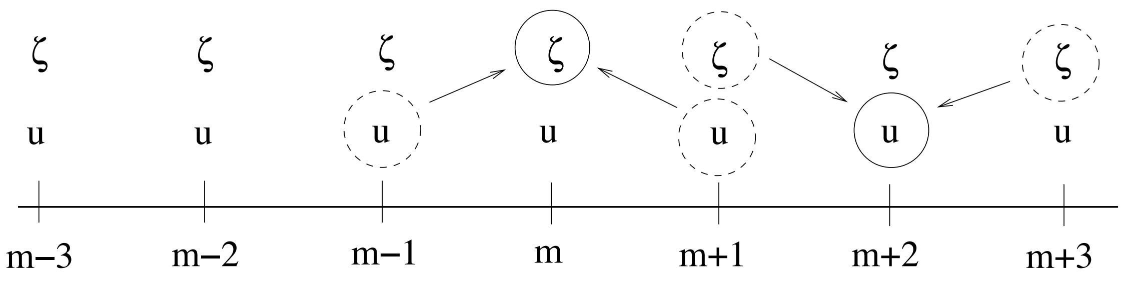


Figure 4.14: Two independent sets of ζ and u.

because for updating ζ_m you need u_{m-1} and u_{m+1}, while for updating u_m you need ζ_{m-1} and ζ_{m+1}, see Figure 4.14. Hence, it is sufficient to consider one of the independent sets as depicted in Figure 4.15. This grid arrangement is called a **staggered grid** where the two dependent variables ζ and u are carried at alternate grid points. The points where the

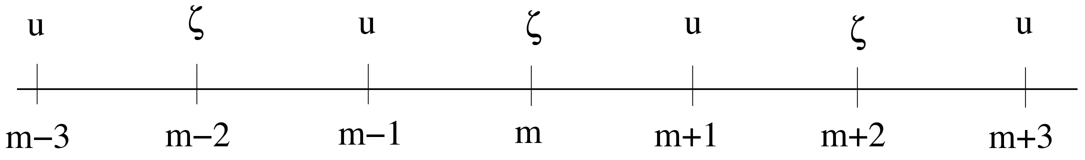


Figure 4.15: A staggered grid.

water level is located are called the **water level points**. Also, we have **velocity points** where the flow velocities are defined. It is common to use whole indices for the water level points and half indices for the velocity points as indicated in Figure 4.16.

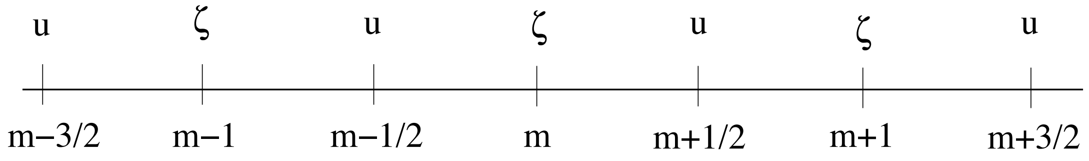


Figure 4.16: A staggered grid with whole and half indices.

The semi-discretized equations become

$$\frac{d\zeta_m}{dt} + h\frac{u_{m+1/2} - u_{m-1/2}}{\Delta x} = 0\,, \quad m = 1, \ldots, M-1$$

$$\frac{du_{m+1/2}}{dt} + g\frac{\zeta_{m+1} - \zeta_m}{\Delta x} = 0\,, \quad m = 1, \ldots, M-1$$

The next step to carry out is time integration. A choice would be the explicit Euler scheme. However, by means of the method of characteristics, one can show that the *linearized* shallow water equations are similar to the following convection equations[2]

$$\frac{\partial R^{\pm}}{\partial t} \pm \sqrt{gh}\frac{\partial R^{\pm}}{\partial x} = 0$$

with $R^{\pm} = u \pm \zeta\sqrt{g/h}$ the Riemann invariants. In Chapter 3 we observed that explicit Euler combined with central differencing forms an unconditionally unstable scheme for the simple convection equation like the one above. This implies that this combination is unstable for the linearized shallow water equations as well. Unless there is some damping involved. One can think of bottom friction as perhaps the most common dissipation term. Obviously, this is not always a save option.

An alternative would be the application of the $\theta-$method, as follows

$$\frac{\zeta_m^{n+1} - \zeta_m^n}{\Delta t} + \frac{h}{\Delta x}\left[\theta\left(u_{m+1/2}^{n+1} - u_{m-1/2}^{n+1}\right) + (1-\theta)\left(u_{m+1/2}^n - u_{m-1/2}^n\right)\right] = 0$$

$$\frac{u_{m+1/2}^{n+1} - u_{m+1/2}^n}{\Delta t} + \frac{g}{\Delta x}\left[\theta\left(\zeta_{m+1}^{n+1} - \zeta_m^{n+1}\right) + (1-\theta)\left(\zeta_{m+1}^n - \zeta_m^n\right)\right] = 0$$

with $1/2 \le \theta \le 1$ to be chosen. For a steady state computation, the best choice is implicit Euler ($\theta = 1$) because (a) time accuracy is not relevant and (b) it is the fastest way to spin up the computation.

Exercise 4.4.1 Explain the reasons (a) and (b).

The above scheme is unconditionally stable, and large time steps can be taken. However, one obtains at each time step a system of $2M$ simultaneous algebraic equations in $2M$ unknowns that have to be solved.

Exercise 4.4.2 Explain why we need to solve a system of equations in this case.

In the next two sections we shall outline the two classical schemes, the leapfrog scheme (Section 4.4.1) and the Preissmann scheme (Section 4.4.2).

4.4.1 The leapfrog scheme

A good candidate for time integration is the midpoint rule. The combination of the midpoint rule and central differences leads to a conditionally stable scheme for the linearized shallow water equations. This discretization method reads

$$\frac{\zeta_m^{n+1} - \zeta_m^{n-1}}{2\Delta t} + h\frac{u_{m+1}^n - u_{m-1}^n}{2\Delta x} = 0$$

$$\frac{u_m^{n+1} - u_m^{n-1}}{2\Delta t} + g\frac{\zeta_{m+1}^n - \zeta_{m-1}^n}{2\Delta x} = 0$$

[2] These equations can also been derived using the similarity transformation.

and is called the **leapfrog** scheme for solving the shallow water equations. Again, we observed two independent sets of unknowns with respect to the time levels. This implies that staggering of the unknowns can also taken place in time. Like in space discretization, we can use whole indices for ζ and half indices for u, so that the leapfrog scheme can be rewritten as

$$\frac{\zeta_m^{n+1} - \zeta_m^n}{\Delta t} + h \frac{u_{m+1/2}^{n+1/2} - u_{m-1/2}^{n+1/2}}{\Delta x} = 0$$

$$\frac{u_{m+1/2}^{n+1/2} - u_{m+1/2}^{n-1/2}}{\Delta t} + g \frac{\zeta_{m+1}^n - \zeta_m^n}{\Delta x} = 0$$

This scheme is also known as the **Hansen** scheme as introduced by Walter Hansen in 1956. Its stencil is depicted in Figure 4.17. It is an explicit scheme: we first compute $u_{m+1/2}^{n+1/2}$

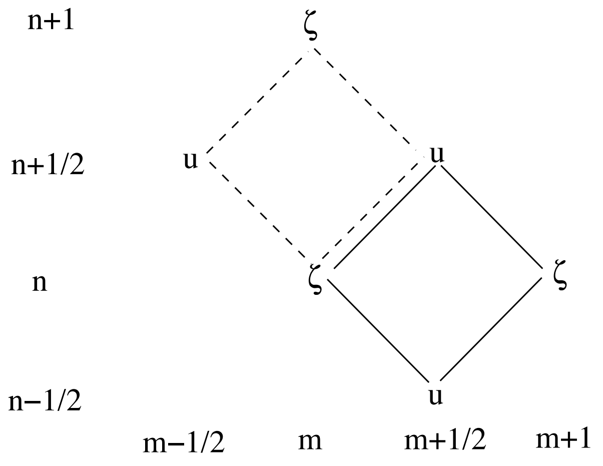


Figure 4.17: The stencil of the leapfrog scheme. The solid line indicate an update in u and the dashed line indicate an update in ζ.

and then ζ_m^{n+1}. So, u and ζ leap frog within one time step. As a consequence, this scheme requires one initial condition for ζ and one for u. Hence, the scheme does not have spurious solution. It is easy to verify that this scheme is second order accurate in both time and space.

Exercise 4.4.3 Verify this.

It is expected that the leapfrog scheme is conditionally stable and one can show that the stability limit is given by

$$\sigma = \frac{\sqrt{gh}\,\Delta t}{\Delta x} \leq 1$$

A formal proof can be found in e.g. Wesseling (2001). In this lecture notes it is sufficient to study the stability properties by considering the simple convection equation as given by

$$\frac{\partial c}{\partial t} \pm \sqrt{gh}\frac{\partial c}{\partial x} = 0$$

The results of this stability analysis are applied to the shallow water equations as well. This also hold for the amplitude- and phase-error analysis. For instance, one can show that the leapfrog scheme has no amplitude error.

Exercise 4.4.4 Verify this.

Exercise 4.4.5 Derive the phase error of the leapfrog scheme.

4.4.2 The Preissmann scheme

A popular numerical method for the simulation of flow in rivers and channel networks is the **Preissmann** scheme as found in many commercial packages. The starting point is the use of colocated grids; see Figure 4.13. Thus the two dependent variables ζ and u are defined in the same grid point. The scheme is based on the trapezoidal rule applied in space. The semi-discretized equations reads

$$\frac{d\overline{\zeta}_{m+1/2}}{dt} + h\frac{u_{m+1} - u_m}{\Delta x} = 0\,, \quad m = 1, \ldots, M-1$$

$$\frac{d\overline{u}_{m+1/2}}{dt} + g\frac{\zeta_{m+1} - \zeta_m}{\Delta x} = 0\,, \quad m = 1, \ldots, M-1$$

The over bar notation indicates averaging

$$\overline{\zeta}_{m+1/2} = \frac{1}{2}\left(\zeta_m + \zeta_{m+1}\right)\,, \quad \overline{u}_{m+1/2} = \frac{1}{2}\left(u_m + u_{m+1}\right)$$

For time integration we apply the $\theta-$scheme of which the trapezoidal rule is a special case ($\theta = 1/2$)

$$\frac{\overline{\zeta}^{n+1}_{m+1/2} - \overline{\zeta}^{n}_{m+1/2}}{\Delta t} + \frac{h}{\Delta x}\left[\theta\left(u^{n+1}_{m+1} - u^{n+1}_{m}\right) + (1-\theta)\left(u^{n}_{m+1} - u^{n}_{m}\right)\right] = 0$$

$$\frac{\overline{u}^{n+1}_{m+1/2} - \overline{u}^{n}_{m+1/2}}{\Delta t} + \frac{g}{\Delta x}\left[\theta\left(\zeta^{n+1}_{m+1} - \zeta^{n+1}_{m}\right) + (1-\theta)\left(\zeta^{n}_{m+1} - \zeta^{n}_{m}\right)\right] = 0$$

We obtain a set of algebraic equations in the values of ζ and u at the corners of a rectangle in space-time (Figure 4.18), which need to be solved every time step. Due to its compactness the Preissmann scheme is very suitable for space varying grid sizes as is often the case in practical applications. Moreover, this scheme appears to be very robust and stable. Furthermore, the order of accuracy is two in both time and space if $\theta = 1/2$, but first order accurate in time when $1/2 < \theta \leq 1$.

Exercise 4.4.6 Show that the Preissmann scheme is second order accurate in time and space if $\theta = 1/2$, and unconditionally stable for $1/2 \leq \theta \leq 1$.

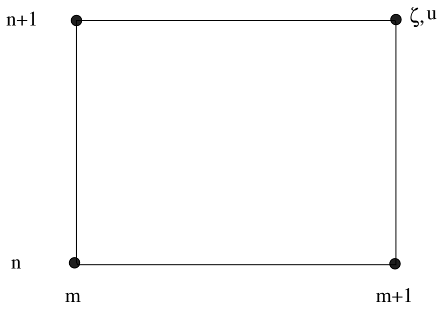


Figure 4.18: The stencil of the Preissmann scheme.

Though the choice $\theta = 1/2$ results in the most accurate scheme, in practice, however, one uses a larger value, such as θ=0.55, since the scheme is more stable as it produces some numerical dissipation.

Exercise 4.4.7 Derive the amplitude error of the Preissmann scheme.

4.4.3 Numerical solution of the 1D shallow water equations

So far we have considered the *linearized* shallow water equations. In this section we treat a numerical method for solving the 1D shallow water equations, Eqs. (4.1.1a)−(4.1.1b). The main difference is the inclusion of advection and bottom friction terms. They may have some impacts on the accuracy and stability of the numerical method. This will become obvious later.

We shall discuss the numerical method as implemented in the open source SWASH package (swash.sf.net). With this program both long and short waves can be simulated[3]. The numerical method is based upon the leapfrog scheme as outlined in Section 4.4.1. So, the scheme for the 1D shallow water equations may be written as follows

$$\frac{\zeta_m^{n+1} - \zeta_m^n}{\Delta t} + \frac{\hat{h}^n_{m+1/2} u^{n+1/2}_{m+1/2} - \hat{h}^n_{m-1/2} u^{n+1/2}_{m-1/2}}{\Delta x} = 0$$

$$\frac{u^{n+1/2}_{m+1/2} - u^{n-1/2}_{m+1/2}}{\Delta t} + \left(u \frac{\partial u}{\partial x} \right)^{n-1/2}_{m+1/2} + g \frac{\zeta^n_{m+1} - \zeta^n_m}{\Delta x} + \left(c_f \frac{u|u|}{h} \right)^{n+1/2}_{m+1/2} = 0$$

where both the advection and friction terms will be approximated later. The water depth $h = \zeta + d$ in the continuity equation is approximated at the previous time level, since it depends on the water level which is only known at that level. Furthermore, this water depth need to be evaluated at the velocity points $m - 1/2$ and $m + 1/2$. Since a staggered

[3]Simulation of short waves is only possible with the addition of non-hydrostatic pressure to the shallow water equations.

grid is employed, these values are missing at those points[4]. This is indicated by putting a hat on top of h. The missing values can be approximated by means of interpolation. An example would be a linear interpolation, as follows

$$\hat{h}^n_{m+1/2} = \frac{1}{2}\left(h^n_m + h^n_{m+1}\right)$$

This approximation, which is equivalent to central differences, is second order of accuracy. Hence, it does not influence the global error of the scheme. Another example is first order upwinding, as follows

$$\hat{h}^n_{m+1/2} = \begin{cases} h^n_m\,, & \text{if } u^n_{m+1/2} \geq 0 \\ \\ h^n_{m+1}\,, & \text{if } u^n_{m+1/2} < 0 \end{cases}$$

Although it is first order accurate, it can be a very useful approximation, in particular, when **flooding and drying** at shallow flats become important. This can be explained as follows. Let us assume $u > 0$ everywhere. The scheme for the continuity equation reads (recall $\partial d/\partial t = 0$)

$$\frac{h^{n+1}_m - h^n_m}{\Delta t} + \frac{h^n_m u^{n+1/2}_{m+1/2} - h^n_{m-1} u^{n+1/2}_{m-1/2}}{\Delta x} = 0$$

We rewrite this equation as follows

$$h^{n+1}_m = \left(1 - \frac{\Delta t}{\Delta x} u^{n+1/2}_{m+1/2}\right) h^n_m + \frac{\Delta t}{\Delta x} u^{n+1/2}_{m-1/2} h^n_{m-1}$$

Exercise 4.4.8 Verify this.

We assume by induction that $h^n_m \geq 0$ and $h^n_{m-1} \geq 0$. Then $h^{n+1}_m \geq 0$ if

$$\frac{\Delta t\, u^{n+1/2}_{m+1/2}}{\Delta x} \leq 1$$

By fulfilling this CFL restriction, we prevent the water depth h from being negative which is unwanted in the case of drying grid points. These grid points become inactive where $h = 0$. The physical implication of this is the ability to simulate the **runup** of long waves on the beaches, around the islands, etc.

The accuracy with which $\hat{h}^n_{m+1/2}$ is approximated can be enhance by using a flux limiting scheme, as follows

$$\hat{h}^n_{m+1/2} = \begin{cases} h^n_m + \frac{1}{2}\Psi(r^+_{m+1/2})(h^n_m - h^n_{m-1})\,, & \text{if } u^n_{m+1/2} \geq 0 \\ \\ h^n_{m+1} - \frac{1}{2}\Psi(r^-_{m+1/2})(h^n_{m+2} - h^n_{m+1})\,, & \text{if } u^n_{m+1/2} < 0 \end{cases}$$

[4]It is assumed that the bottom level d is defined in the water level point.

where $\Psi(r)$ is the flux limiter to avoid unwanted oscillations near sharp gradients, and

$$r^{+}_{m+1/2} = \frac{h^n_{m+1} - h^n_m}{h^n_m - h^n_{m-1}} \quad \text{and} \quad r^{-}_{m+1/2} = \frac{h^n_{m+1} - h^n_m}{h^n_{m+2} - h^n_{m+1}}$$

In this way, the positivity property of the water depth is preserved. The choices for flux limiters Ψ are many and often evenly appropriate. Most of them can be found in the literature, see Hirsch (1990) and Wesseling (2001).

Next we consider the approximation of the advection term $u\partial u/\partial x$. There are many options for this. Examples are

Central differences

$$\left(u\frac{\partial u}{\partial x}\right)^{n-1/2}_{m+1/2} \approx u^{n-1/2}_{m+1/2}\frac{u^{n-1/2}_{m+3/2} - u^{n-1/2}_{m-1/2}}{2\Delta x}$$

First order upwind

$$\left(u\frac{\partial u}{\partial x}\right)^{n-1/2}_{m+1/2} \approx \begin{cases} u^{n-1/2}_{m+1/2}\dfrac{u^{n-1/2}_{m+1/2} - u^{n-1/2}_{m-1/2}}{\Delta x}, & \text{if } u^{n-1/2}_{m+1/2} > 0 \\ \\ u^{n-1/2}_{m+1/2}\dfrac{u^{n-1/2}_{m+3/2} - u^{n-1/2}_{m+1/2}}{\Delta x}, & \text{if } u^{n-1/2}_{m+1/2} < 0 \end{cases}$$

BDF

$$\left(u\frac{\partial u}{\partial x}\right)^{n-1/2}_{m+1/2} \approx \begin{cases} u^{n-1/2}_{m+1/2}\dfrac{3u^{n-1/2}_{m+1/2} - 4u^{n-1/2}_{m-1/2} + u^{n-1/2}_{m-3/2}}{2\Delta x}, & \text{if } u^{n-1/2}_{m+1/2} > 0 \\ \\ u^{n-1/2}_{m+1/2}\dfrac{3u^{n-1/2}_{m+5/2} - 4u^{n-1/2}_{m+3/2} + u^{n-1/2}_{m+1/2}}{2\Delta x}, & \text{if } u^{n-1/2}_{m+1/2} < 0 \end{cases}$$

Exercise 4.4.9 Verify these finite difference formulas.

Exercise 4.4.10 Write down the QUICK scheme for the advection term.

Sometimes central differences are favourable and sometimes an upwind scheme is a better choice. This choice depends, of course, on the application. One may think of an application in which there is no physical dissipation (bottom friction, turbulence, etc.). In such a case the BDF scheme would be a good option, since numerical results have always shown good stability properties when some numerical dissipation is involved.

The bottom friction term $c_f u|u|/h$ is a nonlinear, dissipation term. Practically, it is always convenience to approximate a dissipation term *implicitly*, because this will enhance the stability.

Exercise 4.4.11 Give an explanation for this.

However, the bottom friction term is nonlinear which implies **linearization** in order to find the solution. Consider a nonlinear term u^2 that needs to be approximated at time level $n+1$. Two commonly used linearization techniques are the following (see Appendix A.3).

Picard linearization

$$\left(u^{n+1}\right)^2 \approx u^n\, u^{n+1}$$

Newton linearization

$$\left(u^{n+1}\right)^2 \approx 2u^n\, u^{n+1} - \left(u^n\right)^2$$

They are first order and second order accurate in time, respectively. In SWASH, the bottom friction term is approximated as follows

$$\left(c_f \frac{u|u|}{h}\right)_{m+1/2}^{n+1/2} \approx c_f \frac{|u_{m+1/2}^{n-1/2}|u_{m+1/2}^{n+1/2}}{\overline{h}_{m+1/2}^{n}}$$

with

$$\overline{h}_{m+1/2}^{n} = \frac{1}{2}\left(h_m^n + h_{m+1}^n\right)$$

This completes our description of how the different terms in the shallow water equations are handled. The considered scheme is conditionally stable and the stability limit is given by

$$\frac{\left(\sqrt{g\overline{h}_{m+1/2}^{n+1}} + |u_{m+1/2}^{n+1/2}|\right)\Delta t}{\Delta x} \leq 1 \qquad (4.4.1)$$

Exercise 4.4.12 Verify this.

We may wonder how could it be possible to add some dissipation to the momentum equation, like bottom friction and numerical diffusion due to the advection approximation, while the leapfrog scheme has been applied. The answer is that the two unknowns ζ and u leap frog through time, which makes the approximations second order accurate if *both* continuity and momentum equations are considered. If we just consider the momentum equation with advection and bottom friction terms *only*, then the time integration is actually explicit Euler and implicit Euler, respectively. This implies that adding some dissipation to this equation is yet favourable. Another consequence is that they are first order accurate in time. This might be serious, in particular, when the advection term is involved. A common way to improve this is to apply the **MacCormack** method which is a second order accurate predictor-corrector technique.

A final remark. SWASH is designed for the simulation of short waves approaching a shore. The associated time scales are of a few seconds. Because of accuracy we need to choose a time step of a few orders of magnitude less than the time scale. Because of this, the CFL limit, Eq. (4.4.1), is often not a severe one. An example. Suppose we want to simulate a harmonic wave of 10 seconds in a basin with a depth of 20 m. According to the linear dispersion relationship

$$\omega^2 = gk \tanh kh$$

the wave length is approximately 60 m. On accuracy grounds, we choose at least 50 grid points per wave length. Let us say 60 grid points and so, $\Delta x = 1$ m. By taking $\sqrt{gh}$ as a save upper limit for the phase velocity ("long waves propagate faster than short waves"), we conclude that the maximum time step is 0.07 s, which is reasonable in view of the considered time scale. Therefore an explicit scheme is a good choice for the simulation of short waves. Though it is important that this explicit scheme should not have an amplitude error, which is damaging for short waves. That is why the leapfrog scheme has been chosen in SWASH.

Long waves is another issue. A typical time scale of a tidal wave is of tens of hours. This allow us to choose a much larger time step compared to that of the short waves, but still the CFL limit, Eq. (4.4.1), is acceptable.

Exercise 4.4.13 Consider an M_2 tidal component. Its period is about 12 hours and 25 minutes. Let us consider a tidal basin with a depth of 20 m. Compute the maximum time step Δt according to the CFL limit, Eq. (4.4.1), if we require at least 30 grid points per wave length. Conclusion?

Nevertheless, there are applications for which the best choice would be an implicit scheme. Examples are the real time forecasting of water levels along the coasts and the early warning systems for tsunami waves. For those applications a large time step is desired so that implicit schemes are in the end more cheaper than explicit ones. A popular implicit scheme for solving 2D shallow water equations is the so-called **alternating direction implicit** (ADI) difference scheme. This has been implemented in e.g. WAQUA of Rijkswaterstaat (www.waqua.nl) and Delft3D-FLOW of Deltares (oss.deltares.nl). Another example is a semi-implicit approach based on the $\theta-$scheme as implemented in e.g. SWASH (an alternative to the leapfrog scheme).

4.5 Final remarks

In this chapter we have treated some basic concepts of the numerical solution of the shallow water equations. These hyperbolic equations are the governing ones for the water movement in coastal areas. Specific properties of these equations have been briefly discussed using the method of characteristics. They give some useful insights with respect to the boundary conditions for a given shallow water system (e.g. tidal basin). In addition, the choice of these boundary conditions plays a very important role as it determines the well posedness of the boundary value problem. In particular, the interplay between initial and boundary conditions in a tidal basin is a prominent example of a typical *model behaviour* exhibiting a spin-up effect. Another example is the occurence of artificial backwater in a river. Engineers or end-users should be aware of these typical modelling phenomena. So it is important to try to get insight into an unusually observed model behaviour, and to get some knowledge on the behaviour and properties of the considered physical system. In this chapter we have seen how this can be realized by simplifying the governing equations and try to analyze them.

Advance numerical models like SWASH and Delft3D-FLOW contain a lot of physical and numerical parameters, options for boundary conditions and some other functionalities, which make these models very complicated in use. Also, some of these options or a combination thereof may not be tested properly. Another issues are the bugs or errors in computer programs, and possibly inadequate numerical approximations. Therefore a good (basic) knowledge on numerical models is of crucial importance. So, at least, the possibilities and limitations of these models must be appreciated. Also, one should try to prevent the models to be use as a black box. What helps is being critical in the use of numerical models.

Typically, using a model consists of pre-, main and post-processing parts. During the **pre-processing**, the user is asked for setting up the input files, preparing topological schematizations (e.g. grid, bathymetry), and specifying the boundary conditions. This is actually the most important step since during this step one need to think and to understand what he/she want (goal) and how to manage (realization). Aspects that need some attention are the dimensions (domain size, simulation time, spin-up, time step, grid size, etc. based on space- and time scales) and the boundary conditions (to require well posedness of the problem).

Next, the numerical model will take care of the **main processing**, i.e. reading the data from input files, solving the necessary equations and writing the model outcomes to output files. Although the model will take care of this part, the user or engineer is still responsible to understand what the model is doing. He or she has to be sure which and how the numerical processes are to be undertaken and whether they have been done in a correct manner.

Finally, the user is responsible for the **post-processing** part which means that the usually huge model data, as stored in output files, need to be post processed so that the model outcomes are represented as graphs or animations. This part generally requests additional software to transform data into graphs or animations. Also during this stage the user must be critical. He or she should never trust the model results blindly, even when they look nice and smooth. The results must always be checked whether they correspond to the expected ones or not (e.g. from theory or experiment).

Appendix A

Taylor series

A.1 Taylor series in one dimension

The Taylor series is mainly used for approximating functions when one can identify a small parameter. This series is a representation of a function as a sum of terms which are derived from the derivatives of the function in a single point, and provides an approximation of the function in the neighbourhood of that point. This Taylor series is also called a Taylor expansion. A well-known example is the following. For small x we get an expansion of the function e^x as follows

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \cdots$$

To explain this expansion, let us assume that e^x can be described by the following power series

$$e^x = \sum_{n=0}^{\infty} a_n x^n = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots$$

When we consider the point $x = 0$, then it follows that $a_0 = 1$. On the other hand, the derivative of e^x is itself. Hence, differentiating the series with respect to x one time, we have

$$e^x = \sum_{n=1}^{\infty} a_n x^n = a_1 + 2a_2 x + 3a_3 x^2 + \cdots$$

Apparently, $2a_2 = a_1 = a_0 = 1$, whereas $3a_3 = a_2$, etc. Hence, $a_0 = a_1 = 1$, $a_2 = 1/2$ and $a_3 = 1/6$. Differentiating the series once more, we find the following relations $6a_3 = a_1$, $12a_4 = a_2$, etc. and thus, $a_4 = 1/24$. Differentiating by x the series n times yields the following relation

$$a_n = \frac{1}{n!}$$

In other words, the function e^x is given by the following series

$$e^x = \sum_{n=0}^{\infty} \frac{1}{n!} x^n$$

which is approximately correct around the point $x = 0$. One can prove that this series will converge in a restricted neighbourhood of the point $x = 0$.

The above expansion can also be applied to many other functions, e.g. $\sin(x)$, $\ln(x)$ and $(1+x)^r$. A general formula of this expansion for any function $f(x)$ in the neighbourhood of $x = 0$ is the following power series, the so-called Maclaurin series,

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} x^n$$

where $f^{(n)}$ denotes the $n-$th derivative of f evaluated at the point $x = 0$. The Taylor series involves expanding about a point $x = a$, instead of about zero as in the Maclaurin series. The form of the Taylor series is simply

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!} (x-a)^n$$

It is assumed that $f(x)$ is infinitely differentiable at $x = a$. We call this function well behaved.

As an example we expand the function $\sqrt{x}$ about $x = 1$ till the third degree as follows.

$$f(1) = 1\,, \quad f'(1) = \frac{1}{2}\,, \quad f''(1) = -\frac{1}{4}\,, \quad f'''(1) = \frac{3}{8}$$

so that we have

$$\sqrt{x} = 1 + \frac{1}{2}(x-1) - \frac{1}{8}(x-1)^2 + \frac{1}{16}(x-1)^3 + R$$

with R a remainder term which is supposed to be sufficiently small around the point $x = 1$. One can prove that this series is valid for the interval $0 < x < 2$.

The Taylor series can also be presented in a different way, as follows

$$f(x+h) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x)}{n!} h^n$$

with h a small parameter. Again, we assume that $f(x)$ is smooth enough at the point x.

Given a well-behaved function $y(t)$ with t as the independent variable representing time and a small time step Δt we can bring about the following Taylor expansion

$$y(t + \Delta t) = y(t) + \Delta t y'(t) + \frac{1}{2}\Delta t^2 y''(t) + \cdots + \frac{\Delta t^p}{p!} y^{(p)}(t) + R_p(t) \qquad \text{(A.1.1)}$$

with R_p the remainder term given by

$$R_p(t) = \frac{\Delta t^{(p+1)}}{(p+1)!} y^{(p+1)}(\xi)\,, \quad t < \xi < t + \Delta t$$

Let us *truncate* the Taylor polynomial till the first degree, i.e. $p = 1$,

$$y(t + \Delta t) = y(t) + \Delta t y'(t) + R_1(t)$$

Next we derive an expression for the first derivative

$$y'(t) = \frac{y(t + \Delta t) - y(t)}{\Delta t} - \frac{R_1(t)}{\Delta t}$$

Hence, we can approximate the first derivative by evaluating the following expression which is computable[1]

$$y'(t) \approx \frac{y(t + \Delta t) - y(t)}{\Delta t}$$

assuming that R_1 is small enough. With this approximation we are making an error which is called the **truncation error**. This error is given by

$$\tau_{\Delta t} = \frac{R_1(t)}{\Delta t} = \frac{1}{2} y''(\xi) \Delta t\,, \quad t < \xi < t + \Delta t$$

The truncation error is often notated using the $\mathcal{O}-$symbol of Landau

$$\tau_{\Delta t} = \mathcal{O}(\Delta t)$$

which states that the truncation error is proportional to the time step, assuming that this time step is sufficiently small. We often say that the above derived approximation is first order accurate in time, and we can write this as follows

$$\frac{y(t + \Delta t) - y(t)}{\Delta t} = y'(t) + \mathcal{O}(\Delta t)$$

and thus

$$\tau_{\Delta t} = \frac{y(t + \Delta t) - y(t)}{\Delta t} - y'(t) = \mathcal{O}(\Delta t) \tag{A.1.2}$$

This formula uses the **forward difference** to approximate the derivative. We can derive the same expressions using the **backward difference**

$$\tau_{\Delta t} = \frac{y(t) - y(t - \Delta t)}{\Delta t} - y'(t) = \mathcal{O}(\Delta t)$$

and the **central difference**

$$\tau_{\Delta t} = \frac{y(t + \Delta t) - y(t - \Delta t)}{2\Delta t} - y'(t) = \mathcal{O}(\Delta t^2)$$

[1]A problem is computable if it can be solved on a digital computer by some algorithm. In our context, we can evaluate expressions, like finite differences, on the computer using simple operations like addition, substraction, multiplication and division.

Usually the derivative is one of the term in the differential equation that we wish to solve. Let us consider, for instance, the test equation

$$\frac{dy}{dt} = \lambda y \tag{A.1.3}$$

with λ a real parameter. We may replace this equation by the following approximation

$$\frac{y(t+\Delta t) - y(t)}{\Delta t} = \lambda y(t) \tag{A.1.4}$$

which is known as the **forward Euler** method. Keep in mind that $y(t)$ is a continuous, well-behaved function and is an exact solution of Eq. (A.1.3) but *not the solution of* Eq. (A.1.4). The truncation error of the approximation is given by

$$\tau_{\Delta t} = \frac{y(t+\Delta t) - y(t)}{\Delta t} - \lambda y(t) \overset{\text{Eq. }(A.1.3)}{\overset{\downarrow}{=}} \frac{y(t+\Delta t) - y(t)}{\Delta t} - y'(t) \overset{\text{Eq. }(A.1.2)}{\overset{\downarrow}{=}} \mathcal{O}(\Delta t)$$

Alternatively, substitution of Eq. (A.1.1) into Eq. (A.1.4) yields the same result. However, the manner in which we evaluate the right-hand side of Eq. (A.1.3) is another way to control the truncation error. Let us integrate the test equation on an interval $[t, t+\Delta t]$

$$\int_t^{t+\Delta t} y'\,dt = y(t+\Delta t) - y(t) = \lambda \int_t^{t+\Delta t} y\,dt$$

We may approximate the integral in the right-hand side by means of the trapezoidal rule

$$\int_t^{t+\Delta t} y\,dt \approx \frac{1}{2}\Delta t\,[y(t) + y(t+\Delta t)]$$

Hence, we can apply the following formula to solve approximately the test equation

$$\frac{y(t+\Delta t) - y(t)}{\Delta t} = \frac{1}{2}\lambda\,[y(t) + y(t+\Delta t)]$$

which is known as the **Crank-Nicolson** method or, obviously, the trapezoidal rule. The truncation error can be calculated using the Taylor expansion for $y(t+\Delta t)$, Eq. (A.1.1), which yields

$$\tau_{\Delta t} = \frac{y(t+\Delta t) - y(t)}{\Delta t} - \frac{1}{2}\lambda\,[y(t) + y(t+\Delta t)] = \mathcal{O}(\Delta t^2)$$

and hence, the trapezoidal rule is second order accurate in time. Finally, we can also apply the following approximation

$$\frac{y(t) - y(t-\Delta t)}{\Delta t} = \lambda y(t)$$

or

$$\frac{y(t+\Delta t) - y(t)}{\Delta t} = \lambda y(t+\Delta t)$$

which is known as the **backward Euler** method and its associated truncation error is given by

$$\tau_{\Delta t} = \frac{y(t) - y(t-\Delta t)}{\Delta t} - \lambda y(t) = \frac{y(t) - y(t-\Delta t)}{\Delta t} - y'(t) = \mathcal{O}(\Delta t)$$

A.2 Taylor series in two dimensions

In the previous section we have considered functions of one independent variable. In this section we demonstrate how we can expand a function of two independent variables. We consider a function $f(x,y)$ which is continuously differentiable at the point (x,y). To derive the Taylor series, we first expand the function with respect to x and then with respect to y around point (x,y). Hence,

$$f(x+h,y+k) = f(x,y+k) + f_x(x,y+k)h + \frac{1}{2!}f_{xx}(x,y+k)h^2 + \frac{1}{3!}f_{xxx}(x,y+k)h^3 + \cdots$$

where the subscript x denotes differentation with respect to x, i.e. f_x is the first derivative of f to x, f_{xx} is the second derivative of f to x, etc. Next we expand each term with respect to y, as follows

$$\begin{aligned} f(x+h,y+k) &= f(x,y) + f_y(x,y)k + \frac{1}{2!}f_{yy}(x,y)k^2 + \frac{1}{3!}f_{yyy}(x,y)k^3 + \cdots \\ &+ f_x(x,y)h + f_{xy}(x,y)hk + \frac{1}{2!}f_{xyy}(x,y)hk^2 + \frac{1}{3!}f_{xyyy}(x,y)hk^3 + \cdots \\ &+ \frac{1}{2!}f_{xx}(x,y)h^2 + \frac{1}{2!}f_{xxy}(x,y)h^2k + (\frac{1}{2!})^2 f_{xxyy}(x,y)h^2k^2 + \cdots \\ &+ \frac{1}{3!}f_{xxx}(x,y)h^3 + \frac{1}{3!}f_{xxxy}(x,y)h^3k + \frac{1}{2!3!}f_{xxxyy}(x,y)h^3k^2 + \cdots \end{aligned}$$

Note that a number of mixed derivatives have been appeared. Finally, we regroup the terms having the same degree (the sum of the exponents is constant for each term within a group), as follows

$$\begin{aligned} f(x+h,y+k) &= f(x,y) + f_x(x,y)h + f_y(x,y)k \\ &+ \frac{1}{2!}f_{xx}(x,y)h^2 + f_{xy}(x,y)hk + \frac{1}{2!}f_{yy}(x,y)k^2 \\ &+ \frac{1}{3!}f_{xxx}(x,y)h^3 + \frac{1}{2!}f_{xxy}(x,y)h^2k + \frac{1}{2!}f_{xyy}(x,y)hk^2 + \frac{1}{3!}f_{yyy}(x,y)k^3 + \cdots \end{aligned}$$

We can write the derived expansion in a different way as

$$\begin{aligned} f(x+h,y+k) = f(x,y) \quad &+ \left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right] f(x,y) + \frac{1}{2!}\left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right]^2 f(x,y) \\ &+ \frac{1}{3!}\left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right]^3 f(x,y) + \frac{1}{4!}\left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right]^4 f(x,y) + \cdots \end{aligned}$$

where we have made use of the well-known binomial formula of Newton

$$(p+q)^n = \binom{n}{0}p^n + \binom{n}{1}p^{n-1}q + \binom{n}{2}p^{n-2}q^2 + \cdots + \binom{n}{n-1}pq^{n-1} + \binom{n}{n}q^n$$

with the so-called binomial coefficient

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

These coefficients are easily obtained from the $n-$th row of Pascal's triangle

$$\begin{array}{lccccccccc}
n=0: & & & & & 1 & & & & \\
n=1: & & & & 1 & & 1 & & & \\
n=2: & & & 1 & & 2 & & 1 & & \\
n=3: & & 1 & & 3 & & 3 & & 1 & \\
n=4: & 1 & & 4 & & 6 & & 4 & & 1
\end{array}$$

For instance,

$$\left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right]^2 f(x,y) = f_{xx}(x,y)h^2 + 2f_{xy}(x,y)hk + f_{yy}(x,y)k^2$$

and

$$\left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right]^3 f(x,y) = f_{xxx}(x,y)h^3 + 3f_{xxy}(x,y)h^2k + 3f_{xyy}(x,y)hk^2 + f_{yyy}(x,y)k^3$$

etc.

Thus, the general formula for the Taylor series in two dimensions reads

$$f(x+h, y+k) = \sum_{n=0}^{\infty} \frac{1}{n!}\left[h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right]^n f(x,y)$$

A.3 Linearization

Equations describing a physical phenomena are often nonlinear. These equations are usually hard to solve and difficult to analyze. To dealt with this issue, we commonly linearize the nonlinear equations. The procedure is to apply Taylor expansion for the underlying equations with respect to the *dependent variables*, while retaining only the zero and first order terms.

We consider the following momentum equation

$$\frac{\partial u}{\partial t} + g\frac{\partial \zeta}{\partial x} + c_f\frac{u|u|}{h} = 0$$

with u the flow velocity, ζ the water level, $h = \zeta + d$ the water depth, and c_f the dimensionless friction coefficient due to bottom roughness. Here we assume that c_f is constant. This equation contains one nonlinear term, namely the bottom friction term. It is nonlinear in the dependent variables ζ and u. To linearize this equation, we first assume that there is a small variation of ζ and u on top of a uniform flow. The small variations of these variables are denoted by ζ' and u', respectively. The water level and flow velocity of the uniform flow is given by ζ_0 and u_0, respectively, and so we have $\zeta = \zeta_0 + \zeta'$ and $u = u_0 + u'$. Obviously, the uniform flow is the solution of the momentum equation as well,

$$\frac{\partial u_0}{\partial t} + g\frac{\partial \zeta_0}{\partial x} + c_f \frac{u_0|u_0|}{\zeta_0 + d} = 0$$

Next we expand the nonlinear bottom friction term, which is a function of ζ and u, as given by

$$f(\zeta, u) = c_f \frac{u|u|}{\zeta + d}$$

around the uniform flow, i.e. (ζ_0, u_0), as follows

$$\begin{aligned} f(\zeta_0 + \zeta', u_0 + u') &\approx f(\zeta_0, u_0) + \frac{\partial f(\zeta_0, u_0)}{\partial \zeta}\zeta' + \frac{\partial f(\zeta_0, u_0)}{\partial u}u' \\ &= c_f \frac{u_0|u_0|}{\zeta_0 + d} - c_f \frac{u_0|u_0|}{(\zeta_0 + d)^2}\zeta' + 2c_f \frac{|u_0|}{\zeta_0 + d}u' \end{aligned}$$

where we have taken the derivative of $u|u|$ as $2|u|$. Note that only zero and first order terms have been taken into account.

The complete momentum equation is then given by

$$\frac{\partial u_0}{\partial t} + \frac{\partial u'}{\partial t} + g\frac{\partial \zeta_0}{\partial x} + g\frac{\partial \zeta'}{\partial x} + c_f \frac{u_0|u_0|}{\zeta_0 + d} - c_f \frac{u_0|u_0|}{(\zeta_0 + d)^2}\zeta' + 2c_f \frac{|u_0|}{\zeta_0 + d}u' = 0$$

and dropping the zero order terms yields

$$\frac{\partial u'}{\partial t} + g\frac{\partial \zeta'}{\partial x} - c_f \frac{u_0|u_0|}{(\zeta_0 + d)^2}\zeta' + 2c_f \frac{|u_0|}{\zeta_0 + d}u' = 0$$

which is a linear momentum equation containing only first order terms.

Alternatively, we may expand the bottom friction with respect to the flow velocity only. The linearized bottom friction becomes

$$c_f \frac{u|u|}{h} \approx c_f \frac{u_0|u_0|}{h} + 2c_f \frac{|u_0|}{h}u' = c_f \frac{|u_0|}{h}\left(u_0 + 2u'\right)$$

With $u' = u - u_0$, we have the following linearized momentum equation

$$\frac{\partial u}{\partial t} + \cdots + 2c_f \frac{|u_0|}{h}u - c_f \frac{u_0|u_0|}{h} = 0$$

(We have dropped the water level gradient for the moment as it is not relevant anymore.) This type of linearization is often called the **Newton linearization** and is mostly used in conjunction with time integration. In this context, u_0 is usually evaluated at the previous time step. For instance, time integration of the resulted equation is based on the following one-step scheme

$$\frac{u^{n+1}-u^n}{\Delta t} + \cdots + 2c_f \frac{|u^n|}{h^n} u^{n+1} - c_f \frac{u^n|u^n|}{h^n} = 0$$

with Δt the time step and n the time level. Note that the water depth h is evaluated at the previous time step to keep the bottom friction term be linear. Due to the linearization we are able to compute the velocity at the next time step

$$\left[\frac{1}{\Delta t} + 2c_f \frac{|u^n|}{h^n}\right] u^{n+1} + \cdots = \frac{u^n}{\Delta t} + c_f \frac{u^n|u^n|}{h^n}$$

Another way of linearizing the equation is by means of the well-known **Picard iteration** or the fixed-point method. This method is also closely connected with time integration. Let us consider the momentum equation with the relevant terms,

$$\frac{\partial u}{\partial t} + \cdots + c_f \frac{u|u|}{h} = 0$$

which is integrated in time using the backward Euler method

$$\frac{u^{n+1}-u^n}{\Delta t} + \cdots + c_f \frac{u^{n+1}|u^{n+1}|}{h^n} = 0$$

This is a *nonlinear* recurrent relation for the unknown u^{n+1}. We introduce a Picard iteration with k as iteration counter. A typical linearization of the bottom friction term in iteration k is to use the previously computed $u^{n+1,k-1}$ in the term $|u|$. The unknown $u^{n+1,k}$ is then obtained from the following *linear* recurrent relation

$$\frac{u^{n+1,k}-u^n}{\Delta t} + \cdots + c_f \frac{u^{n+1,k}|u^{n+1,k-1}|}{h^n} = 0$$

The initial guess for the Picard iteration at this time level can be taken as the solution at the previous time step, i.e. $u^{n+1,0} = u^n$. In many cases one Picard iteration is sufficient ($k = 1$). Hence,

$$\frac{u^{n+1}-u^n}{\Delta t} + \cdots + c_f \frac{u^{n+1}|u^n|}{h^n} = 0$$

This final linearization is also known as the **Picard** linearization.

Appendix B

Fourier series

B.1 Complex exponential functions

Any *linear*, homogeneous, ordinary differential equation with *constant coefficients* can be solved by substituting an **exponential** function. An example is the following test equation

$$\frac{dy}{dt} = \lambda y$$

which has the solution

$$y(t) = y_0 e^{\lambda t}$$

where λ is a *real* parameter and y_0 is a constant determined by the initial condition. Note that the physical constraints are such that λ must be real and negative, i.e. $\lambda < 0$. Also note that the independent variable t usually represents time.

The test equation is an example of a first order differential equation. In case of a second or higher order, linear, homogeneous ODE with real but constant coefficients, the solution is still of the exponential form but this time with an exponent being a complex number or purely imaginary. The next example illustrates this. We consider the following second order ODE

$$\frac{d^2y}{dt^2} + y = 0$$

If we define $x = dy/dt$, then we can rewrite this second order ODE as a coupled system of two first order ODEs, as follows

$$\frac{dx}{dt} = -y$$

$$\frac{dy}{dt} = \quad x$$

or, in the matrix-vector form

$$\frac{d}{dt}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}$$

Since the coefficients of the second order differential equations are real, which is always the case, the matrix elements are real too. We can find the eigenvalues of the system, as follows

$$\det\begin{pmatrix} -\lambda & -1 \\ 1 & -\lambda \end{pmatrix} = \lambda^2 + 1 = 0$$

giving

$$\lambda_{1,2} = \pm \mathrm{i}$$

The eigenvalues are thus purely imaginary and the uncoupled system is given by

$$\frac{d}{dt}\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}\begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

whereas the solution can be found through

$$\begin{pmatrix} x \\ y \end{pmatrix} = [\vec{e}_1\ \vec{e}_2]\begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

with $\vec{e}_1$ and $\vec{e}_2$ the eigenvectors of the system. By virtue of the linearity[1], the general solution of the second order ODE is given by

$$y(t) = \alpha e^{-\mathrm{i}t} + \beta e^{\mathrm{i}t}$$

which can be checked by substitution. The values for α and β can be found by means of the initial conditions.

Next we make use of the well-known Euler's formula

$$e^{\mathrm{i}\phi} = \cos\phi + \mathrm{i}\,\sin\phi$$

for any real number ϕ. So, based on this formula, any exponential function with a purely imaginary exponent $\mathrm{i}\phi$ is a complex number with a real part ($= \cos\phi$) and an imaginary part ($= \sin\phi$). The complex solution is thus given by

$$y(t) = \mu\cos t + \nu\mathrm{i}\,\sin t$$

with $\mu = \alpha + \beta$ and $\nu = \beta - \alpha$. We obtain a real-valued solution by simply taking the real part or the imaginary part of the complex solution. This is correct if the coefficients of the second order ODE are real. The solution, either $y(t) = \mu\cos t$ or $y(t) = \nu\sin t$, represents

[1]A linear combination of solutions of a linear ODE is a solution of the equation as well. This property is also known as the principle of superposition.

a periodic motion with a constant amplitude.

Next we consider the following ODE

$$\frac{d^2y}{dt^2} + 2\frac{dy}{dt} + 2y = 0$$

The associated **characteristic equation** is given by

$$\lambda^2 + 2\lambda + 2 = 0$$

and the eigenvalues are

$$\lambda_{1,2} = -1 \pm \mathrm{i}$$

This time the eigenvalues are complex. The general solution is given by

$$y(t) = \alpha e^{-t}\, e^{-\mathrm{i}t} + \beta e^{-t}\, e^{\mathrm{i}t}$$

and α and β are determined by the initial conditions. To obtain a real-valued solution we consider the real part only. Hence

$$y(t) = \mu e^{-t}\, \cos t$$

This is a harmonic motion with a decreasing amplitude. This decay is due to the real part of the eigenvalues. From a physical point of view, this real part must be negative, i.e. $Re(\lambda) < 0$. Thus in any complex solution we can recognize a **damping** effect and an **oscillatory** effect. They are associated with the real part and imaginary part of the eigenvalues, respectively.

The theory discussed so far can be applied to *partial* differential equations as well. In our context, we include the spatial scale x as an independent variable. Hence, we restrict ourselves to one-dimensional cases. A general solution of a *linear* PDE with *constant coefficients* is given by

$$e^{\lambda t + \mu x} \tag{B.1.1}$$

with λ and μ the complex numbers. The nature of the solution depends on these numbers, which in turn depend on the constant coefficients in PDE. This is demonstrated with the following two examples.

The first example deals with a linear convection equation

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0\,, \quad -\infty < x < \infty$$

This equation describes the propagation of any disturbance $c(x,t)$ with a constant speed u (in m/s). Substituting the solution, Eq. (B.1.1) in the equation leads to the following equation

$$\lambda e^{\lambda t + \mu x} + u\mu e^{\lambda t + \mu x} = 0$$

or

$$\lambda + u\mu = 0$$

Since the solution characterizes the propagation of disturbance without *deforming*, both λ and μ must be purely imaginary. So, we may write $\lambda = -\omega\mathrm{i}$ and $\mu = k\mathrm{i}$, and we have

$$c(x,t) = e^{\mathrm{i}(kx-\omega t)}$$

and

$$\frac{\omega}{k} = u$$

A real-valued solution is given by

$$c(x,t) = \cos(kx - \omega t)$$

which describes a **progressive** wave. This can be justified as follows. Let us consider a small time span Δt. During this period the wave propagates over a distance Δx with a constant speed u. Hence,

$$\frac{\Delta x}{\Delta t} = u = \frac{\omega}{k} \quad \Rightarrow \quad k\Delta x - \omega \Delta t = 0$$

Using the identity $\cos(x+y) = \cos x \, \cos y - \sin x \, \sin y$, we have

$$\begin{aligned} c(x+\Delta x, t+\Delta t) &= \cos(kx + k\Delta x - \omega t - \omega\Delta t) \\ &= \cos(kx-\omega t)\cos(k\Delta x - \omega\Delta t) - \sin(kx-\omega t)\sin(k\Delta x - \omega \Delta t) \\ &= \cos(kx - \omega t) \\ &= c(x,t) \end{aligned}$$

Hence, any position in the moving wave has a constant phase $kx - \omega t$. Obviously, the propagation speed of this position is called the **phase speed** and equals $u = \omega/k$. A harmonic wave has a wave length L and a wave period T, and $u = L/T$. Therefore $\omega = 2\pi/T$ is called the **radian frequency** and $k = 2\pi/L$ is called the **wave number**.

Another example considers a linear diffusion equation

$$\frac{\partial T}{\partial t} - \kappa \frac{\partial^2 T}{\partial x^2} = 0$$

This equation describes the spatially spreading of heat (T is temperature) in a rod with length L as time proceeds. At the boundaries, $x = 0$ and $x = L$, we assume a homogeneous Dirichlet condition, i.e. $T(0,t) = T(L,t) = 0$. The rate of spreading is determined by a constant diffusion coefficient $\kappa > 0$ (in m^2/s). Substituting the solution, Eq. (B.1.1) in the equation leads to the following equation

$$\lambda e^{\lambda t + \mu x} - \kappa \mu^2 e^{\lambda t + \mu x} = 0$$

or

$$\lambda - \kappa\mu^2 = 0$$

Thus a general solution is given by

$$T(x,t) = e^{\kappa\mu^2 t + \mu x} = e^{\kappa\mu^2 t}\, e^{\mu x}$$

Since the solution represents spreading as time increases, the first exponent must be real and negative. In addition, μ must have the dimension m^{-1}. An obvious choice would be $\mu = \mathrm{i}k = \mathrm{i}\,2\pi/L$. Hence, a general solution is

$$T(x,t) = e^{-\kappa k^2 t}\, e^{\mathrm{i}kx}$$

whereas a real-valued solution is

$$T(x,t) = e^{-\kappa k^2 t}\, \sin(kx)$$

which satisfies the boundary conditions.

B.2 Fourier series on a finite interval

In the previous section we have derived a general solution for a linear diffusion equation. The question arises whether there are more possible solutions, since a multiple of the argument of sine, i.e. kx, is allowed. In fact

$$T(x,t) = e^{-\kappa n^2 k^2 t}\, \sin(nkx)$$

is another general solution. (Verify this!) Here, $n = 1, 2, 3, \cdots$ and is arbitrary. However, the diffusion equation is linear and from the principle of superposition we know that any linear combination of the solutions also satisfies the equation. Consequently, the series

$$T(x,t) = \sum_{n=1}^{\infty} c_n e^{-\kappa n^2 k^2 t}\, \sin(nkx)$$

is the general solution of the considered equation. Here, the coefficients c_n are determined by the initial condition, which is given by

$$T(x,0) = f(x)\,, \quad 0 \le x \le L$$

with f a given function. Hence, to *satisfy* the initial condition we must have

$$f(x) = \sum_{n=1}^{\infty} c_n\, \sin(nkx) \tag{B.2.1}$$

If somehow we are able to determine the coefficients c_n and the infinite series converges, then we have solved the boundary value problem.

Eq. (B.2.1) can be interpreted as follows. A given function $f(x)$ defined on $0 \leq x \leq L$ can be expanded as a series of sines, *if* it converges. It turns out that it is possible to write an arbitrary function $f(x)$ on an interval $0 \leq x \leq L$ as

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos \frac{2n\pi x}{L} + b_n \sin \frac{2n\pi x}{L} \right) \tag{B.2.2}$$

given that this series converges. This series is called the **Fourier series** of $f(x)$, and consists of the so-called **Fourier components**.

To determine the so-called **Fourier coefficients** a_n and b_n, the following two properties can be employed.

- The function $f(x)$ is *periodic* with L/n.
- The sine and cosine functions form an *orthogonal* set of functions on the interval $0 \leq x \leq L$.

This yields

$$a_n = \frac{2}{L} \int_0^L f(x) \cos \frac{2n\pi x}{L} \, dx \,, \quad n = 0, 1, 2, \cdots$$

and

$$b_n = \frac{2}{L} \int_0^L f(x) \sin \frac{2n\pi x}{L} \, dx \,, \quad n =, 1, 2, \cdots$$

If $f(x)$ is a real function (which is usually the case), then the coefficients a_n and b_n are real as well. Moreover, in general, these coefficients decrease as n increases. The rate at which this is going to happen depends on the smoothness of $f(x)$. The smoother the function, the faster the Fourier coefficients converge to zero. Hence, a smoother function may be approximated well with just a few Fourier components.

For more details, see e.g. Andrews (1986).

We now use the formula above to give a Fourier series expansion of a simple function. Consider a block wave

$$f(x) = \begin{cases} 1 \,, & 0 \leq x < \dfrac{L}{2} \\ 0 \,, & \dfrac{L}{2} \leq x \leq 1 \end{cases}$$

This function is periodic with L, i.e. $f(x + mL) = f(x)$, for $-\infty < x < \infty$ and $m \in \mathbb{Z}$. The Fourier coefficients are then given by

$$a_0 = \frac{2}{L} \int_0^L f(x) \, dx = \frac{2}{L} \int_0^{L/2} dx = 1$$

$$a_n = \frac{2}{L} \int_0^L f(x) \cos \frac{2n\pi x}{L} \, dx = \frac{2}{L} \int_0^{L/2} \cos \frac{2n\pi x}{L} \, dx = \frac{1}{n\pi} \left[\sin \frac{2n\pi x}{L} \right]_0^{L/2} = 0 \,, \quad n \geq 1$$

$$b_n = \frac{2}{L}\int_0^L f(x)\,\sin\frac{2n\pi x}{L}\,dx = \frac{2}{L}\int_0^{L/2} \sin\frac{2n\pi x}{L}\,dx = \frac{1}{n\pi}\,[1-\cos(n\pi)] = \begin{cases} \dfrac{2}{n\pi}\,, & n = 1,3,5,\cdots \\ 0\,, & n = 2,4,6,\cdots \end{cases}$$

The Fourier series for the block wave becomes

$$f(x) = \frac{1}{2} + \frac{2}{\pi}\sum_{n=1,3,5,\cdots}^{\infty}\left(\frac{1}{n}\sin\frac{2n\pi x}{L}\right)$$

The partial sums of the series corresponding to $n = 1$, $n = 3$, $n = 5$ and $n = 51$ are depicted in Figure B.1. We observed under- and overshoots near discontinuities. These

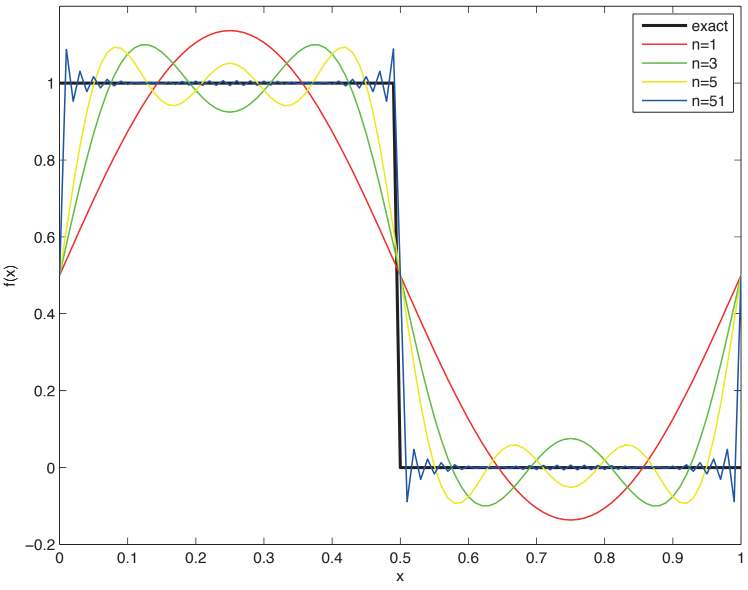


Figure B.1: Partial sums in the Fourier series for the block wave.

wiggles persist even when $n \to \infty$. This is known as the **Gibbs phenomenon**. The reason is that we are trying to approximate a *discontinuous* profile by means of the sum of *continuous* functions.

Many other examples of Fourier series of special functions can be found in many textbooks, e.g. Andrews (1986) and Boyce and DiPrima (1986), and on the internet, e.g. Wikipedia.

B.3 Complex Fourier series

In many applications it is convenient to express a Fourier series in terms of exponential functions instead of sine and cosine functions. Recall the Euler's formula

$$e^{inkx} = \cos nkx + \mathrm{i}\sin nkx$$

with $k = 2\pi/L$ the wave number. It follows that

$$\sin nkx = \frac{e^{\mathrm{i}nkx} - e^{-\mathrm{i}nkx}}{2\mathrm{i}}$$

and

$$\cos nkx = \frac{e^{\mathrm{i}nkx} + e^{-\mathrm{i}nkx}}{2}$$

so that we have

$$a_n \cos nkx + b_n \sin nkx = \frac{1}{2}(a_n - \mathrm{i}b_n)e^{\mathrm{i}nkx} + \frac{1}{2}(a_n + \mathrm{i}b_n)e^{-\mathrm{i}nkx}$$

(Note that $\mathrm{i}^{-1} = -\mathrm{i}$.) Next we define another Fourier coefficient c_n, as follows

$$c_n = \frac{1}{2}(a_n - \mathrm{i}b_n)\,, \quad n = 1, 2, \cdots$$

Its complex conjugate is given by

$$\overline{c}_n = \frac{1}{2}(a_n + \mathrm{i}b_n)$$

If the coefficients a_n and b_n are real, then one can prove that

$$c_n = \overline{c}_{-n}\,, \quad n < 0$$

Hence, the Fourier series (B.2.2) can be written as

$$\begin{aligned} f(x) &= \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(c_n e^{\mathrm{i}nkx} + \overline{c}_n e^{-\mathrm{i}nkx}\right) \\ &= \frac{a_0}{2} + \sum_{n=1}^{\infty} c_n e^{\mathrm{i}nkx} + \sum_{n=-\infty}^{-1} c_n e^{\mathrm{i}nkx} \\ &= \sum_{n=-\infty}^{\infty} c_n e^{\mathrm{i}nkx} \end{aligned}$$

with $c_0 = a_0/2$. The corresponding Fourier coefficient is given by

$$\begin{aligned} c_n = \frac{1}{2}(a_n - \mathrm{i}b_n) &= \frac{1}{L}\int_0^L f(x)\,(\cos nkx - \mathrm{i}\sin nkx)\,dx \\ &= \frac{1}{L}\int_0^L f(x)e^{-\mathrm{i}nkx}dx \end{aligned}$$

B.4 Discrete Fourier transform

From the previous section we have derive the following theorem. Given a (piecewise) continuous function $f(x)$ on an interval $0 \leq x \leq L$, it can be represented as the sum of an infinite set of complex exponentials, as follows

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{\mathrm{i}nkx} \tag{B.4.1}$$

and the Fourier coefficients are given by

$$c_n = \frac{1}{L} \int_0^L f(x) e^{-\mathrm{i}nkx} dx$$

with $k = 2\pi/L$ the wave number.

In science and engineering we are often dealing with *discrete* functions. These functions are defined only at discrete points. One can think of sampling measured quantities (e.g. water level) at discrete time intervals. It is stressed here that we just consider discrete values of the function at discrete points, and there is no data in between these points. Obviously, the Fourier coefficients are given only at discrete points as well. It turns out that these Fourier coefficients are periodic, and can therefore be represented by a finite set of complex exponentials. We have the following definition (Bracewell, 1978).

A sequence of M discrete numbers $X_0, \ldots, X_{M-1}$ is transformed into another sequence of M numbers $x_0, \ldots, x_{M-1}$:

$$x_m = \sum_{j=0}^{M-1} X_j e^{\mathrm{i}\,2\pi \frac{j}{M} m}$$

This is called the **discret Fourier transform** (DFT). However, it can also be interpreted as a discrete Fourier *series* for which the (periodic) discrete function x_m is expanded as that series of harmonics. The coefficient X_j can be considered as the Fourier coefficient, which can be obtained by the inverse of the DFT

$$X_j = \frac{1}{M} \sum_{m=0}^{M-1} x_m e^{-\mathrm{i}\,2\pi \frac{j}{M} m}$$

More details on the DFT can be found in Bracewell (1978). The DFT has found its way in the application of the so-called fast Fourier transform (FFT) which is an efficient algorithm to compute the Fourier transform. See e.g. Press et al. (1996) for details.

In this course we are dealing with many finite difference schemes. These schemes are basically recurrence relations which recursively define a grid function sampled at grid points. Such a grid function can be expanded as a discrete Fourier series. Let us consider a grid function T_m^n defined at an equidistant grid $x_m = m\Delta x\,, m = 0, \ldots, M$ with $\Delta x = L/M$ a mesh width and L the domain length, and at times $t_n = n\Delta t\,, n = 1, 2, 3, \ldots$ with Δt a constant time step. Note that the grid function is periodic with L, i.e. $T_0^n = T_M^n$. The series is then given by

$$T_m^n = \sum_{j=0}^{M-1} \tilde{T}_j^n e^{\mathrm{i}\,2\pi \frac{j}{M} m} \tag{B.4.2}$$

with $\tilde{T}_j^n$ the Fourier coefficient that is time dependent. However, it can be regarded as the amplitude of the jth harmonic at time step t_n. It is easy to see why the expansion, Eq. (B.4.2), can be considered as a Fourier series. The exponent can be rewritten as follows

$$\mathrm{i}\,2\pi \frac{j}{M} m = \mathrm{i}\,2\pi j \frac{\Delta x}{L} m = \mathrm{i}\,jkm\Delta x = \mathrm{i}\,jkx_m$$

so that we have

$$T_m^n = \sum_{j=0}^{M-1} \tilde{T}_j^n e^{\mathrm{i}\,jkx_m}$$

which can be compared with Eq. (B.4.1). Another viewpoint is the following. The grid function T_m^n is decomposed into M harmonics. The jth harmonic has a wave length of L/j. This wave length is denoted as L_j. The corresponding wave number equals $k_j = 2\pi/L_j$. See also Figure 3.2. So, the exponent can also be written as

$$\mathrm{i}\,2\pi \frac{j}{M} m = \mathrm{i}\,2\pi j \frac{\Delta x}{L} m = \mathrm{i}\,2\pi \frac{\Delta x}{L_j} m = \mathrm{i}\,k_j \Delta x\, m = \mathrm{i} m \phi_j$$

with

$$\phi_j \equiv k_j \Delta x$$

the phase angle associated with the jth harmonic. This phase angle covers the frequency domain $[0, 2\pi)$ in steps of $2\pi/M$. For instance, $\phi_j \approx 0$ represents low-frequency waves with length $\sim L$, whereas $\phi_j \approx \pi$ symbolizes high-frequency waves with length of approximately $2\Delta x$. (The shortest resolvable wave of $2\Delta x$ is called the $\pi-$mode.) Hence, the expansion is now

$$T_m^n = \sum_{j=0}^{M-1} \tilde{T}_j^n e^{\mathrm{i} m \phi_j}$$

which is the one that we prefer to use in this reader for different applications, e.g. the Von Neumann stability analysis and the computation of the phase error. The interpretation is clear: a periodic grid function T_m^n is represented as the finite sum of Fourier modes ("sine waves") with different amplitudes $\tilde{T}_j^n$ and phases ϕ_j.

Bibliography

[1] Abbott, M.B., 1979. *Computational hydraulics: elements of the theory of free surface flows*, Pitman.

[2] Abbott, M.B. and D.R. Basco, 1989. *Computational fluid dynamics: an introduction for engineers*, Longman Scientific and Technical.

[3] Andrews, L.C., 1986. *Elementary partial differential equations with boundary value problems*, Academic Press College Division.

[4] Batchelor, G.K., 1967. *An introduction to fluid dynamics*, Cambridge University Press, Cambridge.

[5] Boyce, W.E. and R.C. DiPrima, 1986. *Elementary differential equations and boundary values problems*, Fourth edition, John Wiley and Sons.

[6] Bracewell, R.N., 1978. *The Fourier transform and its applications*, Second edition, McGraw-Hill.

[7] Ferziger, J.H. and M. Perić, 1996. *Computational methods for fluid dynamics*, Springer, Berlin.

[8] Fletcher, C.A.J., 1988. *Computational techniques for fluid dynamics*, Volumes 1 and 2. Springer, Berlin.

[9] Gear, C.W., 1971. *Numerical initial value problems in ordinary differential equations*, Prentice-Hall.

[10] Golub, G.H. and C.F. Van Loan, 1989. *Matrix computations*, Second edition, The John Hopkins University Press.

[11] Henrici, P., 1962. *Discrete variable methods in ordinary differential equations*, John Wiley and Sons.

[12] Hirsch, C., 1990. *Numerical computation of internal and external flows*, Volumes 1 and 2. John Wiley and Sons, Chichester.

[13] Kowalik, Z. and T.S. Murty, 1993. *Numerical modeling of ocean dynamics*, World Scientific, Singapore.

[14] Lambert, J.D., 1991. *Numerical methods for ordinary differential systems*, John Wiley and Sons, New York.

[15] Mitchell, A.R. and D.F. Griffiths, 1994. *The finite difference method in partial difference equations*, John Wiley and Sons.

[16] Patankar, S.V., 1980. *Numerical heat transfer and fluid flow*, McGraw-Hill.

[17] Peyret, R. and T.D. Taylor, 1985. *Computational methods for fluid flow*, Springer-Verlag, Berlin.

[18] Press, W.H., S.A. Teukolsky, W.T. Vetterling and B.P. Flannery, 1996. *Numerical recipes in Fortran 90*, 2nd edition, Cambridge University Press.

[19] Richtmyer, R.D. and K.W. Morton, 1967. *Difference methods for initial value problems*, John Wiley and Sons, New York.

[20] Roache, P.J., 1972. *Computational fluid dynamics*, Hermosa Publishers, Albuquerque, New Mexico.

[21] Strang, G., 1980. *Linear algebra and its applications*, Second edition, Harcourt Brace Jovanovich, Publishers.

[22] Vreugdenhil, C.B., 1994. *Numerical methods for shallow-water flow*, Kluwer, Dordrecht.

[23] Wesseling, P., 2001. *Principles of computational fluid dynamics*, Springer, Berlin.

Index