Numerical Ordinary Differential Equations and Applications

Des Higham, School of Mathematics, University of Edinburgh
(based on notes previously prepared by several colleagues)

January, 2024

Note:
Chapter 1 presents background material, much of which should be familiar.

Contents

Chapter 1

Preliminaries

Main concepts: In this chapter we introduce the necessary background in ordinary differential equations with a few examples.

Keywords: mathematical model, population growth, Lotka-Volterra (predator-prey), ordinary differential equation, initial value problem, existence-uniqueness theorem, Picard's theorem, Lipschitz condition, direction field.

1.1 Ordinary Differential Equations

Ordinary differential equations (ODEs) commonly arise as models of dynamical systems, that is systems whose state depends on time.

A typical example of an ODE arising in this way is the population growth equation

$$\frac{\mathrm{d}N}{\mathrm{d}t} = rN \tag{1.1}$$

where N represents the population at time t and r is a constant, sometimes called the growth coefficient. We will refer to this model and variations of it below. To make the discussion concrete, think of N as the population of aphids (insect pests) on a rose farm.

The equation (1.1) relates two things: on the left, the derivative or *rate of change* of the population with respect to time; on the right, a constant multiple of the population itself. In general an ODE may involve a linear relationship, as here, or the relationship may be nonlinear. Linear equations are easier to handle because they can be solved exactly and the solution written down as a formula that tells us the size of the population at any time. While we will study linear equations often in this course, we will also study the more complicated case where the relationship between the quantity of interest and its derivative is nonlinear. Numerical methods are well suited to handling both cases.

More generally, a **differential equation** of the general form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y) \tag{1.2}$$

specifies the relation between an unknown differentiable function $y(t)$ and its derivative $y'(t) = \frac{\mathrm{d}y}{\mathrm{d}t}$. The word "ordinary" is used because the equation involves derivatives with respect to a single independent variable (in this case t). This is in contrast to a **partial differential equation**, which is one involving derivatives with respect to several independent variables, e.g.

$$\frac{\partial u}{\partial t} = \frac{\partial u}{\partial x} \qquad \text{or} \qquad \frac{\partial^2 u}{\partial x^2} + a(x,y)\frac{\partial^2 u}{\partial y^2} = b(x,y).$$

For ODEs, time is often (but not always) the independent variable, which is suggested by the letter t. We will use the notation $y'(t)$ and occasionally $\dot{y}$ to denote $\frac{\mathrm{d}y}{\mathrm{d}t}$.

A **solution** of the differential equation (1.2) is a differentiable function $y(t)$ that satisfies the differential equation,

$$\frac{\mathrm{d}y(t)}{\mathrm{d}t} = f(t, y(t)),$$

for all t in some interval of interest. For example, for the population model (1.1), the function

$$\tilde{N}(t) = Ce^{rt}$$

is a solution for each constant $C \in \mathbb{R}$, since $\mathrm{d}\tilde{N}/\mathrm{d}t = Cr\exp(rt) = r\tilde{N}(t)$ conforms to (1.1).

To determine a unique solution, it is necessary to specify the value of the function at some point t. An **initial value problem** combines a differential equation, an initial value, and an interval on which the solution is required

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \quad y(t_0) = y_0, \quad t \in [t_0, t_0 + T]. \tag{1.3}$$

In (1.1), assuming an initial population N_0 at $t = 0$ the population at any later time t is $N(t) = N_0\exp(rt)$. If $r < 0$ (i.e., the death rate exceeds the birth rate), the model predicts exponential decay of the population. If $r > 0$ the births outnumber the deaths and the population grows exponentially.

The model (1.1) might be useful to describe the initial phase of an aphid infestation, but one would not expect the exponential growth to go unchecked. Limited food resources will cause an increase in death rate, or a decrease in birth rate, or both when the aphid population gets too large. A rose farm can only sustain a certain number of aphids, say N^*. A more realistic model is then

$$\frac{\mathrm{d}N}{\mathrm{d}t} = rN\left(1 - \frac{N}{N^*}\right).$$

This equation can be normalized by dividing both sides by N^* and letting $n(t) = N(t)/N^*$. In terms of n, this becomes the *logistic model*

$$\frac{\mathrm{d}n}{\mathrm{d}t} = rn(1-n). \tag{1.4}$$

An equation such as (1.4) can be analytically solved using separation of variables. We proceed as follows, using partial fractions:

$$\begin{aligned} r\,\mathrm{d}t &= \frac{\mathrm{d}n}{n(1-n)} \\ \Rightarrow rt + C &= \int \frac{\mathrm{d}n}{n(1-n)} = \int \frac{1}{n} + \frac{1}{1-n}\,\mathrm{d}n \\ &= \ln|n| - \ln|1-n|. \end{aligned}$$

Assuming $n \in (0,1)$, taking the exponential of both sides, and denoting $c = \exp C$,

$$\frac{n}{1-n} = ce^{rt} \quad \Rightarrow \quad n(t) = \frac{ce^{rt}}{1+ce^{rt}}.$$

If $n(0) = n_0$, then $c = n_0/(1-n_0)$ and the solution can be written

$$n(t) = \frac{n_0 e^{rt}}{(1-n_0)+n_0 e^{rt}}.$$

If $r > 0$ it can be checked that

$$\lim_{t\to\infty} n(t) = 1$$

for any initial state $n_0 > 0$. That is, the population eventually saturates ($N \to N^*$) for any initial population. This behavior is shown in the left plot of Figure 1.1.

Now suppose a predator, such as a ladybird, is introduced into the rose farm. The logistic model can be extended to incorporate *predation* by ladybirds as follows:

$$\frac{\mathrm{d}n}{\mathrm{d}t} = rn\left(1 - \frac{n}{s}\right) - \frac{n^2}{1+n^2}. \tag{1.5}$$

The last term gradually "turns on" ladybird predation as the population n grows greater than 1. We have re-introduced the saturation level s because there are now two important populations levels—the natural saturation level, and the level at which ladybirds get "interested". Some solutions of this model are shown in the right plot of Figure 1.1. It is interesting to note that for $s = 20$ and either $r = 0.1$ or $r = 0.6$, the aphid population eventually tends to a constant value, independent of the initial population. This is similar to the case of the logistic equation without predation (1.4). For the case $s = 20$, $r = 0.25$, however, the final aphid population depends on the starting population. For a starting population of about $n_0 < 5$, the aphids settle down to a low level of about $n = 0.5$, whereas for a starting population of $n_0 > 5$, the aphid population eventually stabilizes at a relatively high level of around $n = 14.5$. Such a change in the quality of solutions with a change in parameter values is called a *bifurcation*.

This model is still rather crude, because it assumes an endless supply of ladybirds. A better model might involve a system of two equations representing the populations of both aphids and ladybirds. Let $p(t)$ denote the (normalized) population of ladybirds. In

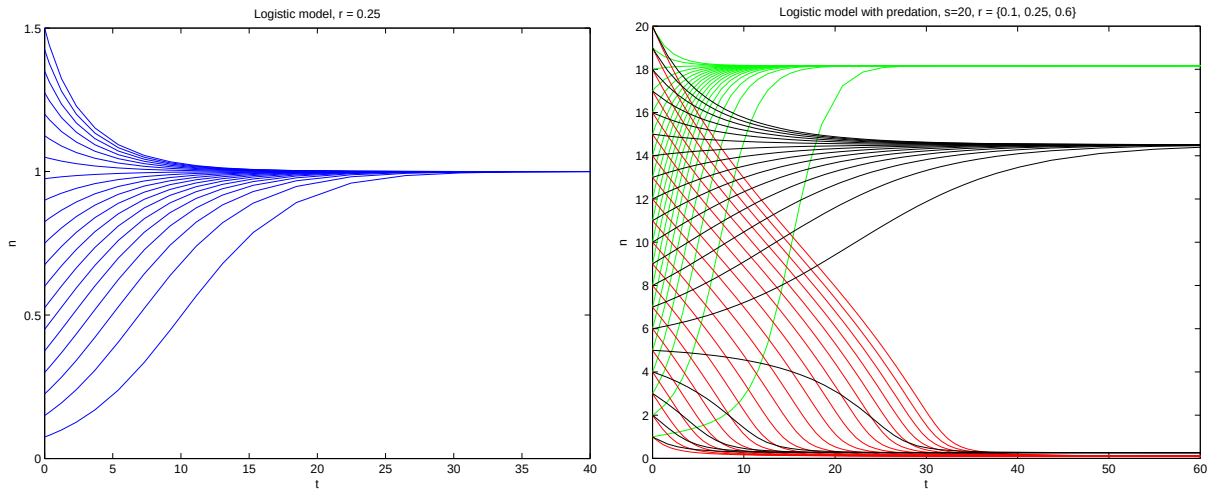

Figure 1.1: Solutions of the logistic model. On the left, solutions of (1.4) with $r = 0.25$ for different initial conditions. On the right, with predation (1.5), $s = 20$ and $r = 0.6$ (green), $r = 0.25$ (black), and $r = 0.1$ (red). For $r = 0.25$, there are two steady states, depending on the initial condition.

the absence of aphids, the ladybirds would die out exponentially: $\mathrm{d}p/\mathrm{d}t = -mp$ for some $m > 0$. But with an increasing number of aphids, the ladybirds will flourish, so the ladybird population might satisfy

$$\frac{\mathrm{d}p}{\mathrm{d}t} = anp - mp$$

for positive constants a and m. Similarly, in the absence of ladybirds, the aphid population would grow exponentially (at least until they saturate the environment). And their death rate should increase in proportion to the number of predators, so

$$\frac{\mathrm{d}n}{\mathrm{d}t} = bn - cnp,$$

for positive constants b and c. It is possible to rescale these equations such that only one constant remains, and the result is the *Lotka-Volterra predator-prey model*

$$\frac{\mathrm{d}n}{\mathrm{d}t} = rn(1-p) \tag{1.6a}$$

$$\frac{\mathrm{d}p}{\mathrm{d}t} = p(n-1), \tag{1.6b}$$

where $r > 0$.

The Lotka-Volterra model is an example of a *system* of two coupled ordinary differential

equations. More generally one may have a **system of d differential equations**:

$$\frac{\mathrm{d}y_1}{\mathrm{d}t} = f_1\left(t, y_1, y_2, \ldots, y_d\right) \tag{1.7}$$

$$\frac{\mathrm{d}y_2}{\mathrm{d}t} = f_2\left(t, y_1, y_2, \ldots, y_d\right) \tag{1.8}$$

$$\vdots \tag{1.9}$$

$$\frac{\mathrm{d}y_d}{\mathrm{d}t} = f_d\left(t, y_1, y_2, \ldots, y_d\right). \tag{1.10}$$

When dealing with systems of differential equations, it is convenient to use vector notation. Let $y(t) : \mathbb{R} \to \mathbb{R}^d$ denote the vector with components $y_1(t), \ldots, y_d(t)$ and $f(t, y) : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ denote the vector with components $f_1(t, y), \ldots f_d(t, y)$, i.e.

$$y(t) = \left(y_1(t), \cdots, y_d(t)\right)^T, \qquad f(t, y(t)) = \left(f_1(t, y), \cdots, f_d(t, y)\right)^T.$$

Then the system of differential equations above can be written using the same notation as (1.2), and the same goes for the initial value problem (1.3), where it is understood that the initial condition y_0 is now a vector in $\mathbb{R}^d$. The special case $d = 1$ is referred to as a *scalar* differential equation. Scalar differential equations are special because they can often be solved in quadratures, which we will discus later. In this text we will often consider systems with small dimension d, say $d < 10$. However, many real applications have very large dimensions: millions of variables are not uncommon. It is important to keep this in mind when one is designing numerical methods.

Going back to our rose garden example, the reproduction rate of aphids is actually not constant, as suggested by the constant parameter r in the above models. For example, aphid reproduction depends on temperature. One way to incorporate the effects of a typical yearly temperature variation is to allow r to depend explicitly on time. For example, the logistic model with predation (1.5) and time dependent reproduction becomes

$$\frac{\mathrm{d}n}{\mathrm{d}t} = r(t)\, n \left(1 - \frac{n}{s}\right) - \frac{n^2}{1 + n^2}, \tag{1.11}$$

where $r(t)$ is a specified function, say $r(t) = r_0(\cos \varepsilon t + c_0)$.

A system of differential equations for which $f(t, y)$ can be written as a function of y only is an **autonomous differential equation**

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(y), \qquad y, f \in \mathbb{R}^d. \tag{1.12}$$

The ODE (1.11) is not autonomous (we say it is nonautonomous) because $\cos \varepsilon t + c_0$ appears in the right hand side.

It is useful to observe that autonomous differential equations are invariant under a translation of time. That means that if we define $\tau = t - t_0$, then $\mathrm{d}t/\mathrm{d}\tau = 1$, and

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = \frac{\mathrm{d}y}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}\tau} = \frac{\mathrm{d}y}{\mathrm{d}t},$$

so the initial value problem $\mathrm{d}y/\mathrm{d}t = f(y)$, $y(t_0) = y_0$ can be replaced by $\mathrm{d}y/\mathrm{d}\tau = f(y)$, $y(0) = y_0$. Thus without loss of generality we can take the initial value of time to be $t = 0$, as long as the system is autonomous.

It is always possible, and sometimes useful, to convert a *nonautonomous* ODE (i.e. one with explicit dependence on t) into an autonomous one. This is done by increasing the dimension d by one. We define $y_{d+1} \equiv t$ and introduce a new independent variable s such that $dt/ds = 1$ and $t(0) = t_0$. The system (1.7)–(1.10) becomes

$$\begin{aligned} \frac{\mathrm{d}y_1}{\mathrm{d}s} &= f_1\left(y_{d+1}, y_1, y_2, \ldots, y_d\right) \\ &\vdots \\ \frac{\mathrm{d}y_d}{\mathrm{d}s} &= f_d\left(y_{d+1}, y_1, y_2, \ldots, y_d\right) \\ \frac{\mathrm{d}y_{d+1}}{\mathrm{d}s} &= 1 \end{aligned}$$

where the independent variable s no longer appears explicitly on the right side.

1.2 Differential Equations

Summing up the main idea from the previous section:

Definition 1.2.1 A *differential equation* relates the value of a given function and its derivatives. In particular,

$$\frac{d}{dt}y = f(t, y) \tag{1.13}$$

is termed an *ordinary differential equation* (ODE) because it involves derivatives with respect to a single *independent variable* (t). Here y is the *dependent variable*. □

We should always think of a differential equation as an equation involving a function. So (1.13) really means

$$\frac{d}{dt}y(t) = f(t, y(t))$$

where the equality would be expected to hold in some specified time interval of interest.

Example.
The differential equation

$$\frac{\mathrm{d}y}{\mathrm{d}t} = g(t)$$

is often easy to solve. Because the right hand side does not involve y at all, it means that we are giving a precise expression for the derivative of y and it follows that, if G is an antiderivative of g (i.e. $G'(t) = g(t)$), then

$$y(t) = G(t) + C.$$

The constant C is arbitrary, and without any additional information, it should always be retained in describing the solution.

More complicated differential equations occur frequently in practice. Sometimes we can solve them easily, but more often we cannot. Most commonly, differential equations arise in mathematical modelling through the application of physical laws.

1.2.1 Linearity

An important feature of some differential equations is *linearity*. We say that a scalar differential equation $\mathrm{d}y/\mathrm{d}t = f(t, y)$ is linear (really, linear in $\underline{y}$) if $f(t, y) = a(t)y + b(t)$, for some functions a and b. It is crucial to note that for a linear differential equation,

- a and b need not be linear functions of t;
- the solution is not typically a linear function of t.

We refer to a and b as coefficient functions. If a differential equation is linear, and if, furthermore, a and b are constant, we say that the differential equation is a constant coefficient, linear differential equation.

Example.
The following are all linear differential equations

a.
$$\frac{\mathrm{d}y}{\mathrm{d}x} = kx^2y - cx^3,$$

b.
$$x' = e^{-t\beta}x,$$

c.
$$\dot{u} = \cos(\omega t)u - \sin(\omega t).$$

In the first, x is the independent variable, y the dependent variable, and k and c are parameters. Here, kx^2 and $-cx^3$ are coeficient functions. In the second, t is the independent variable, x is the dependent variable, β is a free parameter. In the third, t is the independent variable, u the dependent variable, and ω is a parameter.
The following is a linear **system** of ordinary differential equations:

$$\begin{aligned} \dot{x} &= u \\ \dot{u} &= -\omega^2 x. \end{aligned}$$

It describes the oscillations of a body moving in one dimension (position x, velocity u) attached to a fixed point at the origin by a spring. The coefficient ω is the *frequency* of the oscillator.

Example.
The following are examples of *nonlinear* differential equations

a.
$$\frac{\mathrm{d}u}{\mathrm{d}t} = u^2,$$

b.
$$\mathrm{d}x/\mathrm{d}t = kx^2t - cx^3.$$

Duffing's oscillator is a non-autonomous, nonlinear system of ordinary differential equations

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= u, \\ \frac{\mathrm{d}u}{\mathrm{d}t} &= -\beta x - \alpha x^3 - \delta u - \gamma\cos(\omega t). \end{aligned}$$

Example.
Glycolosis is a reaction in living cells in which two chemicals are formed and react according to

$$A \xrightarrow{k_1} X, \quad 2X + Y \xrightarrow{k_2} 3X, \quad B + X \xrightarrow{k_3} Y, \quad X \xrightarrow{k_4} C.$$

This represents the interaction of 5 separate species. Species A is converted to X at a constant rate k_1. 2 units of X combine with one of Y to produce 3 units of X (there may be some by-products of this reaction, but they are ignored as they have no further impact on the process of interest). X also combines with another constituent B to produce Y. Moreover, X breaks down to C at a constant rate. To write mathematical equations for this process, we use as variables the concentrations $x = \{X\}$ and $y = \{Y\}$ of X and Y, also $a = \{A\}$ and $b = \{B\}$, then we have

$$\frac{\mathrm{d}x}{\mathrm{d}t} = a + k_2x^2y - bk_3x - k_4x \tag{1.14}$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = bk_3x - k_2x^2y. \tag{1.15}$$

Many interesting differential equations (in fact, almost all the interesting ones!) are *nonlinear* in which case finding solutions can be challenging.

1.2.2 Initial Value Problems

Throughout this course, we focus on the initial value problem,

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), \quad y(t_0) = y_0. \tag{1.16}$$

A solution of (1.16) is a function y that satisfies

$$\frac{\mathrm{d}}{\mathrm{d}t}y(t) = f(t, y(t)), \quad y(t_0) = y_0, \tag{1.17}$$

on some given interval $t_0 \leq t \leq T$. In some applications we are also interested in the behaviour of $y(t)$ as $t \to \infty$.

We will use the notation (1.16) to describe both the scalar equation (in which case $y : \mathbb{R} \to \mathbb{R}$, and $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$) and the d-dimensional system ($y : \mathbb{R} \to \mathbb{R}^d$, $f : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$). There are essentially three approaches to analysing (1.16):

1. *analytical solution* of (1.16), i.e. finding a formula for y to satisfy (1.16),
2. *qualitative study* of the properties of solutions based on analytical and/or graphical techniques,
3. computation of an *approximate solution* of (1.16) by a numerical method.

An important concept that one must grasp is that (1.16) may have a well-defined, unique solution but it may not be possible to give an explicit formula for it in terms of basic arithmetic operations ($+$, $-$, $*$, etc.) and compositions of standard functions like $\sqrt{\cdot}$, cos, ln, etc. One may regard the set of standard functions as a "dictionary" in terms of which various other functions may be described, but this dictionary is too abridged to express the solution of a typical differential equation. One approach is to look to "enrich the language" by adding functions to the dictionary, and mathematicians have attempted to construct a broader vocabulary by analysing a wide variety of *special functions*, usually constructed to solve particular differential equations. In general, however, this approach has limited scope.

The qualitative approach is different, and, most would argue, more powerful, since it can be applied to a wide variety of situations. Here, we attempt to understand the behavior of families of solutions without actually obtaining an explicit representation.

This course focusses on item 3.—the design, analysis and implementation of computational methods that produce approximate solutions.

1.3 The Existence-Uniqueness Theorem

Let us assume that y has a continuous derivative, i.e. $y \in C^1([a, b])$ for some interval $[a, b]$. From equation (1.17), it appears that f should also be a continuous function. It turns out to be useful to assume that f has a slightly stronger property than continuity.

1.3.1 Picard's Theorem

We now state a fundamental theorem for ordinary differential equations, Picard's Theorem on existence and uniqueness of solutions. We begin with the scalar case. We will not give a proof, since the focus of this course is numerical methods.

Theorem 1.3.1 (Picard's Theorem) *Let the function $f(\cdot, \cdot)$ be a continuous function of its arguments in a region of the plane containing the rectangle*

$$D = \{(t, y) \,|\, t_0 \leq t \leq T, |y - y_0| \leq K\},$$

where T, K are positive constants. Suppose, moreover, that there exists a constant $L > 0$ (a Lipschitz constant*) such that*

$$|f(t,u) - f(t,v)| \leq L|u - v|, \quad whenever (t,u),(t,v) \in D. \tag{1.18}$$

Because the set D is closed and bounded, we know that there is a nonnegative constant M_f such that

$$M_f = \max\{|f(t,u)| \,|\, (t,u) \in D\}.$$

Assume that

$$M_f(T - t_0) \leq K.$$

Then there exists a unique continuously differentiable function $t \mapsto y(t)$, defined on $[t_0, T]$, such that

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t,y), \quad y(t_0) = y_0.$$

It is usually more useful in practice to replace the assumption of Lipschitz continuity by the stronger assumption that f is continuous in t and continuously differentiable in y, meaning that $\partial f/\partial y$ is a continuous function (of t and y). Note that $\partial f/\partial y$ being continuous implies the existence of a Lipschitz constant. If f has a continuous partial derivative with respect to y, then on any finite domain $D \subset \mathbb{R}^2$, we may take

$$L = \sup_D \left|\frac{\partial f}{\partial y}\right|,$$

and the function can be seen satisfy the Lipschitz condition (1.18) on D. This follows from the Mean Value Theorem.

We can extend the theorem to systems of differential equations (1.7)–(1.10). Here is one version.

Theorem 1.3.2 (Local Existence/Uniqueness of Solutions for ODE systems) *Suppose $f : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ is continuous and has continuous partial derivatives with respect to all components of the dependent variable y in a neighborhood of the point (t_0, y_0). Then there is an interval $I = (t_0 - \delta, t_0 + \delta)$ containing t_0 and a unique function y which is continuously differentiable on I such that y satisfies (1.16).*

Note that this is a somewhat limited result. It only tells us that there is a unique solution in the nearby vicinity of the initial point (t_0, y_0). In practice, a solution on a larger interval can often be found.

Example.
Let $f(t,y) = 1 + y^2$, then it is easy to see that f and $\partial f/\partial y$ are continuous everywhere. However, a solution of the differential equation through $t_0 \in (-\pi/2, \pi/2)$, y_0 is easily confirmed to be

$$y(t) = \tan(t - t_0 + \arctan(y_0)).$$

This function has a vertical asymptote (and is not defined) at $t^* = t_0 + \frac{\pi}{2} - \arctan(y_0)$.

Example.
Let $f(x, y) = (3/2)(x-y)^{1/3}+1$. f is defined everywhere, but the derivative of f does not exist at $y = x$. By setting $w = y - x$, one may rewrite the differential equation as

$$\frac{dw}{dx} = -\frac{3}{2}w^{1/3},$$

Solving this separable equation, reverting to the original variables, and using the initial condition $y(x_0) = y_0$ results in the solution formula

$$y = x \pm ((y_0 - x_0)^{2/3} - x + x_0)^{3/2},$$

where the sign is chosen based on whether y_0 is greater than or less than x_0. Clearly this solution is uniquely defined for $x \leq x_0 + (y_0 - x_0)^{2/3}$, i.e., *until the solution reaches the line* $y = x$. Several solutions are graphed in Figure 1.2.

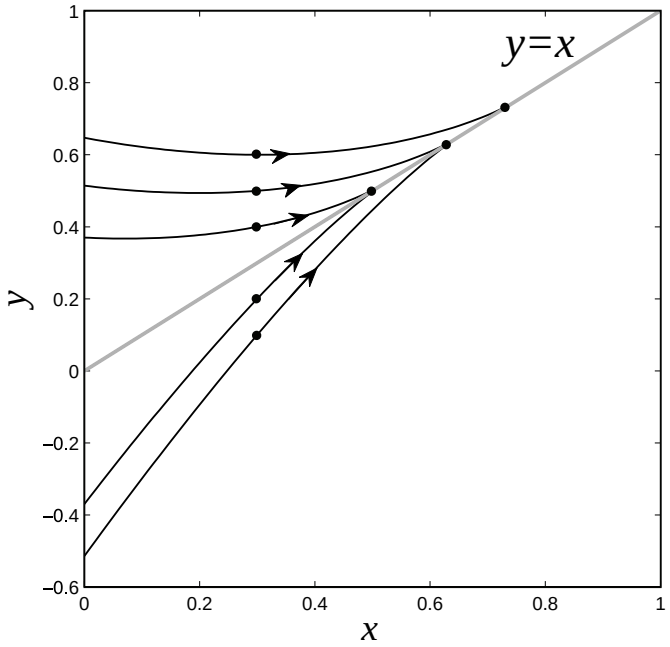

Figure 1.2: Solutions of $\mathrm{d}y/\mathrm{d}x = (3/2)(x-y)^{1/3}+1$ are plotted starting from several initial points on the line $x = 0.3$. Each solution is defined up until it meets the line $y = x$. Dots show initial points and final points of solutions.

1.4 Slope Fields

A useful technique for analysing scalar differential equations (including nonseparable ones) is to construct the *slope field*. Imagine that we have a solution of the initial value problem

(1.16). Then the graph of the solution includes the point $Q = (t_0, y_0)$, where $y_0 = y(t_0)$. What can we say about the slope of the solution at the point Q? Well, we know that the slope of (the tangent line to) a smooth curve is just the derivative y'. Obviously we have a formula for this value at the point (t_0, y_0):

$$y'(t_0) = f(t_0, y_0).$$

It should be noted that calculating the slope of the solution passing through a given point does not require knowledge of anything but the value of the function f at that point. Figure 1.3 illustrates the situation.

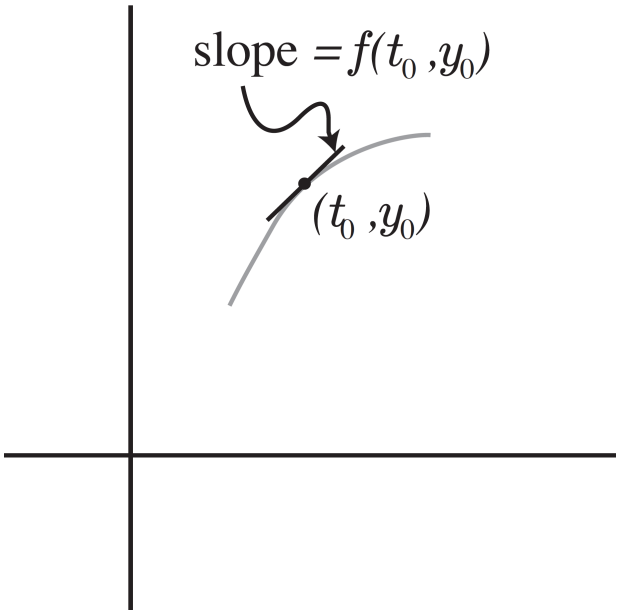

Figure 1.3: The slope of the tangent to the solution curve at a point is obtained by evaluating f at that point.

The value f at a point can be illustrated graphically by the line segment having that slope, to which we could add an arrow to indicate the direction of forward progress.

We can sketch this directed segment at any arbitrary point in the plane, and so we can draw such arrows on a grid of points. The collection of these arrows is what is known as the *slope field*, also called the *direction field*. By examining the slope field it is often possible to gain a great deal of insight into the general behavior of solutions, without actually finding those solutions.

Plotting slope fields is normally quite tedious. We often would need to evaluate the function f at a large collection of points in the xy plane, and then draw the short directed segments at each of these points. This sort of work is very natural on the computer. Below we show the slope field for the differential equation

$$\frac{dy}{dt} = t\cos(y) - y\sin(t). \tag{1.19}$$

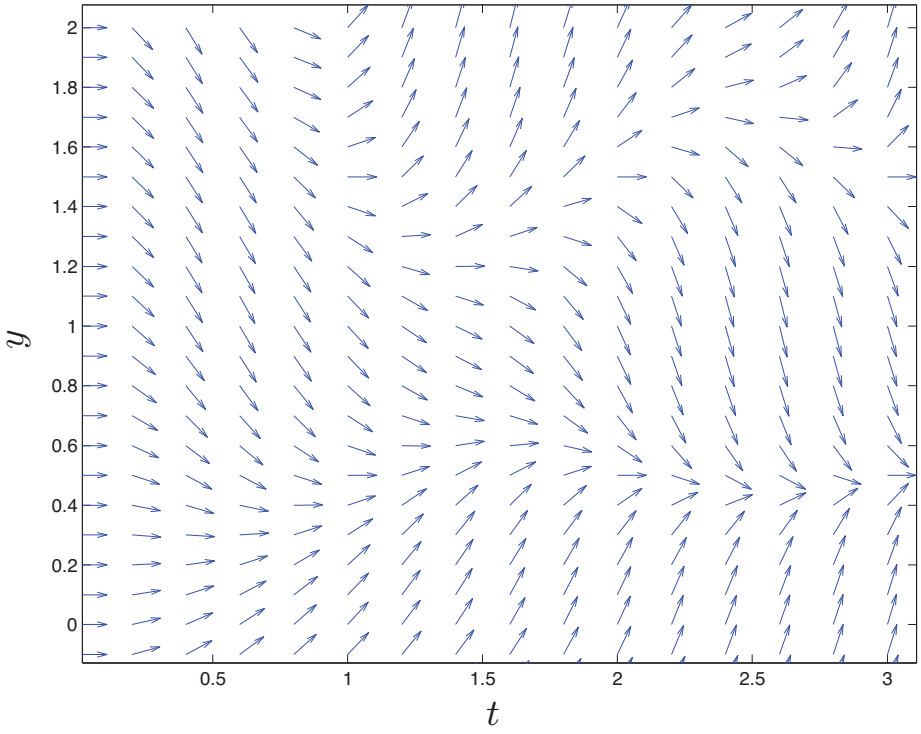

Figure 1.4: A section of the "twisty" slope field for the differential equation $\frac{dy}{dt} = t\cos(y) - y\sin(t)$.

We could imagine using the slope field to approximately follow a solution from a given point, by just keeping our curve tangent to the lines of the slope field at any point. This is one way to motivate Euler's method, which is studied in the next chapter. We can also see how solutions behave in a particular part of the plane, e.g. whether solutions tend to come together or spread apart. The slope field is an example of a graphical technique for qualitative analysis of differential equations.

Chapter 2

Euler's Method and Taylor Series Methods

Main concepts: In this chapter we introduce numerical integrators and time-stepping, then we prove the convergence of Euler's method, discuss the flow map perspective on numerical methods, and introduce Taylor Series methods which in some cases provide greater accuracy.

Keywords: Euler's method, trapezoidal rule, algorithm, order notation, convergence, Taylor series methods.

2.1 The Graphical Version of Euler's Method

Try this simple experiment. Turn to Figure 1.4, the graph of the slope field for

$$\frac{\mathrm{d}y}{\mathrm{d}t} = t\cos y - y\sin t.$$

Place your pencil down on the graph at any particular point and look at the arrows near that point. Estimate which way the arrow would point at the place where your pencil point rests. Now draw a small line segment, say about a half a centimeter, stop but do not raise the pencil from the paper. Look at the arrows near this new point and estimate the direction of the slope field arrow at this place. Now make another move of about a half a centimeter in this new direction, stop and repeat the process until the pencil leaves the graph. The curve that results is an approximation to a solution of the differential equation and the method you used is essentially Euler's method.

2.2 Definition of Euler's Method on a Scalar ODE

Of course, Euler's method is much better presented in a formula if we are to implement it on a computer. To describe it, we need some notation first. We consider a scalar differential equation

$$\frac{dy}{dt} = f(t, y)$$

to be approximated for t in the interval $[a, b]$. We assume an initial condition $y(a) = y_0$ is given. We also assume a unique exact solution $y(t)$ is known to exist on this interval even though we can't necessarily write a formula for it. Break up the interval $[a, b]$ into N equal-length sub-intervals of length $h = (b - a)/N$, defining a set of $N + 1$ points, t_0, $t_1, \ldots, t_N$, where $t_0 = a$ and $t_N = b$ and more generally,

$$t_m = a + hm.$$

We will approximate the values of the solution $y(t)$ at the points t_0, $t_1, \ldots, t_N$. The idea is to use the Taylor Series approximation of the solution.

Recall that a twice continuously differentiable function $y(t)$ can be expanded near $t = t_0$ as

$$y(t) = y(t_0) + y'(t_0)(t - t_0) + \frac{1}{2}y''(\tau)(t - t_0)^2,$$

where τ is a point between t and t_0. (This is a Taylor series with remainder.)

Let us evaluate the Taylor series about t_0 at t_1:

$$y(t_1) = y(t_0) + y'(t_0)(t_1 - t_0) + \frac{1}{2}y''(\tau)(t_1 - t_0)^2.$$

Since $t_1 = t_0 + h$, we could instead write

$$y(t_1) = y(t_0) + hy'(t_0) + \frac{h^2}{2}y''(\tau).$$

Because of the initial condition, we may write y_0 in place of $y(t_0)$. The differential equation itself tells us that $y'(t_0) = f(t_0, y_0)$, so we may write

$$y(t_1) = y_0 + hf(t_0, y_0) + \frac{h^2}{2}y''(\tau).$$

We do not know anything about τ, so we cannot compute the last term exactly, but note that if h is small the h^2 term should be much smaller in magnitude than the other terms. We can approximate the solution at t_1 by just dropping the last term.

$$y(t_1) \approx y_0 + hf(t_0, y_0).$$

To define Euler's method, we use this approach to construct a value y_1 which approximates $y(t_1)$ by writing

$$y_1 = y_0 + hf(t_0, y_0).$$

Note that this just means that we approximate y at t_1 by taking a small step in the direction of the slope field line at (t_0, y_0) (see Figure 1.3).

Once we have computed y_1 in this way, how do we proceed? The answer is to just repeat the process, starting now from the point (t_1, y_1) and computing y_2 from the formula

$$y_2 = y_1 + hf(t_1, y_1).$$

Note that we are following a slope field line starting at (t_1, y_1) and *not* $(t_1, y(t_1))$, since $y_1 \neq y(t_1)$. The situation is graphed in Figure 2.1. We can obviously continue this process,

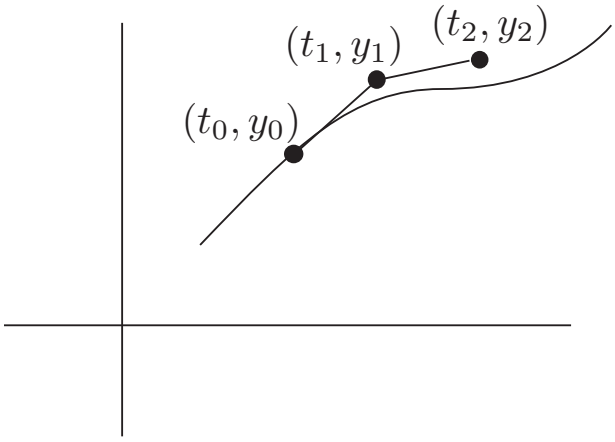

Figure 2.1: 2 Steps of Euler's method starting from (t_0, y_0).

building step-by-step a collection of numbers y_0, y_1, $y_2, \ldots, y_N$, using the recursion

$$y_{n+1} = y_n + hf(t_n, y_n).$$

We anticipate

$$y_i \approx y(t_i),$$

although we have not yet established the quality of this approximation.

2.3 Algorithm for Euler's Method

An *algorithm* is a mathematically precise (typically quite terse) step-by-step procedure. By contrast, an *implementation* of an algorithm refers to the precise sequence of instructions that would used to program a computer in some given language to perform the steps of the algorithm.

An algorithm for Euler's method is easily constructed:

Algorithm: Euler's Method on a scalar ODE

inputs: stepsize h, function $f : \mathbb{R}^2 \to \mathbb{R}$, coordinates of initial point t_0, y_0, number of steps N ($t_0 + Nh$ is the right endpoint of the solution interval)
outputs: a set of values $y_1, \ldots y_N$ representing an approximate solution
for $n = 0, 1, 2 \ldots, N-1$
$y_{n+1} = y_n + hf(t_0 + nh, y_n)$;
end

The notation "**for** $n = 0, 1, \ldots, N-1$ statement; statement; ... **end**" means execute the indicated statements repeatedly, while successively incrementing the index n from 0 to $N-1$.

2.3.1 Twisty Slope Field (again)

Next we consider the example of the twisty slope field given at the end of the previous section of these notes.

$$\frac{\mathrm{d}y}{\mathrm{d}t} = t\cos(y) - y\sin(t). \tag{2.1}$$

Let us compute the solution started from $t_0 = 0, y_0 = 2$ on the interval $[0, 20]$. In Figure 2.2 we have repeated this process for three different choices of the stepsize h ($h = 1, 0.5, 0.25$) using numbers of steps $N = 20, 40, 80$, so that the solution is always being computed on $[0, 20]$.

Is Euler's method a reliable scheme for solving differential equations? In this example the three solutions are wildly different. Which is correct? The ODE is too complicated for us to write the exact solution (it is not separable), so we can only guess at this point. However, from the arguments above, we expect the method to become more accurate as h becomes smaller; we will investigate this further after looking at another example.

2.4 Application of Euler's method to a system of differential equations

Euler's method can be applied to systems of differential equations as easily as to scalar differential equations. The method can be motivated via a vector-valued version of the Taylor series expansion. Let $\mathrm{d}y/\mathrm{d}t = f(t, y)$ represent a system where $y \in \mathbb{R}^d$, i.e. at each point t the solution $y(t)$ will be a vector with d components. Assume an initial condition $y(t_0) = y_0 \in \mathbb{R}^d$. Through the point (t_0, y_0) we assume the existence of a unique solution for which we wish to construct a discrete approximation. Now define, for $n = 1, 2, \ldots$,

$$y_{n+1} = y_n + hf(t_n, y_n).$$

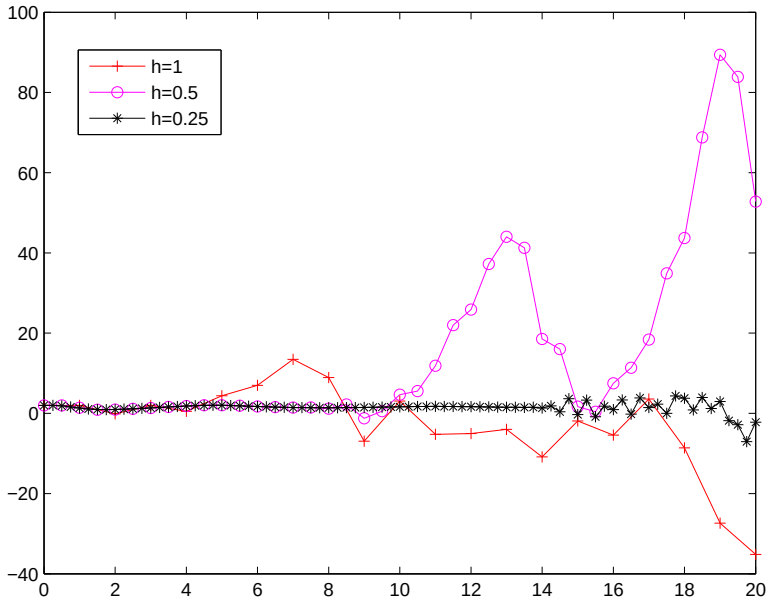

Figure 2.2: Euler method solutions for $h = 1, 0.5, 0.25$

This gives a sequence of points in $\mathbb{R}^d$. The points will, for sufficiently small h, approximate the solution at a corresponding value of t:

$$y_n \approx y(t_0 + nh).$$

2.5 Solving a system of ODEs using Euler's method

Let us apply Euler's method to solve the Lotka-Volterra model (1.6) with $r = 2$. Take the initial value to be $(n, p) = (0.5, 0.5)$.

The relationship between predators and prey is shown in Figure 2.3. This figure is troublesome because it appears to show a trajectory which spirals out. It seems to suggest a flaw in the model (if this trend continues, both populations may become unbounded, which is not the kind of behaviour we would expect from an isolated ecological system.) The cause of this behaviour becomes more clear if we consider the effect of decreasing the stepsize, as shown in the right panel of Figure 2.3, where stepsizes $h = 0.1$, $h = 0.01$ and $h = 0.001$ are compared. Evidently the growth of the solution is in fact a consequence of growing numerical error (also called discretization error). A solution with a small stepsize of $h = 0.001$ appears to close up, suggesting the correct trajectory is a periodic orbit.

This obviously raises a very serious question. Since many studies of dynamical systems now rely heavily on computer experiments, there is a serious risk of drawing false conclusions, when the numerical dynamics and the true continuous dynamics are doing different

things. A goal of this course is to understand what sort of properties a numerical method should have so that we can trust the computed results.

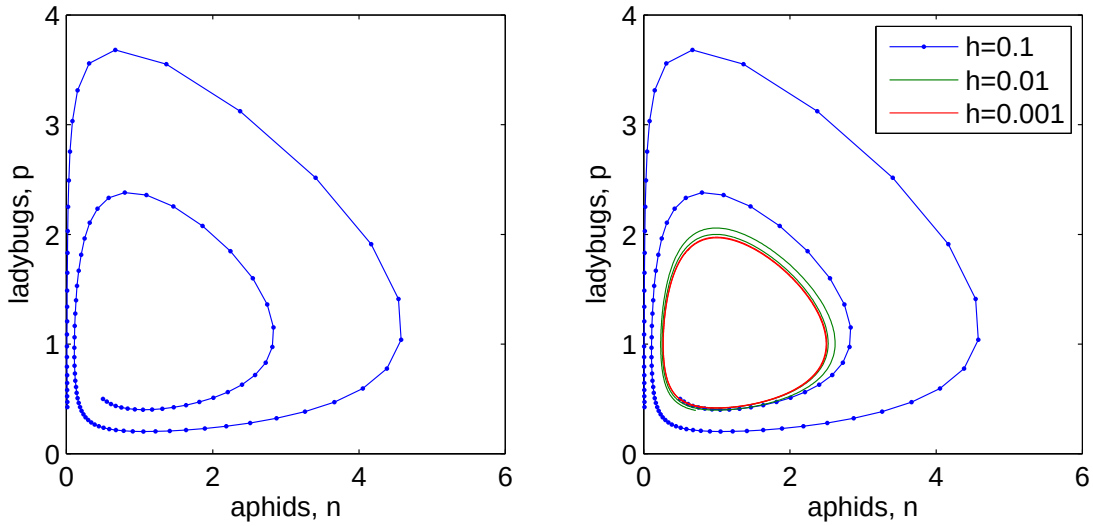

Figure 2.3: The locus curve of predators and prey for the Lotka-Volterra model, left with $h = 0.1$ and right, with $h = 0.1$ (100 steps), $h = 0.01$ (1000 steps) and $h = 0.001$ (10000 steps).

2.6 Convergence of Euler's Method

Euler's method employs a stepsize h which can be made as small as we like. If we fix the time interval $[a, b]$, then the stepsize is inversely proportional to the number of timesteps, i.e. $N = (b - a)/h$. This means that if we choose h very small we have to do a lot of work to compute our approximate solution.

In this section, we will prove that Euler's method "converges." This means that as h tends to zero the error tends to zero. This implies that we can approximate the solution with as much accuracy as we like if we are willing to pay the computational price.

We consider the scalar initial value problem

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), \qquad y(a) = y_0,$$

and suppose that a twice continuously differentiable and unique solution $y(t)$ exists on $[a, b]$. We will suppose that f is continuous everywhere with a continuous partial derivative with respect to y. (In practice this assumption could be relaxed later.) We will moreover assume that the partial derivative of f with respect to y is bounded in magnitude,

$$\left|\frac{\partial f}{\partial y}\right| \leq L. \tag{2.2}$$

We will assume the constant L is positive. (If $L = 0$ it means that f is actually independent of y, which gives a special case.)

A slight technicality is that, to reach the right endpoint $t_N = b$ exactly, we must have $N = (b-a)/h$, an integer number of steps. This means that if a and b are fixed, then h cannot take on all values but only integer fractions of $b-a$. We will therefore only consider the sequence of stepsizes defined by $(b-a)/N$ which goes to zero as we increase N.

Given N and with $h = (b-a)/N$, we consider the Euler numerical approximation defined by

$$y_{n+1} = y_n + hf(t_n, y_n), \qquad n = 0, 1, \ldots, N-1. \tag{2.3}$$

We would like to analyse how much error is introduced in performing this calculation. Define the error as

$$e_n = y_n - y(t_n).$$

We require a bound on the magnitude of the error $|e_n|$, $n = 0, 1, \ldots, N$.

Using the Taylor's series expansion of the solution, we have

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(\tau_n)$$

where τ_n is a point between t_n and t_{n+1}. Since $y'(t_n) = f(t_n, y(t_n))$ (because y is a solution of the differential equation), we may write

$$y(t_{n+1}) = y(t_n) + hf(t_n, y(t_n)) + \frac{h^2}{2}y''(\tau_n) \tag{2.4}$$

Notice that the first two terms on the right side of this equation are very similar to the form of Euler's method (which perhaps does not come as a surprise). Taking the difference of (2.3) and (2.4) we find that

$$y_{n+1} - y(t_{n+1}) = y_n + hf(t_n, y_n) - y(t_n) - hf(t_n, y(t_n)) - \frac{h^2}{2}y''(\tau_n).$$

Rearranging the expressions, we see that

$$e_{n+1} = e_n + h[f(t_n, y_n) - f(t_n, y(t_n))] - \frac{h^2}{2}y''(\tau_n).$$

Next, take absolute values of both sides and use the triangle inequality:

$$|e_{n+1}| \le |e_n| + h|f(t_n, y_n) - f(t_n, y(t_n))| + \frac{h^2}{2}|y''(\tau_n)|.$$

Because y'' is continuous, we know that there exists a value M such that

$$|y''(t)| \le M, \quad a \le t \le b. \tag{2.5}$$

This implies

$$|e_{n+1}| \le |e_n| + h|f(t_n, y_n) - f(t_n, y(t_n))| + \frac{h^2}{2}M. \tag{2.6}$$

The Mean Value Theorem tells us that

$$f(t_n, y_n) - f(t_n, y(t_n)) = \frac{\partial f}{\partial y}(t_n, \eta_n)(y_n - y(t_n))$$

where η_n lies between y_n and $y(t_n)$. Because of the bound on the derivative, we may now write, in place of (2.6):

$$|e_{n+1}| \leq |e_n| + hL|y_n - y(t_n)| + \frac{h^2}{2}M = (1 + hL)|e_n| + \frac{h^2}{2}M.$$

In this expression, we may view h, L, and M as constants. Thus we need to understand the recurrence inequality

$$v_{n+1} \leq Av_n + B.$$

where $A = 1 + hL$ and $B = h^2M/2$.

The following Lemma is extremely useful.

Lemma 2.6.1 *Suppose a given sequence of nonnegative numbers satisfies*

$$v_{n+1} \leq Av_n + B,$$

where A, B are nonnegative numbers and $A > 1$, then, for $n = 0, 1, \ldots$,

$$v_n \leq A^n v_0 + \frac{A^n - 1}{A - 1}B$$

This can be proved straightforwardly by induction: you are asked to prove a more general version in Worksheet 1. This type of result is an example of a *discrete Gronwall inequality*.

In this particular case, we have $v_0 = |y_0 - y(t_0)|$. By definition, $y(t)$ is the solution of the initial value problem with $y(t_0) = y_0$, so we know that $v_0 = 0$. This leaves the second part only. From the definitions of A and B we have

$$|e_n| \leq \left(\frac{(1 + hL)^n - 1}{1 + hL - 1}\right)\frac{h^2M}{2},$$

thus

$$|e_n| \leq [(1 + hL)^n - 1]\frac{M}{2L}h.$$

Next, we use the fact that, for positive hL,

$$1 + hL < 1 + hL + h^2L^2/2 + h^3L^3/3! + \ldots = e^{hL},$$

and

$$(1 + hL)^n - 1 < (1 + hL)^n < (e^{hL})^n = e^{nhL}.$$

Since $t_n - a = nh$, and $t_n \leq t_N = b$, we have

$$(1 + hL)^n - 1 < e^{(b-a)L}.$$

This is just a positive constant, and the important point is that it is independent of h. Hence we may write

$$|e_n| \leq e^{(b-a)L}\frac{M}{2L}h. \tag{2.7}$$

The quantity

$$D = e^{(b-a)L}\frac{M}{2L} \tag{2.8}$$

is just a positive constant, so we have proved

$$|e_n| \leq Dh, \qquad n = 0, 1, \ldots N. \tag{2.9}$$

This means that the error is bounded by Dh in magnitude. In particular we see that as h is reduced, the error tends to zero.

It must be stressed that D does not depend on N or h, but only on features of the problem and the solution.

We have shown the following theorem:

Theorem 2.6.1 *Consider the initial value problem*

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), \qquad y(a) = y_0,$$

and suppose there exists a unique, twice differentiable, solution $y(t)$ on $[a, b]$. Suppose further that f is continuous everywhere with a continuous, bounded partial derivative with respect to y:

$$\left|\frac{\partial f}{\partial y}\right| \leq L,$$

with $L > 0$. Then, for $n = 0, 1, \ldots, N$, and some constant D, the solution y_n given by Euler's method at t_n satisfies

$$|y_n - y(t_n)| \leq Dh,$$

where $h = (b - a)/N$ and $t_n = a + hn$.

2.6.1 Computational analysis of Euler's method

Knowing that Euler's method converges as the stepsize tends to zero, we return to one of the previous examples and study the behaviour in more detail. We consider the Logistic Differential Equation (1.4) with $r = 1$; that is, $\dot{y}(t) = y(t)\,(1 - y(t))$. For this example it is possible to compute the error since we have the exact solution for comparison.

Figure 2.4 shows the error, $y(t_n) - y_n$, at each time point; i.e., the difference between the exact and numerical solutions.

The maximum absolute value of the error on this interval turns out to be 0.0297. How does this change as we reduce the stepsize? We can just re-run the calculation, changing h and N so that hN does not change, and compute the maximum global error for each value of the stepsize h. In this way we construct a table of values (rounding to three significant figures):

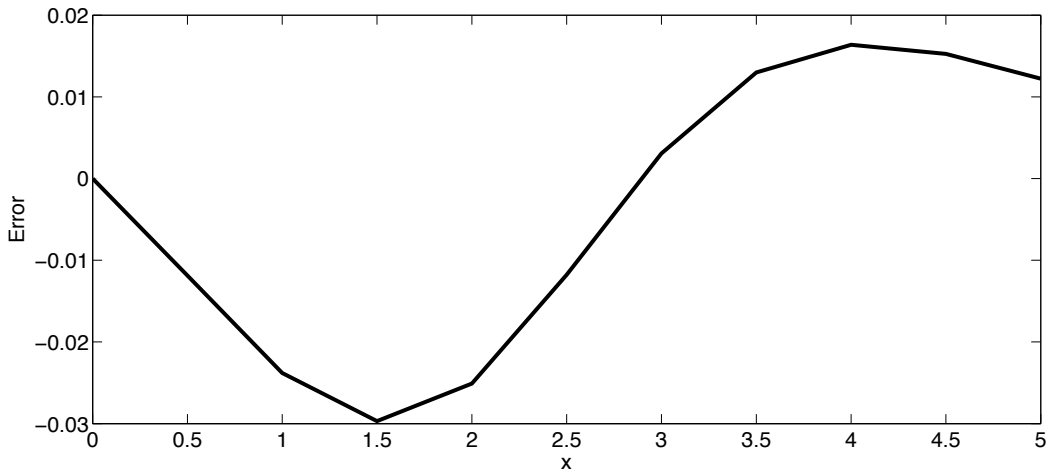

Figure 2.4: Error vs t for Euler solution of Logistic Differential Equation ($t_0 = 0, y_0 = 0.2, N = 10, h = 0.5$).

h	Max Global Error
1	0.0584
0.5	0.0297
0.25	0.0144
0.2	0.0115
0.125	0.00709

As we can see, the error is reduced as we reduce h. We can even see the proportional relationship between error and stepsize, by plotting the maximum global error vs the stepsize (Figure 2.5).

The global error bound (2.9) is useful in that it shows that the worst case error is diminished by reducing stepsize. It does not completely explain the result shown in Figure 2.5. This is because there could be a large difference between the actual error and the upper bound that we obtained. In practice, however, this bound is indicative of a linear relationship that is observed in practice, even if D as provided in the analysis is often a severe overestimate of the actual coefficient.

2.6.2 Twisty Example (again)

Review Figure 2.2 for the "twisty slope field" example. How does our new understanding about the convergence of Euler's method explain what is happening here? In complicated examples like this one, the error may be very large indeed for large stepsizes, only reaching a useful level when the stepsize is quite small. Here the smallest stepsize used, $h = 0.25$, is still not small enough to compute an accurate solution using Euler's method. We can see this very clearly if we graph it against the exact solution, as in Figure 2.6. Apparently the approximation is good only in the early going and quickly becomes poor towards the

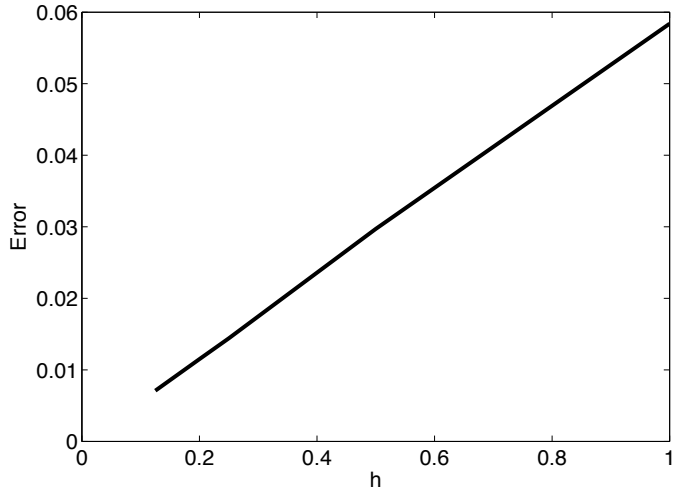

Figure 2.5: Maximum global error vs stepsize h for Euler solution of Logistic Differential Equation ($t_0 = 0, y_0 = 0.2$).

end of interval, as error is compounded.

In Figure 2.7 we have plotted the maximum global error against stepsize for the numerical solution of (2.1) on $[0, 20]$ started from $y(0) = 2$. It turns out to be useful to use a log-log scale to examine this, since small values of the stepsize would be squeezed together in a standard graph. If

$$\text{error} \approx Kh$$

then we can expect

$$\log \text{error} \approx \log K + \log h$$

this means that for a method whose maximum global error goes to zero like a multiple of h as $h \to 0$, we expect the graph in log-log scale to be a straight line with slope 1.

The linear scaling behaviour is only observed asymptotically for small values of the stepsize.

2.7 More Accurate Methods

Euler's method is a simple, practical scheme for solving ordinary differential equations, but it is not terribly accurate, meaning that it may require a lot of tiny timesteps to get a solution for which the error is sufficiently small. In later parts of the course, we will learn about some more accurate numerical methods. In order to do this, we will give a general method for analysing the global error of a method.

As a prerequisite, we need to introduce terminology that allows us to compare the accuracy of two methods.

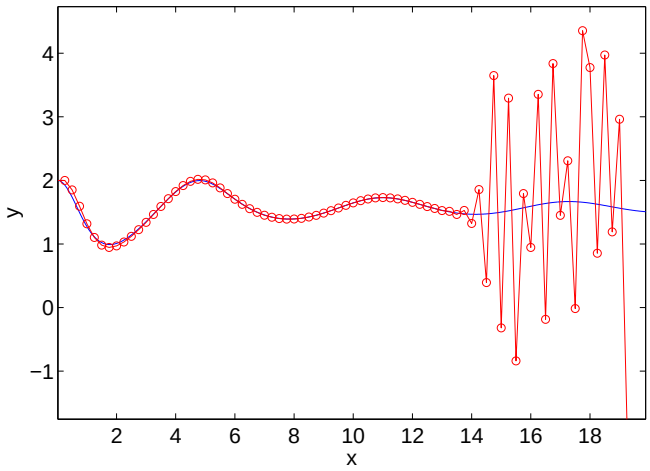

Figure 2.6: Numerical solution for the twisty slope field ($h = 0.25$) compared to the exact solution.

2.7.1 Order Notation

There is a convenient "Big O" or "Landau" notation for discussing the rate of convergence of a numerical method. We say that a quantity is $O(h^p)$ if it decays at least as quickly as h^p when h is small enough. More formally, we write $z = O(h^p)$ if there exist positive constants h_0 and C such that

$$|z| \leq Ch^p, \qquad \text{for all } 0 < h < h_0.$$

The $O(h^p)$ notation emphasizes that we are primarily interested in p rather than C, and we say that "z is of order p" or "z is of pth order" or "z is Big O of h to the p."

In our case, h represents the stepsize, which cannot be negative; hence we are always concerned with the limit $h \to 0$ from above. We will use the order notation to give insights into the behaviour of various types of error with respect to the stepsize.

We note that this order notation is based on a upper bound, and hence, for example, $z = O(h^{p+1}) \Rightarrow z = O(h^p)$. In practice, we usually assume that when writing expressions like $z = O(h^2)$ and $z = O(h^3)$, the largest possible value of p is being quoted.

2.7.2 The Flow Map

Consider a scalar initial value problem

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y), \quad y(a) = y_0, \tag{2.10}$$

which is assumed to have a unique solution on $[a, b]$.

Starting from any point t_0, y_0, we may follow the solution emanating from $y(t_0) = y_0$ up to $t \leq b$; i.e., we may write the solution as $y = y(t; t_0, y_0)$. Alternatively if we fix t_0 and

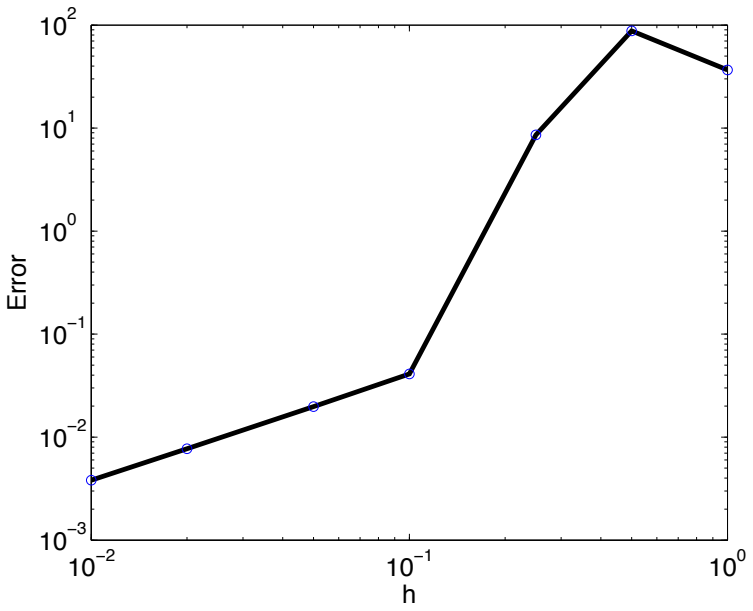

Figure 2.7: Errors for different stepsizes plotted in log-log scale.

h, we may consider the map from y_0 to $y(t_0 + h; t_0, y_0)$ in its own right. We call this the *flow map* Φ and write

$$y(t_0 + h; t_0, y_0) = \Phi_{t_0,h}(y_0).$$

It can be seen that this is actually a family of maps, parametrised by h.

Example.
The flow map for

$$\frac{\mathrm{d}y}{\mathrm{d}t} = ry,$$

is

$$\Phi_{t_0,h}(y_0) = e^{rh}y_0.$$

Since there is nothing special about t_0 and y_0, we may label these points t, y and write

$$\Phi_{t,h}(y) = e^{rh}y.$$

Example.
The flow map for

$$\frac{\mathrm{d}y}{\mathrm{d}t} = y\cos(t),$$

is

$$\Phi_{t,h}(y) = ye^{\sin(t+h)-\sin(t)}.$$

2.7.3 Numerical Methods as Flow Map Approximations

Determining the flow map is equivalent to solving the differential equation. The idea of a method such as Euler's method is to replace the flow map by a computable approximation. In Euler's method, we take the current state (t_n, y_n) and then we calculate the slope of the solution curve by $f(t_n, y_n)$ and approximate the value at t_{n+1} by $y_{n+1} = y_n + hf(t_n, y_n)$. We can view this as a flow map approximation of the form

$$\hat{\Phi}_{t,h}(y) = y + hf(t, y).$$

Note that we can view $f(t, y)$ as the derivative of the solution passing through the point (t, y), so we could write Euler's method as

$$y_{n+1} = y_n + h\dot{y}_n$$

where $\dot{y}_n$ represents the derivative of the solution passing through the point (t_n, y_n). In this way we see that Euler's method is like using a first order Taylor series approximation of the solution.

Methods like this that approximate the solution through iteration of an approximate flow map are called *one-step* methods.

2.7.4 Taylor Series Methods

Euler's method is based on the first two terms in a Taylor's series expansion. More generally, we can use a higher order expansion in powers of h. Continuing with the scalar case, we know that (assuming smoothness)

$$y(t_0 + h) = y(t_0) + y'(t_0)h + \frac{1}{2}y''(t_0)h^2 + \frac{1}{6}y'''(t_0)h^3 + \dots$$

Since

$$y' = f(t, y),$$

we have

$$y'' = \frac{d}{dt}y' = \frac{d}{dt}f(t, y).$$

Differentiating f partially with respect to each of its variables, and using the chain rule, we obtain

$$\begin{aligned} y'' &= \frac{\partial}{\partial t}f(t, y)\frac{dt}{dt} + \frac{\partial}{\partial y}f(t, y)\frac{dy}{dt} \\ &= \frac{\partial}{\partial t}f(t, y) + \frac{\partial}{\partial y}f(t, y)y'. \end{aligned}$$

We may write this as

$$y'' = f_t + f_y y' = f_t + f_y f.$$

Thus

$$\Phi_{t,h}(y) = y + hf(t,y) + \frac{1}{2}h^2\big(f_t(t,y) + f_y(t,y)f(t,y)\big) + \frac{1}{6}y'''h^3 + \ldots,$$

where we could also have computed y''' or any higher derivatives as functions of t and y. By truncating the Taylor expansion at a finite number of terms, we could hope to construct approximations of the solution of a given order of accuracy. The approximation that results allows the solution to be determined at successive points $a + h$ in terms of the solution at a (and the solution at $a + 2h$ in terms of values at $a + h$, etc.).

Note that even though we are talking about Taylor Series methods as if they generate approximations of solutions of differential equations, we have not actually shown this! We have only provided a convergence proof for Euler's method. We defer this until the section on convergence of one-step methods, a few lectures down the road. That theory will also cover **systems** of ODEs.

Because Taylor Series methods are easily constructed at all orders, including very high ones, they are particularly suited to simulations when very high accuracy is needed, e.g. they are used commonly to compute spacecraft trajectories. Orders over 20 are sometimes used in these cases.

Example.
We will apply a 2nd order Taylor Series method to the Logistic Differential Equation (with $r = 1$), and examine the performance compared to Euler's method as $h \to 0$.
The 2nd order Taylor Series method is

$$y_{n+1} = y_n + hf(t_n, y_n) + \frac{h^2}{2}\big(f_y(t_n, y_n)f(t_n, y_n) + f_t(t_n, y_n)\big).$$

The Logistic Differential Equation is

$$\frac{\mathrm{d}y}{\mathrm{d}t} = ry(1 - y), \qquad r > 0.$$

Taking $r = 1$, we compute the required derivatives:

$$f_t \equiv 0,$$

and

$$f_y = 1 - 2y.$$

Therefore the 2nd order Taylor series method can be written

$$y_{n+1} = y_n + hy_n(1 - y_n) + \frac{h^2}{2}(1 - 2y_n)y_n(1 - y_n).$$

We computed the solution starting from $y(0) = 0.2$ using both this and Euler's method with a stepsize of $h = 0.2$ for a total of 30 steps. We plot the numerical solutions together with the exact solution on the left in Figure 2.8. At first glance, the numerical and exact solutions look very similar–they sit right on top of each other–and it does not seem that the Taylor series method is actually better than the Euler method solution. However, a close-up of the right end of the interval shown on the right in the same figure demonstrates that indeed there is a remarkable and dramatic improvement from the higher order method.

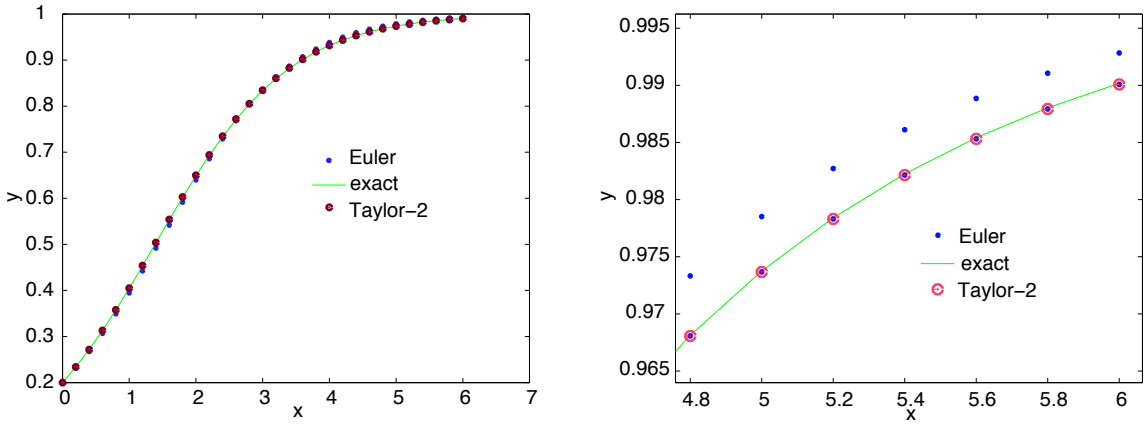

Figure 2.8: A solution of the Logistic Equation and numerical solutions computed using Euler's method and the 2nd order Taylor Series method. The figure on the right is a close up view of the right end of the trajectory.

Chapter 3

Convergence of One-Step Methods

Main concepts: In this chapter we prove the convergence of a large class of one-step methods to the exact flow map, and we discuss ways of designing higher order methods which may offer improved accuracy and better efficiency in numerical simulations, in particular Runge-Kutta methods.

Keywords: Taylor series, local error, convergence theory for one-step methods, interpolation, quadrature, collocation, Runge-Kutta methods, consistency, order conditions.

3.1 Approximation

For the numerical methods that we have seen so far, it is intuitively reasonable to expect that, overall, the accuracy of the method should increase if the stepsize h is made smaller. In particular, we would expect the overall error to tend to zero if we take the asymptotic limit $h \to 0$. (Note that using a smaller h increases the computational effort.) To make this notion precise, we will first define the error and then set up an appropriate asymptotic limit that allows us to define *convergence*. For ease of exposition, we focus on the case of an autonomous ODE system. The nonautonomous case is discussed briefly in subsection 3.8.

Definition 3.1.1 We define the **global error** after n time steps to be the difference between the discrete approximation and the exact solution

$$e_n := y_n - y(t_n). \tag{3.1}$$

□

A good approximation will produce a global error that is small in norm at each step, that is,

$$\max_{n=0,1,\ldots N} \|e_n\| \leq \delta,$$

for some user-specified tolerance δ (e.g., $\delta = 10^{-4}$). For a given vector field f (which we will generally assume to be Lipschitz), initial value y_0 and time interval T, we have only one free parameter, the timestep $h = T/N$. If the method is going to be useful, we must be able to meet any required tolerance by choosing h sufficiently small.

Definition 3.1.2 A method is said to be **convergent** if, for any given T,

$$\lim_{\substack{h\to 0,\\ h=T/N}} \max_{n=0,1,\ldots,N} \|e_n\| = 0.$$

□

It is important to understand the asymptotic limit used in this definition. By taking the maximum over $n = 0, 1, 2, \ldots, N$, we are considering the worst-case global error norm over all time points between 0 and T. Then, by taking the limit as h tends to zero, we look at the behaviour of this error measure for arbitrarily small stepsizes. Note that using a smaller stepsize, h, requires us to take more steps, N, to reach $T = Nh$. Hence, in our asymptotic limit, $N \to \infty$ as $h \to 0$.

Our aim is now to establish conditions for the convergence of a large class of one-step methods.

A tool of singular importance in numerical analysis is Taylor's series. For a *scalar function* ($d = 1$), assuming $y(t)$ is ν-times continuously differentiable, Taylor's theorem (with remainder) says there is a point $t^* \in [t, t + \tau]$ such that

$$y(t + \tau) = \sum_{i=0}^{\nu-1} \frac{\tau^i}{i!} \frac{\mathrm{d}^i y}{\mathrm{d}t^i}(t) + \frac{\tau^\nu}{\nu!} \frac{\mathrm{d}^\nu y}{\mathrm{d}t^\nu}(t^*).$$

For a vector function ($d > 1$), such a statement holds for each component, but in general the remainder term must be evaluated at a different t^* for each component. The norm of the last remainder term is therefore bounded on $[t, t + \tau]$, and we have

$$\left\| \frac{\tau^\nu}{\nu!} \frac{\mathrm{d}^\nu y}{\mathrm{d}t^\nu}(t) \right\| \leq C\tau^\nu.$$

So we may write

$$y(t + \tau) = \sum_{i=0}^{\nu-1} \frac{\tau^i}{i!} \frac{\mathrm{d}^i y}{\mathrm{d}t^i}(t) + \mathcal{O}(\tau^\nu) \tag{3.2}$$

in general.

3.2 Convergence of generalized one-step methods

We are considering an autonomous ODE system $y' = f(y)$. The solution may be described by a flow map $\Phi_t(y)$. Given that $y(t)$ is the solution at time t, we have $y(t+h) = \Phi_h(y(t))$. We assume that we are given a numerical method of one-step type, described by the map $\Psi_h(y)$. Obviously $\Psi_h(y)$ should be an approximation of $\Phi_h(y)$.

Definition 3.2.1 We define the **local error** of a one-step numerical method as the difference between the flow-map and its discrete approximation

$$le(y,h) = \Psi_h(y) - \Phi_h(y).$$

□

The local error measures just how much error is introduced in a single timestep of size h. Let us assume that, on our (invariant) domain of interest D, we can expand le in powers of h (typically using Taylor series).

Definition 3.2.2 Suppose the local error for a method satisfies

$$\|le(y,h)\| \leq Ch^{p+1}, \tag{3.3}$$

where C is a constant that depends on $y(t)$ and its derivatives, and $p \geq 1$. Then the method is said to be **consistent** at order p. □

Definition 3.2.3 Suppose that a method satisfies an h-dependent Lipschitz condition on D (the spatial domain)

$$\|\Psi_h(u) - \Psi_h(v)\| \leq (1 + h\hat{L})\|u - v\|, \qquad \forall u, v \in D. \tag{3.4}$$

Then the method is said to be **stable**. □

The constant $\hat{L}$ is not necessarily the same as the Lipschitz constant for the vector field.

We will now show that consistency and stability imply convergence. The global error in (3.1) can be viewed as the difference between n iterations of Ψ_h and n iterations of Φ_h. Thus

$$e_{n+1} = y_{n+1} - y(t_{n+1}) = \Psi_h(y_n) - \Phi_h(y(t_n)).$$

To this expression we now add and subtract $\Psi_h(y(t_n))$, which is the numerical solution started from a point on the solution trajectory, then take norms to obtain

$$\begin{aligned} \|e_{n+1}\| &= \|\Psi_h(y_n) - \Psi_h(y(t_n)) + \Psi_h(y(t_n)) - \Phi_h(y(t_n))\| \\ &\leq \|\Psi_h(y_n) - \Psi_h(y(t_n))\| + \|\Psi_h(y(t_n)) - \Phi_h(y(t_n))\|. \end{aligned}$$

Now we use our two assumptions (3.3) and (3.4) to write

$$\begin{aligned} \|e_{n+1}\| &\leq (1 + h\hat{L})\|y_n - y(t_n)\| + Ch^{p+1} \\ &= (1 + h\hat{L})\|e_n\| + Ch^{p+1}. \end{aligned}$$

Applying the inequality from Workshop Sheet 1 then yields

$$\|e_n\| \leq \frac{Ch^{p+1}}{\kappa - 1}(\kappa^n - 1) + \kappa^n\|e_0\|$$

where $\kappa = 1 + h\hat{L}$. Finally, since $1 + h\hat{L} \leq e^{h\hat{L}}$ and therefore $\kappa^n \leq e^{nh\hat{L}} = e^{T\hat{L}}$, if we assume the initial condition is exact ($e_0 = 0$), we get the uniform bound

$$\|e_n\| \leq h^p \frac{C}{\hat{L}}(e^{T\hat{L}} - 1), \tag{3.5}$$

which proves convergence at order p. The key point here is that $(C/\hat{L})(e^{T\hat{L}} - 1)$ does not depend upon h. We summarize this result in an important convergence theorem:

Theorem 3.2.1 (Convergence of One-Step Methods) *Given a differential equation (1.12) and a one-step numerical method Ψ_h which satisfies conditions (3.3) and (3.4) (consistency and stability), the global error satisfies*

$$\max_{n=0,\ldots,N} \|e_n\| = \mathcal{O}(h^p).$$

This theorem is powerful. Without specifying anything about the construction of the method, it guarantees the convergence of any one-step method that is consistent and satisfies an h-dependent Lipschitz condition.

As an example, let us use Theorem 3.2.1 to prove the convergence of Euler's method for smooth vector fields f. Consider a compact domain $D \subset \mathbb{R}^d$ and suppose f is smooth on D and has Lipschitz constant L on D (since f is smooth, we can take $L = \max_D \|\frac{\partial f}{\partial y}\|$). Then, since

$$\|\Psi_h(y) - \Psi_h(z)\| = \|y + hf(y) - (z + hf(z))\| \leq \|y - z\| + hL\|y - z\| = (1 + hL)\|y - z\|,$$

the numerical flow map is Lipschitz with $\hat{L} = L$.

The exact solution satisfies

$$\Phi_h(y) = y + h\frac{\mathrm{d}y}{\mathrm{d}t} + \frac{h^2}{2}\frac{\mathrm{d}^2y}{\mathrm{d}t^2} + \mathcal{O}(h^3) = y + hf(y) + \frac{h^2}{2}\frac{\mathrm{d}^2y}{\mathrm{d}t^2} + \mathcal{O}(h^3).$$

Therefore the local error is

$$le(y,h) = y + hf(y) - \left[y + hf(y) + \frac{h^2}{2}\frac{\mathrm{d}^2y}{\mathrm{d}t^2} + \mathcal{O}(h^3)\right] = \mathcal{O}(h^2),$$

and we can apply Theorem 3.2.1 with $C = \max_D \|\frac{\mathrm{d}^2y}{\mathrm{d}t^2}\|$ to show that Euler's method is convergent with order $p = 1$.

In the proof of the Theorem 3.2.1, the relation (3.5) indicates that the magnitude of the global error bound will be reduced in proportion to h^p. For example, when using Euler's method in practice, we typically observe that reducing the stepsize by a factor of 10 also reduces the global error by a factor of around 10. We say for this reason that Euler's method has order equal to one. The error incurred by Euler's method in each time step is $\mathcal{O}(h^2)$. The loss of one order of h occurs because the number of time steps needed to cover a fixed interval of length T increases as $h \to 0$ at a rate proportional to $1/h$.

Generally, we say that a one-step method has order p (or is "of pth order") if it satisfies the stability condition (3.4) and if $p \geq 1$ is the largest integer for which (3.3) holds.

Motivated by Theorem 3.2.1, we now look the design of one-step methods that are consistent at higher values of p.

3.3 Construction of more general one-step methods

One-step methods for an autonomous scalar initial value problem

$$\mathrm{d}y/\mathrm{d}t = f(y), \quad y(t_0) = y_0,$$

can be constructed in a variety of ways. A natural approach is to integrate both sides of the differential equation over one timestep,

$$y(t+h) - y(t) = \int_t^{t+h} f(y(\tau))\mathrm{d}\tau,$$

and then to approximate the integral on the right by a finite sum. For example, the simplest approximation is

$$\int_t^{t+h} f(y(\tau))\mathrm{d}\tau \approx hf(y(t)). \tag{3.6}$$

The general idea in discretization is to replace the approximate relation satisfied by the true solution

$$y(t+h) - y(t) \approx hf(y(t))$$

by an exact equation relating the approximate values,

$$y_{n+1} - y_n = hf(y_n).$$

In this case, we see that the choice (3.6) leads to Euler's method.

A numerical method to approximate a definite integral of one independent variable is known as a *numerical quadrature rule*. We can approximate the integral by any of a variety of different numerical quadrature methods, yielding in this way various methods which may have better accuracy, or other favorable properties. For example, the *trapezoidal rule* (also referred to as *trapezium rule*) approximation to the integral is

$$\int_t^{t+h} f(y(\tau))\mathrm{d}\tau \approx \frac{h}{2}\Big(f(y(t)) + f(y(t+h))\Big),$$

which results in the trapezoidal rule numerical method

$$y_{n+1} = y_n + h\big(f(y_n) + f(y_{n+1})\big)/2.$$

Note that this is an *implicit* method, meaning that the method is defined implicitly by an equation that has to be solved to advance the step (this is because the right hand side contains a function that depends on y_{n+1}).

In this chapter we introduce one-step methods derived using quadrature over polynomial interpolants, using collocation. We begin with a review of polynomial interpolation.

3.4 Polynomial interpolation

Let $\mathbb{P}_s$ denote the space of real polynomials of degree $\leq s$. Given a set of s distinct *abscissa points* $c_1 < \cdots < c_s$, $c_i \in \mathbb{R}$, and corresponding *data* $g_1, \ldots, g_s$, there exists a unique polynomial $P(x) \in \mathbb{P}_{s-1}$ satisfying $P(c_i) = g_i$, $i = 1, \ldots, s$. This polynomial is called the **interpolating polynomial**.

The Lagrange interpolating polynomials ℓ_i, $i = 1, \ldots, s$ for a set of abscissae are defined by

$$\ell_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{s} \frac{x - c_j}{c_i - c_j}. \tag{3.7}$$

The ith Lagrange interpolating polynomial ℓ_i is the interpolating polynomial corresponding to the data $\{g_i = 1; g_j = 0, j \neq i\}$, The set of Lagrange interpolating polynomials form a basis for $\mathbb{P}_s$. In this basis, the interpolating polynomial $P(x)$ assumes the simple form

$$P(x) = \sum_{i=1}^{s} g_i \ell_i(x). \tag{3.8}$$

3.5 Numerical quadrature

Now consider a smooth function g on the real line. We can approximate the definite integral of g on the interval $[0, 1]$ by exactly integrating the interpolating polynomial of order $s - 1$ based on s points $0 \leq c_1 < c_2 < \ldots < c_s \leq 1$. The points c_i are then known as *quadrature points*. The data are the values of g at the quadrature points $g_i = g(c_i)$, $i = 1, \ldots, s$.

Defining the weights

$$b_i = \int_0^1 \ell_i(x)\,\mathrm{d}x,$$

the quadrature formula becomes

$$\int_0^1 g(x)\,\mathrm{d}x \approx \int_0^1 P(x)\,\mathrm{d}x = \sum_{i=1}^{s} b_i g(c_i). \tag{3.9}$$

To approximate the integral $\int_{t_0}^{t_0+h} g(t)\,dt$, we define a function $t(x) = t_0 + hx$. Then $dt = h\,dx$, and (3.9) becomes

$$\int_{t_0}^{t_0+h} g(t)\,\mathrm{d}t = \int_0^1 g(t_0 + hx)\,h\mathrm{d}x \approx h \sum_{i=1}^{s} b_i g(t_0 + hc_i). \tag{3.10}$$

By construction, a quadrature formula using s distinct abscissa points will exactly integrate any polynomial in $\mathbb{P}_{s-1}$. However, with a judicious choice of the abscissae we can do even better. We say that a quadrature rule has *order* p if it exactly integrates any polynomial in $\mathbb{P}_{p-1}$. It can be shown that $p \geq s$ always holds, and, for optimal choice of the c_i, one has $p = 2s$.

3.6 One-step collocation methods

A very elegant approach to construct one-step methods of given order of accuracy uses the idea of *collocation*. We will construct the method for the first time step interval $[t_0, t_0 + h] = [t_0, t_1]$.

Let $0 \le c_1 < c_2 < \cdots < c_s \le 1$ be distinct nodes on the unit interval. The *collocation polynomial* $u(t) \in \mathbb{R}^d$ is a (vector-valued) polynomial of degree s satisfying

$$\begin{aligned} u(t_0) &= y_0 \\ u'(t_0 + c_i h) &= f(u(t_0 + c_i h)), \quad i = 1, \ldots, s \end{aligned} \tag{3.11}$$

and the numerical solution of the *collocation method* over the interval $[t_0, t_0 + h]$ is given by $y_1 = u(t_0 + h)$. In other words, we construct a polynomial that passes through y_0 and agrees with the ODE (1.3) at s nodes on $[t_0, t_0 + h]$. Then we let the numerical solution be the value of this polynomial at $t_0 + h = t_1$. See Figure 3.1.

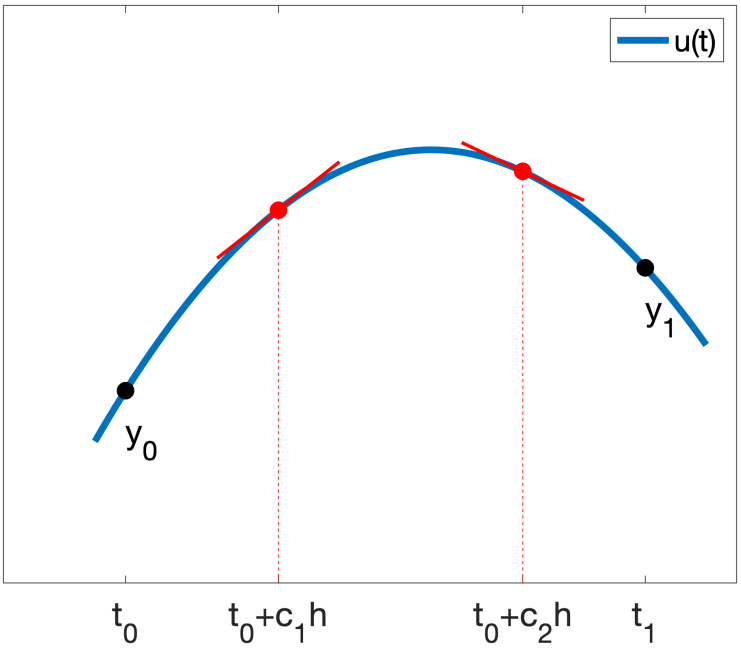

Figure 3.1: . Here, we collocate at $s = 2$ nodes. The collocation polynomial $u(t)$ satisfies the ODE at $t_0 + c_1 h$ and $t_0 + c_2 h$.

Let F_i be the values of the (as yet undetermined) interpolating polynomial at the nodes:

$$F_i := u'(t_0 + c_i h), \quad i = 1, \ldots, s.$$

Now we use the Lagrange interpolation formula (3.8) to define the polynomial $u'(t)$ passing through these points:

$$u'(t) = \sum_{i=1}^{s} F_i \ell_i\Big(\frac{t - t_0}{h}\Big). \tag{3.12}$$

Integrating this equation over the intervals $[t_0, t_0 + c_i h]$ (and introducing the change of variables $x = (t - t_0)/h$, i.e., $\mathrm{d}t = h\,\mathrm{d}x$) gives

$$u(t_0 + c_i h) = y_0 + h \sum_{j=1}^{s} F_j \int_0^{c_i} \ell_j(x)\,\mathrm{d}x, \quad i = 1, \ldots, s. \tag{3.13}$$

Let us denote

$$a_{ij} := \int_0^{c_i} \ell_j(x)\,\mathrm{d}x, \qquad b_i := \int_0^1 \ell_i(x)\,\mathrm{d}x, \qquad i, j = 1, \ldots, s.$$

Substituting (3.13) into the collocation conditions (3.11) gives

$$F_i = f\Big(y_0 + h \sum_{j=1}^{s} a_{ij} F_j\Big), \qquad i = 1, \ldots, s.$$

Similarly integrating (3.12) over $[t_0, t_0 + h]$ gives

$$y_1 := u(t_0 + h) = y_0 + h \sum_{i=1}^{s} F_i \int_0^1 \ell_i(x)\,dx = y_0 + h \sum_{i=1}^{s} b_i F_i.$$

To summarize, the collocation method is written

$$F_i = f\Big(y_n + h \sum_{j=1}^{s} a_{ij} F_j\Big), \quad i = 1, \ldots, s, \tag{3.14a}$$

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i F_i. \tag{3.14b}$$

One first solves the coupled, sd-dimensional nonlinear system (3.14a). Subsequently, the update (3.14b) is explicit.

Remark 3.6.1 *Collocation methods have the useful feature that we obtain a continuous approximation of the solution $u(t)$ on each interval $[t_n, t_{n+1}]$.*

Remark 3.6.2 *In terms of order of accuracy, the optimal choice is attained by using so-called Gauss-Legendre collocation methods and placement of the nodes at the roots of a shifted Legendre polynomial. For $s = 1$, 2 and 3, the quadrature points are:*

$$\begin{aligned} c_1 &= \tfrac{1}{2}, && p = 2, \\ c_1 &= \tfrac{1}{2} - \tfrac{\sqrt{3}}{6},\ c_2 = \tfrac{1}{2} + \tfrac{\sqrt{3}}{6}, && p = 4, \\ c_1 &= \tfrac{1}{2} - \tfrac{\sqrt{15}}{10},\ c_2 = \tfrac{1}{2},\ c_3 = \tfrac{1}{2} + \tfrac{\sqrt{15}}{10}, && p = 6. \end{aligned}$$

3.7 The Family of Runge-Kutta Methods

Recall from the previous chapter the definition of a collocation method (3.14). We saw that in these collocation methods, the coefficients b_i and a_{ij} are defined by certain integrals of the Lagrange interpolating polynomials associated with a chosen set of quadrature nodes or *abscissae* c_i, $i = 1, \ldots, s$.

A natural generalization of collocation methods is obtained by allowing the coeffcients c_i, b_i, and a_{ij}, $i, j = 1, \ldots, s$ to take on arbitrary values, not necessarily related to quadrature rules. In fact, we no longer assume the c_i to be distinct[1]. The result is the class of **Runge-Kutta methods**. The formulas above still hold; however, an alternative form introduces the **stage values** Y_i which can be viewed as intermediate values of the solution y at time $t_n + c_i h$ and are defined by $Y_i = y_n + h \sum_j a_{ij} F_j$, so that $F_i = f(Y_i)$ holds for each i. We hence define a Runge-Kutta method to be any method of the form

$$Y_i = y_n + h \sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, \ldots, s, \tag{3.15a}$$

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i f(Y_i). \tag{3.15b}$$

Here, s is termed the *number of stages* of the Runge-Kutta method, the b_i are the *weights*, and a_{ij} are the *internal coefficients*. We emphasize that (3.15) is simply an alternative way to write (3.14), but in (3.15) we are no longer assuming that the coefficients arise from a collocation process.

It is easy to see that with this definition, Euler's method and trapezoidal rule are Runge-Kutta methods. For example Euler's method can be put into the form (3.15a)-(3.15b) with $s = 1$, $b_1 = 1$, $a_{11} = 0$ (and $c_1 = 0$). The trapezoidal rule has $s = 2$, $b_1 = b_2 = 1/2$, $a_{11} = a_{12} = 0$, $a_{21} = a_{22} = 1/2$ (and $c_1 = 0$, $c_2 = 1$).

Each Runge-Kutta method generates an approximation of the flow map. To see this, note that from the formula (3.15a) it follows that each Y_i has a functional dependence on y_n and h, $Y_i = Y_i(y_n, h)$. Thus the Runge-Kutta method can be viewed as

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i f(Y_i(y_n, h)),$$

and the method generates a discrete flow-map approximation

$$\Psi_h(y) = y + h \sum_{i=1}^{s} b_i f(Y_i(y, h)).$$

[1]In the formulation for autonomous ODEs, the c_i do not appear explicitly. However, we will assume the internal consistency condition $c_i = \sum_{j=1}^{s} a_{ij}$ to hold throughout these lecture notes.

3.7.1 Some Runge-Kutta methods

Let $s = 2$ and choose $c_1 = 0$ and $c_2 = 2/3$, $b_1 = 1/4$, $b_2 = 3/4$. We need $Y_1 \approx y(t_n)$ and $Y_2 \approx y(t_n + \frac{2}{3}h)$. The latter can be obtained using an Euler step:

$$\begin{aligned} Y_1 &= y_n, \\ Y_2 &= y_n + \frac{2}{3}hf(Y_1), \\ y_{n+1} &= y_n + h\left(\frac{1}{4}f(Y_1) + \frac{3}{4}f(Y_2)\right). \end{aligned} \tag{3.16}$$

Another example with $s = 2$ is the popular method due to Heun:

$$\begin{aligned} Y_1 &= y_n, \\ Y_2 &= y_n + hf(Y_1), \\ y_{n+1} &= y_n + \frac{h}{2}\big(f(Y_1) + f(Y_2)\big). \end{aligned} \tag{3.17}$$

The most famous Runge-Kutta method has four stages (this method is sometimes referred to as *the* Runge-Kutta method or RK4):

$$\begin{aligned} Y_1 &= y_n, \\ Y_2 &= y_n + \frac{h}{2}f(Y_1), \\ Y_3 &= y_n + \frac{h}{2}f(Y_2), \\ Y_4 &= y_n + hf(Y_3), \\ y_{n+1} &= y_n + h\left(\frac{1}{6}f(Y_1) + \frac{1}{3}f(Y_2) + \frac{1}{3}f(Y_3) + \frac{1}{6}f(Y_4)\right). \end{aligned} \tag{3.18}$$

The formulas above are often represented schematically in a *Butcher table*:

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} \quad = \quad \begin{array}{c|ccc} c_1 & a_{11} & \cdots & a_{1s} \\ \vdots & \vdots & & \vdots \\ c_s & a_{s1} & \cdots & a_{ss} \\ \hline & b_1 & \cdots & b_s \end{array}$$

where $A = (a_{ij})$, $b = (b_i)$ and $c = (c_i)$. For instance, the methods (3.16), (3.17) and (3.18) listed in this section are represented by

$$\begin{array}{c|cc} 0 & & \\ 2/3 & 2/3 & \\ \hline & 1/4 & 3/4 \end{array} \qquad \begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array} \qquad \begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

It is standard practice to leave the trailing zero elements in the matrix A blank in the Butcher table.

The previous examples of this subsection are all explicit methods. Any Runge-Kutta method for which the matrix $A = (a_{ij})$ is strictly lower triangular is said to be *explicit*. For explicit methods, the stage values Y_i and final approximation y_{n+1} may be computed in order, without the need to solve any equations.

If A has a nonzero on or above the diagonal, then we say that the Runge-Kutta method is *implicit*. In this case we need to solve one or more equations in order to find y_{n+1}.

A simple example of an implicit RK method is the *implicit midpoint rule*

$$\begin{array}{c|c} 1/2 & 1/2 \\ \hline & 1 \end{array}, \quad Y_1 = y_n + \frac{h}{2} f(Y_1), \quad y_{n+1} = y_n + hf(Y_1)$$

which can also be written in simplified form as (just substitute $Y_1 = (y_{n+1} + y_n)/2$ in the above wherever it occurs):

$$y_{n+1} = y_n + hf\Big(\frac{y_{n+1} + y_n}{2}\Big). \tag{3.19}$$

3.7.2 Accuracy of Runge-Kutta methods

We will now investigate the accuracy of the methods introduced above in terms of the order of consistency.

We are studying an autonomous ODE system, with RHS given by the function $f(y) : \mathbb{R}^d \to \mathbb{R}^d$. In order to perform Taylor expansions, we will introduce some compact notation. We will denote by f' the Jacobian, that is, the matrix of partial derivatives:

$$f' = \Big(\frac{\partial f_i}{\partial y_j}\Big), \quad 1 \le i \le d, \quad 1 \le j \le d.$$

So, when evaluated at any point y, the Jacobian $f'(y)$ is a matrix in $\mathbb{R}^{d\times d}$.

We may then regard f'' as a three-dimensional array of second order partial derivatives:

$$f'' = \Big(\frac{\partial^2 f_i}{\partial y_j \partial y_k}\Big), \quad 1 \le i \le d, \quad 1 \le j \le d, \quad 1 \le k \le d.$$

Given two vectors a and b in $\mathbb{R}^d$, for fixed i we can regard

$$\frac{\partial^2 f_i}{\partial y_j \partial y_k}$$

as an operator that maps a and b into a real number:

$$\sum_{j=1}^{d} \sum_{k=1}^{d} \frac{\partial^2 f_i}{\partial y_j \partial y_k} a_j b_k.$$

We can do this for every component i to get a vector, which we write as $f''(a,b)$. (More formally, f'' is a bilinear operator that maps two vectors into a single vector.) We use this notation to obtain useful expressions for $y'(t)$ and $y''(t)$. First we note that, by construction,

$$y_i''(t) = \frac{d}{dt} y'_i(t) = \frac{d}{dt} f_i(y).$$

Since y is a function of t, we must now apply the chain rule to obtain

$$\begin{aligned} y_i''(t) &= \sum_{j=1}^{d} \frac{\partial f_i}{\partial y_j} y_j'(t) \\ &= \sum_{j=1}^{d} \frac{\partial f_i}{\partial y_j} f(y(t))_j. \end{aligned} \tag{3.20}$$

The right hand side is the ith component of the matrix-vector product $f'(y)f(y)$. Hence, we may write $y''(t) = f'f$.

Differentiating (3.20) with respect to t and again applying the chain rule, we obtain

$$y_i'''(t) = \sum_{j=1}^{d} \sum_{k=1}^{d} \frac{\partial^2 f_i}{\partial y_j \partial y_k} y_k'(t) y_j'(t) + \sum_{j=1}^{d} \frac{\partial f_i}{\partial y_j} y_j''(t).$$

This expression matches the ith component of $f''(y', y') + f'f'f$. Hence, we may write $y''' = f''(f,f) + f'f'f$.

In summary we have shown that

$$\begin{aligned} y' &= f, \\ y'' &= f'f, \\ y''' &= f''(f,f) + f'f'f. \end{aligned}$$

In all cases, y and its derivatives are assumed to be evaluated at t, and f and its derivatives at y. The process may clearly be continued in order to obtain expressions for higher order time derivatives of y in terms of f and its partial derivatives. We will stop at y''' and briefly mention the general case later.

For the exact solution of the ODE we have the Taylor series expansion

$$y(t+h) = y(t) + hy'(t) + \frac{1}{2}h^2 y''(t) + \frac{1}{6}h^3 y'''(t) + O(h^4).$$

In terms of the flow map, and using our expressions for the time derivatives, this may be written

$$\Phi_h(y) = y + hf + \frac{h^2}{2} f'f + \frac{h^3}{6}\left[f''(f,f) + f'f'f\right] + \mathcal{O}(h^4). \tag{3.21}$$

The same type of expansion procedure can be carried out for a Runge-Kutta method. For example, for Euler's method, we have simply

$$\Psi_h(y) = y + hf(y).$$

This means that the discrete and continuous series match to $O(h^2)$ and the local error expansion can be written

$$le(y,h) = \frac{h^2}{2} f'f + O(h^3).$$

Again, we have dropped the argument y of f for notational compactness.

For the trapezoidal rule (and other implicit schemes) the derivatives need to be computed by implicit differentiation. For simplicity, with y fixed, write $z = z(h) = \Psi_h(y)$. Then $z(h)$ must satisfy the implicit relation

$$z = y + \frac{h}{2}\big(f(y) + f(z)\big).$$

Observe that $z(0) = y$. Next we differentiate the expression for z,

$$z' = \mathrm{d}z/\mathrm{d}h = \frac{1}{2}\big(f(y) + f(z)\big) + \frac{h}{2} f'(z)z'. \tag{3.22}$$

(Note that z does not satisfy the differential equation, so we *cannot* replace z' here by $f(z)$!) For $h = 0$, the last term of (3.22) vanishes and we have

$$z'(0) = f(y).$$

Differentiate (3.22) once more with respect to h to obtain

$$z'' = \frac{1}{2} f'(z)z' + \frac{1}{2} f'(z)z' + \frac{h}{2}\big(f''(z)(z',z') + f'(z)z''\big), \tag{3.23}$$

for $h = 0$, the last term of (3.23) drops out and we have

$$z''(0) = f'(z(0))z'(0) = f'(y)f(y).$$

Using these expressions, we may write the first few terms of the Taylor series in h of $\Psi_h(y)$:

$$\Psi_h(y) = z(h) = z(0) + hz'(0) + \frac{h^2}{2} z''(0) + \ldots = y + hf + \frac{h^2}{2} f'f + \ldots \tag{3.24}$$

Comparing this with the expansion for the exact solution (3.21) we see that the first few terms are identical. Thus the local error vanishes to *at least* $O(h^3)$. To get the next term in the local error expansion, we differentiate (3.23):

$$\begin{aligned} z''' \;=\; & f''(z)(z',z') + f'(z)z'' + (1/2)\left(f''(z)(z',z') + f'(z)z''\right) + \\ & \frac{h}{2}\big(f'''(z)(z',z',z') + 2f''(z)(z',z'') + f''(z)(z',z'') + f'(z)z'''\big), \end{aligned}$$

which evaluates to

$$z'''(0) = (3/2)\left(f''(f,f) + f'f'f\right),$$

which means that the expansions (3.24) and (3.21) *do not* match in the term of 3rd order: for the trapezoidal rule, we have

$$le(y,h) = -\frac{1}{12}\left(f''(f,f) + f'f'f\right)h^3 + \mathcal{O}(h^4).$$

3.7.3 Order conditions for Runge-Kutta methods

We now show that the type of analysis above, where we studied Euler's method and the trapezoidal rule, can be extended to any method within the Runge-Kutta class. In this more general setting, we just need to compute the derivatives of $\Psi_h(y) = z(h)$. To do this we write the general RK method in the following form:

$$z(h) = y + h\sum_{i=1}^{s} b_i F_i,$$

where

$$F_i = f(Y_i), \tag{3.25}$$

$$Y_i = y + h\sum_{j=1}^{s} a_{ij} F_j. \tag{3.26}$$

Now the calculations required are straightforward, viewing F_i, Y_i, and z as functions of h with y assumed fixed (i.e. the prime denotes differentiation with respect to h):

$$z' = \sum_{i=1}^{s} b_i F_i + h\sum_{i=1}^{s} b_i F_i',$$

which immediately tells us that

$$z'(0) = \sum_{i=1}^{s} b_i F_i(0) = \sum_{i=1}^{s} b_i f(Y_i(0)) = \Big(\sum_{i=1}^{s} b_i\Big) f(y).$$

For the method to have order $p = 1$, we must have

$$\sum_{i=1}^{s} b_i = 1. \tag{3.27}$$

This is the first of what are termed the "order conditions". Any RK method satisfying (3.27) is consistent, and by Theorem 3.2.1, convergent independent of the a_{ij}. To get the next one, we compute the second derivative of z:

$$\begin{aligned} z'' &= \sum_{i=1}^{s} b_i F_i' + \sum_{i=1}^{s} b_i F_i' + h\sum_{i=1}^{s} b_i F_i'' \\ &= 2\sum_{i=1}^{s} b_i F_i' + h\sum_{i=1}^{s} b_i F_i''. \end{aligned} \tag{3.28}$$

From (3.25) and (3.26) we know that

$$\begin{aligned} F_i' &= f'(Y_i) Y_i', \\ Y_i' &= \sum_{j=1}^{s} a_{ij} F_j + h\sum_{j=1}^{s} a_{ij} F_j'. \end{aligned}$$

At $h = 0$, we have

$$Y_i'(0) = \sum_{j=1}^{s} a_{ij} F_j(0) = \sum_{j=1}^{s} a_{ij} f(y).$$

Let $c_i = \sum_{j=1}^{s} a_{ij}$, so $Y_i'(0) = c_i f(y)$. Then

$$F_i'(0) = c_i f'(y) f(y)$$

and hence

$$z''(0) = 2\Big(\sum_{i=1}^{s} b_i c_i \Big) f'(y) f(y).$$

For this term to cancel with y'', we must have

$$\sum_{i=1}^{s} b_i c_i = \frac{1}{2}. \tag{3.29}$$

For $p = 2$, we must therefore have *both* conditions (3.27) and (3.29).

To continue this process, we would need to differentiate (3.28). This is a bit tedious. To make it a little simpler, we will not bother to compute terms which vanish when we evaluate at $h = 0$:

$$z''' = 3 \sum_{i=1}^{s} b_i F_i'' + h(\ldots).$$

Then

$$F_i'' = f''(Y_i)(Y_i', Y_i') + f'(Y_i) Y_i''$$

$$Y_i'' = 2 \sum_{j=1}^{s} a_{ij} F_j' + h(\ldots).$$

Now we evaluate everything at $h = 0$ and find

$$z'''(0) = 3 \sum_{i=1}^{s} b_i \Big(f''(Y_i(0))(Y_i'(0), Y_i'(0)) + f'(Y_i(0)) Y_i''(0) \Big).$$

Recall from the earlier work that $Y_i(0) = y$, $Y_i'(0) = c_i f(y)$, and we also have

$$Y_i'' = 2 \sum_{j=1}^{s} a_{ij} F_j'(0) = 2 \sum_{j=1}^{s} a_{ij} c_j f'(y) f(y).$$

So finally,

$$z'''(0) = 3 \sum_{i=1}^{s} b_i \Big(c_i^2 f''(y)(f(y), f(y)) + 2\Big(\sum_{j=1}^{s} a_{ij} c_j \Big) f'(y) f'(y) f(y) \Big),$$

which means that, for the method to have order $p = 3$, we must have *two* more conditions satisfied (in addition to (3.27) and (3.29)):

$$\sum_{i=1}^{s} b_i c_i^2 = \frac{1}{3}, \qquad \sum_{i=1}^{s}\sum_{j=1}^{s} b_i a_{ij} c_j = \frac{1}{6}.$$

The number of order conditions explodes with increasing order. For order 4, there are four additional conditions or 8 in all, there are 17 at order 5, 37 at order 6, 85 at order 7, and 200 at order 8. The task of enumerating the conditions can be greatly simplified by using techniques from graph theory ("Butcher trees," the ingenious invention of J. C. Butcher from New Zealand who has pioneered much of the study of Runge-Kutta methods). In order to actually find methods, it is not enough to work out the order conditions: we need to identify sets of coefficients, a_{ij} and b_i which allow us to exactly satisfy a large system of multivariate polynomials. Butcher also developed a set of simplifying conditions, which greatly ease the development of higher order methods.

There exist explicit methods of order $p = s$ for up to four stages ($s \leq 4$). It can be shown that the conditions for order 5 cannot be solved with only 5 stages under the restriction that the method be explicit (i.e. $a_{ij} = 0$, $j \geq i$). The Gauss-Legendre RK methods, which were mentioned earlier in this chapter, achieve the maximum possible order for s-stage (implicit) Runge-Kutta methods; they have order $p = 2s$.

3.8 Nonautonomous differential equations

The methods of this chapter are easily extended to treat the case that the system is nonautonomous, $y' = f(t, y)$. The easiest way to understand how to do this is to use the method mentioned at the end of Section 1.1 which extends the system of equations to include an additional variable whose derivative is 1. This turns nonautonomous differential equation systems into autonomous ones with one additional variable. Then the Runge-Kutta methods of this chapter are directly applicable. We find that the time variable corresponding to the ith stage is $t_n + c_i h$.

Let us write out the equations of an RK method applied in the nonautonomous case:

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i f(t_n + c_i h, Y_i)$$

where, for $i = 1, 2, \ldots, s$,

$$Y_i = y_n + h \sum_{j=1}^{s} a_{ij} f(t_n + c_j h, Y_j).$$

With the simplifying assumption $\sum_{j=1}^{s} a_{ij} = c_i$, the order conditions mentioned in this chapter are unchanged for the nonautonomous case.

Chapter 4

Stability of Runge-Kutta Methods

Main concepts: Fixed points (equilibrium points), fixed points for Runge-Kutta methods, stability of fixed points, stability of maps, Runge-Kutta stability function, stability domain.

Keywords: Fixed point, stability, linearization, stability region, A-stability.

A lot can be said about the qualitative behaviour of ODEs by looking at the local solution behaviour in the neighbourhood of fixed points. Similarly, we can study the qualitative behaviour of Runge-Kutta methods by looking at fixed points and linearized problems.

In this chapter, we are concerned with issues around long-time behaviour; hence the asymptotic limit that we have in mind is $n \to \infty$ with the stepsize h fixed.

Throughout the chapter, we also assume that the ODE system is autonomous.

The theory of dynamical systems is rich and we cannot hope to do the topic justice here. For a more detailed discussion of the underpinning theory, see the following, particularly chapters 1-3 and 8: M.W. Hirsch, S. Smale, and R.L. Devaney (2004). *Differential Equations, Dynamical Systems and an Introduction to Chaos*. Academic Press/Elsevier, San Diego.

4.1 Fixed Points

Fixed points represent the simplest solutions to differential equations.

Definition 4.1.1 A **fixed point** y^* of the autonomous ODE system $\mathrm{d}y/\mathrm{d}t = f(y)$ is a point for which $f(y^*) = 0$. It follows that the constant-in-time solution $y(t) \equiv y^*$ arises when $y(0) = y^*$. □

Fixed points are also called equilibrium points, equilibria and steady states.

In the scalar case, assume f is a C^1 function and suppose all the fixed points $y_1 < y_2 < \ldots$ of an autonomous scalar equation $\mathrm{d}y/\mathrm{d}t = f(y)$ are known. Then, we can often

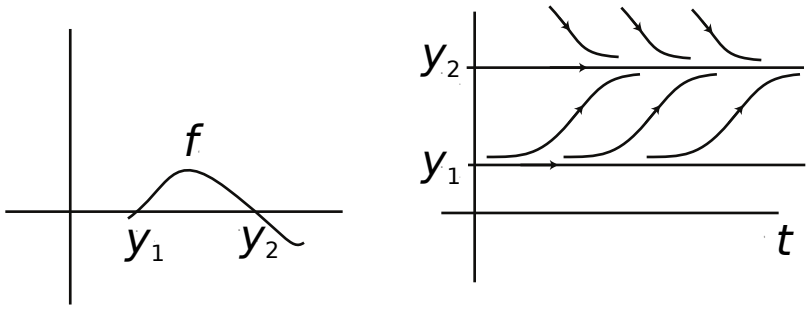

Figure 4.1: The shape of solutions near fixed points can be inferred from knowledge of f.

understand a great deal about the solutions of the system by evaluating the derivative (or derivatives) of f at the fixed points. For example, if we know that $f'(y_1) > 0$, then in a small neighbourhood of y_1, the function f is increasing: for y a little below y_1, we must have

$$\mathrm{d}y/\mathrm{d}t = f(y) < 0$$

and

$$\mathrm{d}^2y/\mathrm{d}t^2 = f'(y)f(y) < 0.$$

This tells us that a little below y_1, y is decreasing and concave down. Similarly, a little above y_1 the graph of y is increasing and concave up. If, moreover, $f'(y_2) < 0$, then, with the assumed smoothness of f, the solutions of the differential equation can be inferred to have the appearance shown on the right of Figure 4.1.

The arrows on the solution curves sketched in Figure 4.1 indicate the direction of time. Notice that curves tend away from y_1 and towards y_2. Loosely speaking, we say that a fixed point y^* is *asymptotically stable* if, for all values u sufficiently close to y^*, the solution through u tends asymptotically to y^* as $t \to \infty$. If instead nearby solutions tend away from the fixed point we say it is unstable. Thus y_1 in Figure 4.1 would correspond to an unstable fixed point and y_2 to an asymptotically stable one. We will define this concept more precisely in the next chapter.

A fixed point of an ODE system corresponds to a fixed point of the flow map, i.e. a point $y^* \in \mathbb{R}^d$ such that

$$\Phi_t(y^*) = y^*, \quad \forall t > 0.$$

We will denote the set of fixed points of Φ_t by $\mathcal{F}$:

$$\mathcal{F} = \{y \in \mathbb{R}^d : f(y) = 0\}.$$

For example, one can check that the Lotka-Volterra model (1.6) has the following fixed points:

$$\mathcal{F} = \{(p = 0, n = 0), (p = 1, n = 1)\}.$$

4.2 Fixed Points of Numerical Methods

A fixed point for a one-step numerical method may be defined analogously:

Definition 4.2.1 For a one-step numerical method described by the map Ψ_h, a **fixed point** y^* is a point for which $\Psi_h(y^*) = y^*$. It follows that the method produces the constant-in-time sequence $y_n \equiv y^*$ when $y_0 = y^*$. □

We will see that for a numerical method, the fixed points may depend on the stepsize h as well as f . We will denote the set of fixed points of Ψ_h by $\mathcal{F}_h$:

$$\mathcal{F}_h = \{y \in \mathbb{R}^d : \Psi_h(y) = y\}.$$

An important practical question is whether the sets $\mathcal{F}$ and $\mathcal{F}_h$ coincide—ideally, the numerical method should reproduce any fixed points possessed by the ODE and should not introduce extra ones.

Consider first Euler's method with numerical flow map given by

$$\Psi_h(y) = y + hf(y).$$

Suppose $y^* \in \mathcal{F}$. Then $f(y^*) = 0$ and therefore $\Psi_h(y^*) = y^*$. Thus, $\mathcal{F} \subseteq \mathcal{F}_h$. Suppose, conversely, that $y^* \in \mathcal{F}_h$. Then, $\Psi_h(y^*) = y^* = y^* + hf(y^*)$, so $hf(y^*) = 0$. It follows that for $h > 0$, $\mathcal{F}_h \subseteq \mathcal{F}$ and therefore, for Euler's method, $\mathcal{F}_h = \mathcal{F}$.

Now consider the implicit midpoint rule (3.19), which we repeat here for convenience:

$$\frac{y_{n+1} - y_n}{h} = f\left(\frac{y_{n+1} + y_n}{2}\right). \tag{4.1}$$

Suppose first that y^* is a fixed point of this numerical method. Then inserting $y_n = y_{n+1} = y^*$ into (4.1), we obtain $f(y^*) = 0$, giving a fixed point of the ODE. So $\mathcal{F}_h \subseteq \mathcal{F}$. Conversely, if y^* is a fixed point of the ODE, then $f(y^*) = 0$. Taking $y_n = y_{n+1} = y^*$ we see that (4.1) is satisfied, and hence y^* is also a fixed point for the numerical method. Hence $\mathcal{F} \subseteq \mathcal{F}_h$. So again $\mathcal{F}_h = \mathcal{F}$ and the sets are equivalent.

Next, consider the explicit Runge-Kutta method with Butcher table

$$\begin{array}{c|cc} 0 & & \\ 1 & & \\ \hline & 0 & 1 \end{array},$$

which we can write simply as

$$y_{n+1} = y_n + hf\left(y_n + hf(y_n)\right).$$

It is clear that if $f(y_n) = 0$, then $y_{n+1} = y_n$, so $\mathcal{F} \subseteq \mathcal{F}_h$ here. The converse is not necessarily true, however. We illustrate this by applying the method to the logistic equation

$$y' = y(1 - y)$$

to obtain the map

$$\Psi_h(y) = y + h\left[y + hy(1-y)\right]\left[1 - y - hy(1-y)\right].$$

For $y \in \mathcal{F}_h$ we have

$$\left[y + hy(1-y)\right]\left[1 - y - hy(1-y)\right] = 0,$$

so one of the terms in square brackets must equal zero. In the first case, we have

$$y(1 + h(1-y)) = 0 \quad \Rightarrow \quad \{y = 0 \text{ or } y = 1 + \frac{1}{h}\}.$$

In the second case,

$$1 - y - hy(1-y) = 0 \quad \Rightarrow \quad \{y = 1 \text{ or } y = \frac{1}{h}\}.$$

Hence, $\mathcal{F}_h = \{0, 1, 1 + h^{-1}, h^{-1}\}$. The fixed points $1 + h^{-1}$ and h^{-1}, which are not fixed points of the underlying ODE, are termed *spurious fixed points*. Note that in this example the spurious fixed points become large as the time step is reduced.

While the set of fixed points of a numerical method $\mathcal{F}_h = \{y \in \mathbb{R}^d : y = \Psi_h(y)\}$ is not generally the same as the set of fixed points $\mathcal{F} = \{y \in \mathbb{R}^d : f(y) = 0\}$ of the flow map, for many classes of methods, the fixed points of the numerical method are a superset of those of the flow map: $\mathcal{F}_h \supseteq \mathcal{F}$. For example, we can easily show this for Runge-Kutta methods:

Theorem 4.2.1 *For Runge-Kutta methods, $\mathcal{F}_h \supseteq \mathcal{F}$.*

Proof Recall the form of a Runge-Kutta method

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i f(Y_i)$$

where

$$Y_i = y_n + h\sum_{j=1}^{s} a_{ij} f(Y_j).$$

Let $y_n = y^*$ be a fixed point of the ODE, so $f(y^*) = 0$. If we set $Y_1 = Y_2 = \ldots = Y_s = y^*$, then we see that all the internal equations are satisfied. Moreover, $y_{n+1} = y_n$, implying that $y_n = y^*$ is a fixed point of the numerical method. □

Thus , for Runge-Kutta methods, $\mathcal{F}_h$ includes all the fixed points of $\mathcal{F}$ but it may have some spurious ones. Notice that the existence of spurious fixed points depends not only on the method but on the differential equation we are solving.

You might think that a Runge-Kutta method which admits spurious fixed points would be of limited interest, since it is then quite possible that a numerical trajectory converges to a point that is nowhere near the stable equlibria of the original dynamical system.

Nonetheless, such methods are used frequently in dynamical systems studies. For example, the 4th order Runge-Kutta method (3.18) is arguably the most popular method of all for dynamical studies, and yet you can check that it admits spurious fixed points on the logistic ODE.

The reason that we can use methods like this is that in general (as in the example above), as $h \to 0$, all the spurious fixed points tend to infinity. This gives us a simple way to test for spurious fixed points: we simply solve the problem repeatedly with different stepsizes. The fixed points which stay put regardless of stepsize are genuine. This is also true of typical methods encountered in practice which are not of Runge-Kutta type. As the analysis quickly becomes rather technical we skip the details here and move on to the issue of stability of fixed points of numerical methods.

4.3 Stability of Fixed Points for ODE Systems

We now consider the stability of a fixed point of an ODE system. In general, stability concerns the behaviour of solutions near a fixed point in the long term. Given an autonomous system of differential equations (1.12), denote the solution of the system with the initial condition $y(0) = y_0$ by $y(t; y_0)$.

Definition 4.3.1 The fixed point y^* is

(1) **stable in the sense of Lyapunov** (or just **stable**) for the given differential equation if $\forall \epsilon > 0$ (sufficiently small), $\exists \delta > 0$, such that, if $\|y_0 - y^*\| < \delta$ then

$$\|y(t; y_0) - y^*\| < \epsilon, \qquad t > 0;$$

(2) **asymptotically stable** if it is stable in the above sense and, additionally, there exists $\gamma > 0$ such that, for any initial condition y_0 such that $\|y_0 - y^*\| < \gamma$,

$$\lim_{t\to\infty} \|y(t; y_0) - y^*\| = 0;$$

(3) **unstable** if it is not stable (in the sense of Lyapunov). □

The simplest way to illustrate the types of situations that can arise is to consider some linear systems:

$$\frac{d}{dt}y = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} y, \qquad \frac{d}{dt}y = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} y, \qquad \frac{d}{dt}y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} y. \tag{4.2}$$

The origin is the only fixed point in each case. It is straightforward to show that the first system is asymptotically stable (and also stable), the second is unstable, and the third is stable but not asymptotically stable. See Figure 4.2.

One sometimes hears it said in engineering or science that a certain *system* is stable. This use of the term may be the same as that used above but suggesting that the fixed point of interest is understood, or it may be a different usage of the term entirely. (Some caution is urged.)

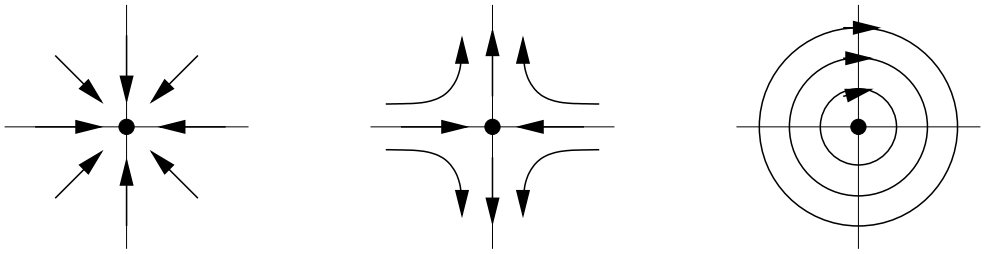

Figure 4.2: Phase diagrams for the three linear systems in $\mathbb{R}^2$ given in (4.2). Here the horizontal and vertical axes correspond to y_1 and y_2, respectively.

There are a number of methods for analyzing stability in general. One of these is the Poincaré-Lyapunov Theorem, which is based on looking at the eigenvalues of the linear part of the problem. We state a simplified version here, without proof, which also is sometimes called the Linearization Theorem.

Theorem 4.3.1 (Linearization Theorem) *Consider the equation in $\mathbb{R}^d$:*

$$\frac{\mathrm{d}y}{\mathrm{d}t} = By + F(y) \tag{4.3}$$

subject to initial condition $y(0) = y_0 \in \mathbb{R}^d$. Assume B is a constant $d \times d$ matrix with eigenvalues having all negative real parts and the function F is C^1 in a neighborhood of $y = 0 \in \mathbb{R}^d$, with $F(0) = 0 \in \mathbb{R}^d$ and $F'(0) = 0 \in \mathbb{R}^{d\times d}$, where $F'(y)$ is the Jacobian matrix of F. Then $y = 0 \in \mathbb{R}^d$ is an asymptotically stable fixed point. If B has any eigenvalues with positive real part, then $y = 0 \in \mathbb{R}^d$ is an unstable fixed point.

Although the conditions on A and F may at first appear restrictive, Theorem 4.3.1 is actually quite powerful. First, if we have a fixed point y^* of a differential equation (1.12) which is not at the origin, we can always shift it to the origin by introducing a translation. Define $\tilde{y} = y - y^*$ and then we have

$$\frac{\mathrm{d}\tilde{y}}{\mathrm{d}t} = \frac{\mathrm{d}y}{\mathrm{d}t} = f(\tilde{y} + y^*) \equiv \tilde{f}(\tilde{y}),$$

which is in the form (1.12) but has a fixed point at $\tilde{y} = 0$.

Second, consider an arbitrary differential equation (1.12) with $f \in C^2$ and $f(0) = 0$. We may proceed as follows to use Theorem 4.3.1. First expand f in a Taylor series expansion about 0:

$$f(y) = f(0) + f'(0)y + R(y).$$

We have $f(0) = 0$. Let $B = f'(0)$ and define $F(y) = R(y)$. We see that we can write our equation in the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = By + F(y),$$

and it is easy to verify that $F(0) = 0$ and $F'(0) = 0$. If the eigenvalues of B have negative real parts, then we can conclude that the origin is asymptotically stable. Let us state an alternative form of the Linearization Theorem based on this:

Theorem 4.3.2 (Linearization Theorem II) *Suppose that f in (1.12) is C^2 and has a fixed point y^*. If the eigenvalues of*

$$J = f'(y^*)$$

all lie strictly in the left complex half-plane, then the fixed point y^* is asymptotically stable. If J has any eigenvalue in the right complex half-plane, then y^* is an unstable point.

4.4 Stability of Fixed Points of Maps

We now discuss discrete-time analogues of the ideas in the previous section. This will lead into definitions for numerical methods.

Definition 4.4.1 Consider a general map Ψ from $\mathbb{R}^d$ to $\mathbb{R}^d$ and a fixed point y^*, so that $y^* = \Psi(y^*)$. Define $y^n(y_0)$ as the iteration of the map n times applied to y_0, so $y^1(y_0) = \Psi(y_0)$, $y^2(y_0) = \Psi(\Psi(y_0))$, etc. We say that
(1) y^* is **stable in the sense of Lyapunov** (or just **stable**) if $\forall \epsilon > 0$ (sufficiently small), $\exists \delta > 0$ such that, if $\|y_0 - y^*\| < \delta$ then

$$\|y^n(y_0) - y^*\| < \epsilon, \qquad n \geq 0.$$

(2) y^* is **asymptotically stable** if it is stable in the above sense and there exists $\gamma > 0$ such that, for any initial condition y_0 such that $\|y_0 - y^*\| < \gamma$,

$$\lim_{n\to\infty} \|y^n(y_0) - y^*\| = 0.$$

(3) y^* is **unstable** if it is not stable (in the sense of Lyapunov). □

It is easy to see that these definitions are analogues of our previous definitions for continuous dynamics: take $\Psi \equiv \Phi_t$, the time-t exact flow map of the differential equation. Then, if a fixed point of the continuous dynamics is stable, it is also a stable fixed point for Ψ.

What are the conditions for stability of a fixed point y^* of a general map? In other words: *when does the sequence of iterates obtained by successively applying Ψ starting from a point near y^* eventually converge to y^*?*

First consider a scalar linear iteration, $\Psi(y) = ay$, $a \in \mathbb{R}$, started from a point $y_0 \neq 0$. The iteration of Ψ yields the sequence $y_n = a^n y_0$. Then we have (i) $|y_n| \to 0$, if $|a| < 1$, (ii) $|y_n| \equiv |y_0|$, if $|a| = 1$, (iii) $|y_n| \to \infty$, $|a| > 1$. The fixed point is at the origin, so we see that we have (i) asymptotic stability, (ii) stability, and (iii) instability.

Now consider a linear iteration in $\mathbb{R}^d$. Then we have $\Psi(y) = Ky$, $K \in \mathbb{R}^{d\times d}$. We consider the convergence to the fixed point at the origin of the sequence of points $y_0, y_1, \ldots$ generated by

$$y_n = Ky_{n-1}.$$

We have $y_1 = Ky_0$, $y_2 = Ky_1 = K^2y_0$, etc. The question of stability concerns the norms of powers of K:

$$\|K^n y_0\|.$$

Let $\rho(K)$ denote the *spectral radius* of the matrix K, i.e. the radius of the smallest circle centered at the origin and enclosing all eigenvalues of K. We have a standard theorem of linear iterations to apply to this case:

Theorem 4.4.1 *Let $z_n = \|K^n y_0\|$.*
Then (i) $z_n \to 0$, as $n \to \infty$, for all y_0, if and only if,

$$\rho(K) < 1.$$

Moreover, (ii) $z_n \to \infty$ for some y_0, if and only if

$$\rho(K) > 1.$$

Finally, (iii) if $\rho(K) = 1$ and eigenvalues on the unit circle are semisimple, then $\{z_n\}$ remains bounded as $n \to \infty$.

When the eigenvectors of K form a basis of $\mathbb{R}^d$, then all parts are easy to prove by diagonalizing the matrix. When K does not have a basis of eigenvectors, we may use the Jordan canonical form. Recall that an eigenvalue is semisimple if the dimension of its largest Jordan block is 1. We will omit the detailed proof, which can be found for example in the book of Hirsch, Smale and Devaney mentioned at the beginning of the chapter.

Finally we turn our attention to a more general iteration of the form

$$y_n = \Psi(y_{n-1}), \tag{4.4}$$

where Ψ is a map on $\mathbb{R}^d$.

Note that

$$y^* = \Psi(y^*). \tag{4.5}$$

Take the difference of (4.4) and (4.5) to obtain

$$y_n - y^* = \Psi(y_{n-1}) - \Psi(y^*).$$

Next, we use Taylor's Theorem for functions from $\mathbb{R}^d$ to $\mathbb{R}^d$ to obtain

$$y_n - y^* = \Psi'(y^*)(y_{n-1} - y^*) + R(y_{n-1} - y^*).$$

$\Psi'(y^*)$ here refers to the Jacobian matrix of Ψ. $R(\Delta)$ is a remainder term which goes to zero quadratically:

$$R(\Delta) = O(\|\Delta\|^2).$$

When $\Delta = y_{n-1} - y^*$ is small in norm, its squared norm is very small. It is natural to think of simply neglecting the remainder term $R(y_{n-1} - y^*)$, i.e.,

$$y_n - y^* \approx \Psi'(y^*)(y_{n-1} - y^*),$$

in which case the convergence would appear to be related to convergence of a linear iteration.

Whether we can neglect the remainder depends on the eigenvalues of Ψ'. It is possible to show the following theorem:

Theorem 4.4.2 *Given a smooth (C^2) map Ψ, (i) the fixed point y^* is asymptotically stable for the iteration $y_{n+1} = \Psi(y_n)$ if*

$$\rho(\Psi'(y^*)) < 1.$$

(ii) the fixed point y^* is unstable if $\rho(\Psi'(y^*)) > 1$.

The marginal case $\rho(\Psi') = 1$ is delicate and must be considered on a case-by-case basis.

4.5 Linear Stability of Numerical Methods

Based on the results quoted in the previous sections in this chapter, we may argue that major insights about stability behaviour of one-step numerical methods can be gleaned by studying the simplified setting where an ODE has been linearized about a fixed point. This type of *linear stability analysis* has proved to be extremely fruitful, and occupies the remainder of this chapter.

Hence, we turn our attention to a general linear ODE system of the form

$$\frac{\mathrm{d}y}{\mathrm{d}t} = By,$$

where B is a $d \times d$ matrix with a basis of eigenvectors, it can easily be shown that the general solution can be written in the compact form

$$y(t) = \sum_{i=1}^{d} C_i e^{\lambda_i t} u_i,$$

where $\lambda_1, \lambda_2, \ldots, \lambda_d$ are the eigenvalues, $u_1, u_2, \ldots, u_d$ are the corresponding eigenvectors, and $C_1, C_2, \ldots, C_d$ are coefficients. It is easy to see that this means that stability is determined by the eigenvalues. For example, if all the eigenvalues lie in the left half plane, then the origin is stable in the sense of Lyapunov. Also, if all the eigenvalues have negative real part, then the origin is asymptotically stable.

A related statement can be shown to hold for many of the numerical methods in common use. For example, consider Euler's method applied to the linear problem $\mathrm{d}y/\mathrm{d}t = By$:

$$y_{n+1} = y_n + hBy_n = (I + hB)y_n.$$

If we let y_n be expanded in the eigenbasis (say $u_1, u_2, \ldots, u_d$), we may write

$$y_n = \alpha_1^{(n)} u_1 + \alpha_2^{(n)} u_2 + \ldots + \alpha_d^{(n)} u_d.$$

If we now apply Euler's method, we find

$$\begin{aligned} y_{n+1} &= (I+hB)(\alpha_1^{(n)}u_1 + \alpha_2^{(n)}u_2 + \ldots + \alpha_d^{(n)}u_d), \\ &= \alpha_1^{(n)}(I+hB)u_1 + \alpha_2^{(n)}(I+hB)u_2 + \ldots + \alpha_d^{(n)}(I+hB)u_d \end{aligned}$$

Now, as u_i is an eigenvector of B, we have

$$(I+hB)u_i = u_i + hBu_i = u_i + h\lambda_i u_i = (1+h\lambda_i)u_i$$

and this implies

$$y_{n+1} = \sum_{i=1}^{d} \alpha_i^{(n)}(1+h\lambda_i)u_i.$$

We may also write

$$y_{n+1} = \sum_{i=1}^{s} \alpha_i^{(n+1)}u_i,$$

and, comparing these last two equations and using the uniqueness of the basis representation, we must have

$$\alpha_i^{(n+1)} = (1+h\lambda_i)\alpha_i^{(n)}.$$

It follows from this that the origin is a stable point if $|1+h\lambda_i| \leq 1$, $i = 1, 2, \ldots, d$, and is asymptotically stable if $|1+h\lambda_i| < 1$, $i = 1, 2, \ldots, d$. The condition for stability can be stated equivalently as requiring that for every eigenvalue λ of B, $h\lambda$ must lie inside a disk of radius 1 centred at $z = -1$ in the complex plane. We call it the **region of absolute stability** of Euler's method.

Thus, given the set of eigenvalues of B, this condition implies a restriction on the maximum stepsize h that can be used if the origin is to remain stable for the numerical map. This is illustrated in Figure 4.3.

4.5.1 Stability functions

To develop a general theory, let us consider first the scalar case. When applied to $y' = \lambda y$, Euler's method has the form

$$y_{n+1} = P(h\lambda)y_n,$$

where $P(\mu) = I + \mu$. Many methods, including all *explicit* Runge-Kutta methods, have the form $y_{n+1} = P(h\lambda)y_n$, for some polynomial $P(x)$, when applied to $y' = \lambda y$. More generally, *all* Runge-Kutta methods, including the implicit ones, when applied to $y' = \lambda y$ can be written in the form

$$y_{n+1} = R(h\lambda)y_n,$$

where $R(\mu)$ is a rational function of μ, i.e. the ratio of two polynomials P and Q,

$$R(\mu) = P(\mu)/Q(\mu).$$

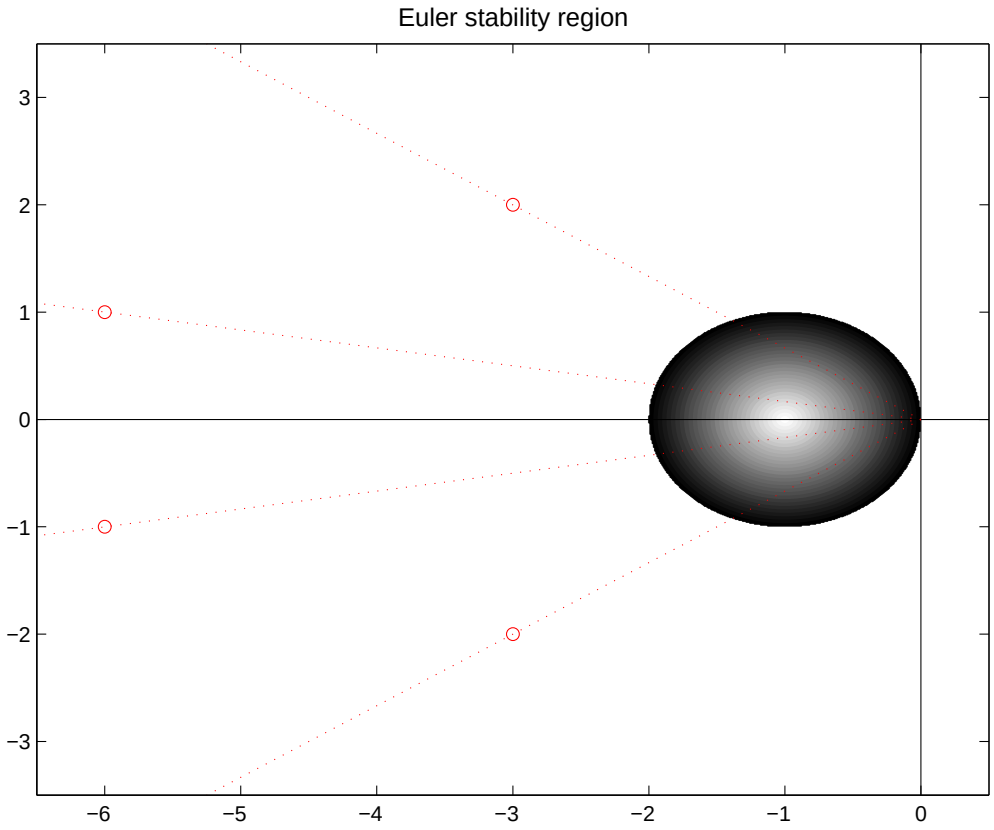

Figure 4.3: The spectrum of B is scaled by h. Asymptotic stability of the origin is recovered if $h\lambda$ is in the region of absolute stability $|1 + z| < 1$ in the complex plane.

R is called the **stability function** of the RK method.

To see this, apply a general Runge-Kutta method to the linear test problem $y' = \lambda y$. We get for the internal stages the relations

$$Y_i = y_n + \mu \sum_{j=1}^{s} a_{ij} Y_j, \quad i = 1, \ldots, s.$$

Introducing the vector $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^s$, the above system can be cast in matrix form (with $Y = (Y_1, \ldots, Y_s)^T$ and A the matrix with entries a_{ij}),

$$Y = y_n \mathbf{1} + \mu A Y,$$

which means $Y = y_n (I - \mu A)^{-1} \mathbf{1}$. Similarly we can write

$$y_{n+1} = y_n + \mu \sum_{j=1}^{s} b_j Y_j = y_n + \mu b^T Y.$$

Combining these expression yields the desired stability function R:

$$y_{n+1} = R(\mu) y_n, \qquad R(\mu) = 1 + \mu b^T (I - \mu A)^{-1} \mathbf{1}. \tag{4.6}$$

That this function is a rational function in μ follows from Kramer's rule for computing the inverse of a matrix.

Next, consider applying the general RK method to $y' = By$:

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i B Y_i,$$

where

$$Y_i = y_n + h \sum_{j=1}^{s} a_{ij} B Y_j.$$

Now expand y_n, y_{n+1} and Y_i, $i = 1, \ldots, s$ in the eigenbasis (a linearly independent set of d eigenvectors) of B. For simplicity, we write the eigenvectors as columns of a matrix U, so that $BU = U\Lambda$, where Λ is the diagonal matrix of eigenvectors. So $U^{-1}BU = \Lambda$ is the diagonalization of B. We define z_n and Z_i, $i = 1, \ldots, s$ by

$$y_n = U z_n, \qquad Y_i = U Z_i, \qquad i = 1, \ldots, s,$$

then we can rewrite our Runge-Kutta method as

$$U z_{n+1} = U z_n + h \sum_{i=1}^{s} b_i B U Z_i,$$

or

$$z_{n+1} = z_n + h \sum_{i=1}^{s} b_i \Lambda Z_i.$$

Now for the internal variables, we have

$$U Z_i = U z_n + h \sum_{j=1}^{s} a_{ij} B U Z_j,$$

or

$$Z_i = z_n + h \sum_{j=1}^{s} a_{ij} \Lambda Z_j.$$

We may observe that, since Λ is a diagonal matrix, the system completely decouples into d independent scalar iterations: this is equivalent to the application of the same Runge-Kutta method to the d scalar (complex) differential equations

$$\frac{\mathrm{d}z^{(i)}}{\mathrm{d}t} = \lambda_i z^{(i)},$$

where $y = Uz$ and $z^T = (z^{(1)}, z^{(2)}, \ldots, z^{(d)})$.

The result of all this is that we can analyze the stability of the origin for a given numerical method by just considering the scalar problem $\mathrm{d}y/\mathrm{d}t = \lambda y$. We state this in a theorem:

Theorem 4.5.1 *Given the differential equation $y' = By$, where we suppose that the $d \times d$ matrix B has a basis of eigenvectors and the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_d$, consider applying a particular Runge-Kutta method. The RK method has a stable (asymptotically stable) fixed point at the origin when applied to*

$$\frac{\mathrm{d}y}{\mathrm{d}t} = By,$$

if and only if the same method has a stable (asymptotically stable) fixed point at the origin when applied to each of the scalar differential equations

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \lambda y,$$

where λ is an eigenvalue of B.

Since we know that a Runge-Kutta method applied to $\mathrm{d}z/\mathrm{d}t = \lambda z$ can be written

$$z_{n+1} = R(h\lambda)z_n$$

we have the following corollary:

Corollary 4.5.1 *Consider a linear differential equation $\mathrm{d}y/\mathrm{d}t = By$ with diagonalizable matrix B. Let an RK method be given with stability function R. The origin is stable for this RK method applied to $\mathrm{d}y/\mathrm{d}t = By$ (at stepsize h) if and only if*

$$|R(\mu)| < 1$$

for all $\mu = h\lambda$, where $\lambda \in \sigma(B)$. (The origin is also unstable if $|R(\mu)| > 1$ for any $\mu = h\lambda$.) Here $\sigma(B)$ denotes the set of eigenvalues of B.

4.5.2 Stability regions and A-stability

The key issue for understanding the long-term dynamics of Runge-Kutta methods near fixed points concerns the region where $\hat{R}(\mu) = |R(\mu)| < 1$. This is called the **stability region** of the numerical method. Let us examine a few stability functions and regions:

Euler's Method:

$$\hat{R}(\mu) = |1 + \mu|.$$

The stability region is the set of points such that $\hat{R}(\mu) < 1$. The condition

$$|1 + \mu| < 1$$

means μ lies inside of a disc of radius 1, centred at the point -1.

Trapezoidal rule: the stability region is the region where:

$$\hat{R}(\mu) = \left|\frac{1 + \mu/2}{1 - \mu/2}\right| < 1.$$

This occurs when

$$|1 + \mu/2| < |1 - \mu/2|,$$

or, when $\mu/2$ is closer to -1 than to 1, which is just the entire left complex half-plane.

Another popular method: *Implicit Euler*,

$$z_{n+1} = z_n + h\lambda z_{n+1},$$

$$\hat{R}(\mu) = |1 - \mu|^{-1},$$

which means the stability region is the *exterior* of the disk of radius 1 centred at 1 in the complex plane. These are some simple examples. All three of these are graphed in Figure 4.4.

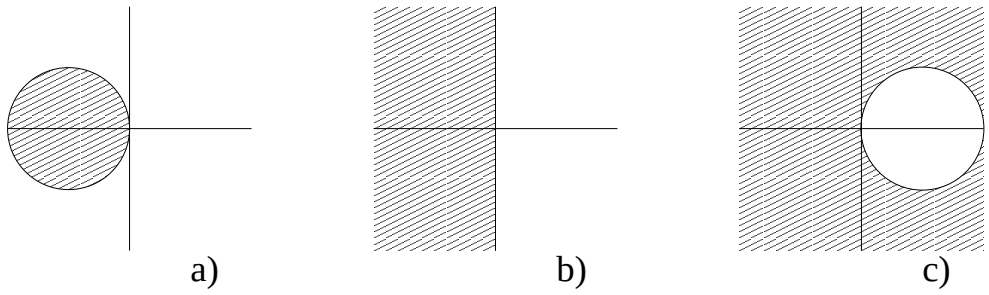

Figure 4.4: Stability Regions: (a) Euler's method, (b) trapezoidal rule, (c) implicit Euler

For the fourth-order Runge-Kutta method (3.18), the stability function is found to be:

$$R(\mu) = 1 + \mu + \frac{1}{2}\mu^2 + \frac{1}{6}\mu^3 + \frac{1}{24}\mu^4.$$

Note that as we would expect, $R(h\lambda)$ agrees with the Taylor Series expansion of $\exp(h\lambda)$ through fourth order. To graph R we could use the following trick. For each value of μ, R is a complex number. The boundary of the stability region is the set of all μ such that $R(\mu)$ is on the unit circle. That means

$$R(\mu) = e^{i\theta},$$

for some $\theta \in [0, 2\pi]$. One way to proceed is to solve the equation $R(\mu) = e^{i\theta}$ for various values of θ and plot the points. There are four roots of this quartic polynomial equation, and in theory we can obtain them in a closed form expression. Unfortunately they are a bit tedious to work out in practice. Certainly we will need some help from Maple, Mathematica, or other computer algebra package.

A second approach is to just plot some level curves of $\hat{R}$ viewed as a function of $\mathrm{Re}(\mu)$ and $\mathrm{Im}(\mu)$, the real and imaginary parts of μ. In fact, the single level curve $\hat{R} = 1$ is precisely the boundary of the stability region! It is useful to have the nearby ones for a range of values near 1.

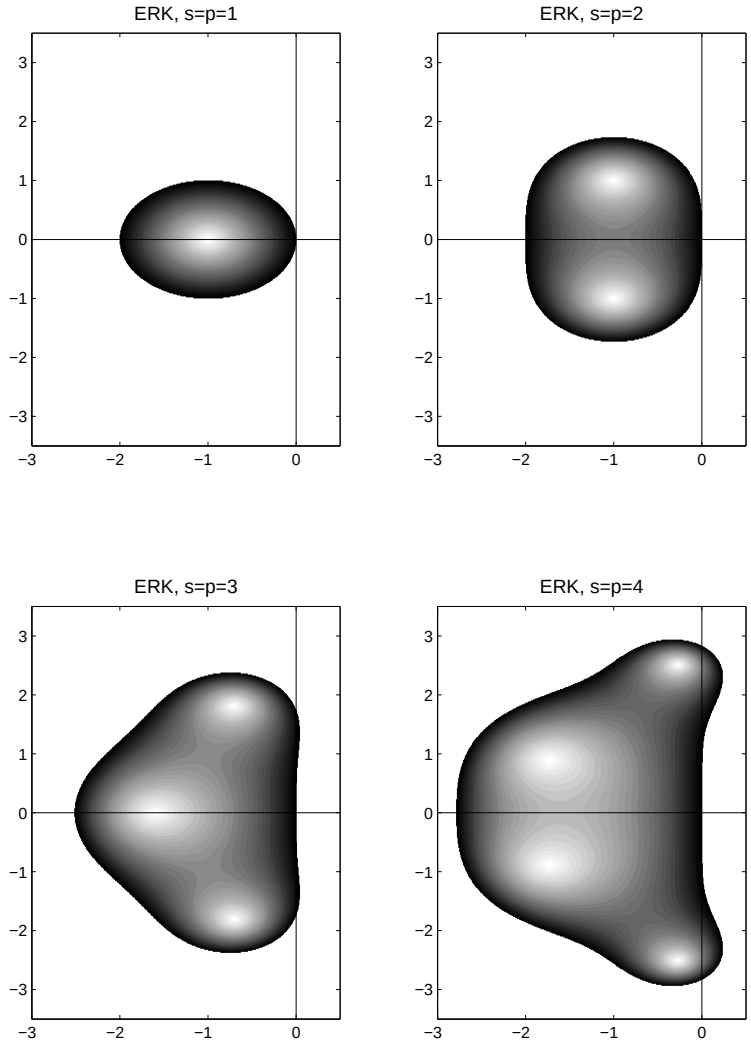

Figure 4.5: Explicit Runge-Kutta Stability Regions

Figure 4.5 shows stability regions for some Runge-Kutta methods up to order 4. The shading in the figure indicates the magnitude $|R(z)|$ within the stability region.

What is the meaning of these diagrams? They tell us a huge amount. Consider first a scalar differential equation $\mathrm{d}y/\mathrm{d}t = \lambda y$, with possibly complex λ. We know that for the differential equation, the origin is stable for λ lying in the left half plane, or, if we think of the map $\Phi_h = e^{h\lambda}$ as defining a discrete dynamics, the origin is stable independent of h if λ is in the left half plane. For a numerical method, the origin is stable if $h\lambda$ lies in the stability region. This is not usually going to happen independently of h, even if λ lies in the left half plane. For the methods seen above, this is only true of trapezoidal rule and implicit Euler. For Euler's method and for the 4th order RK method the origin is only stable for the numerical method provided h is suitably restricted. Note that, in the case of the Euler method, if λ lies strictly in the left half plane, you can see that for h sufficiently

small, we will have $\hat{R}(h\lambda) < 1$, since the rescaling of λ by a smaller value of h moves the point $h\lambda$ towards the origin.

On the other hand, observe that this will be impossible to achieve if λ lies on the imaginary axis. Thus Euler's method is a very poor choice for integrating a problem like $\mathrm{d}z/\mathrm{d}t = \mathbf{i}\,\omega\,z$, where ω is real and $\mathbf{i}$ denotes $\sqrt{-1}$.

Overall, a key message is that a linear ODE system, and its Runge–Kutta discretisation, can be decomposed into scalar linear problems involving the eigenvalues as coefficients. Hence, we can analyze stability of the system by focusing on the scalar version for each eigenvalue.

A very valuable feature when we are interested in preserving the asymptotic stability of the origin under discretization arises if the stability region includes the entire left half plane. In this case we say that the method is *A-stable*. An **A-stable numerical method** has the property that, on linear systems, if the origin is stable for the ODE then it is also stable for the numerical method, regardless of the stepsize.

Chapter 5

Linear Multistep Methods

Main concepts: We have seen Taylor series and Runge-Kutta methods. In this chapter we introduce the other main class of ODE methods: linear multistep methods. The chapter contains an overview of the analytical issues relevant to these methods.

Keywords: Linear multistep methods, difference operators, residual, convergence, root condition, stability.

Assuming the solution has been computed for some number of time steps, the main idea of multistep methods is to use the last k step values to approximate the solution at the next step. As before, we let h denote the stepsize and let $y_n \approx y(t_0 + nh)$ denote the numerical approximation.

For simplicity, in this chapter we will focus on scalar, autonomous ODEs; so $\mathrm{d}y/\mathrm{d}t = f(y)$ with initial condition $y(t_0) = y_0$. (The extension to nonautonomous ODEs and systems is straightforward.)

For notational convenience, we will sometimes use y'_n to represent $f(y_n)$.

Definition 5.0.1 A k-step linear multistep method (LMM) is defined as

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f(y_{n+j}). \tag{5.1}$$

□

This is a formula showing how y_{n+k} is defined, given $y_{n+k-1}, y_{n+k-2}, \ldots, y_n$.

For such a k-step linear multistep method, we require that $\alpha_k \neq 0$ and either $\alpha_0 \neq 0$ or $\beta_0 \neq 0$[1] Furthermore, the coefficients in (5.1) are not uniquely defined, since multiplication through by a constant defines the same method. Usually the coefficients are normalized such that either $\alpha_k = 1$, or $\sum_j \beta_j = 1$.

[1] If $\alpha_k = 0$ then y_{n+k} does not appear in the LHS of (5.1). If $\alpha_0 = \beta_0 = 0$ then we are not making use of y_n and we can take k to be smaller.

When using (5.1) in a computer code, at time step $n+k-1$ it is assumed that the values y_{n+j}, $j = 0, \ldots, k-1$ are already computed, so y_{n+k} is the only unknown in the formula. If β_k is nonzero the method is implicit, otherwise it is explicit. Since the initial value problem (1.3) only specifies $y_0 = y(t_0)$, it is necessary to first generate data y_j, $j = 1, \ldots, k-1$ before the formula (5.1) can be applied. This could be done, for example, by using forward Euler or another one-step method in the first $k-1$ steps.

5.1 Examples

Some examples of linear multistep methods are:

- The θ-method, which generalizes all linear one-step methods,

$$y_{n+1} - y_n = h(1-\theta)f(y_n) + h\theta f(y_{n+1}). \tag{5.2}$$

  Here we have $\alpha_0 = -1$, $\alpha_1 = 1$, $\beta_0 = 1-\theta$ and $\beta_1 = \theta$. Important methods are forward Euler ($\theta = 0$), backward Euler ($\theta = 1$) and trapezoidal rule ($\theta = 1/2$). For any $\theta > 0$, this method is implicit.

- *Leapfrog* is an explicit two-step method ($k = 2$) given by $\alpha_0 = -1$, $\alpha_1 = 0$, $\alpha_2 = 1$ and $\beta_1 = 2$:

$$y_{n+2} - y_n = 2hf(y_{n+1}). \tag{5.3}$$

- The class of *Adams methods* have $\alpha_k = 1$, $\alpha_{k-1} = -1$ and $\alpha_j = 0$ for $j < k-1$. *Adams-Bashforth methods* are explicit, additionally satisfying $\beta_k = 0$. Examples of 1, 2 and 3-step methods are (using notation $f_n \equiv f(y_n)$):

$$y_{n+1} - y_n = hf_n \tag{5.4}$$

$$y_{n+2} - y_{n+1} = h\left(\frac{3}{2}f_{n+1} - \frac{1}{2}f_n\right) \tag{5.5}$$

$$y_{n+3} - y_{n+2} = h\left(\frac{23}{12}f_{n+2} - \frac{4}{3}f_{n+1} + \frac{5}{12}f_n\right). \tag{5.6}$$

  Adams-Moulton methods are implicit, with $\beta_k \neq 0$.

- The *Backward differentiation formulae (BDF)* are a class of linear multistep methods satisfying $\beta_j = 0$, $j < k$ and generalizing backward Euler. The two-step method (BDF-2) is

$$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = h\frac{2}{3}f(y_{n+2}). \tag{5.7}$$

5.2 Derivation of multistep methods

The simplest multistep methods can be constructed by "guessing". For example the leapfrog method for $y' = f(t, y)$ takes the form

$$y_{n+2} - y_n = 2hf(t_{n+1}, y_{n+1}) \equiv 2hf_{n+1}.$$

This can be seen as a natural centred difference approximation for the derivative. However, to get general families of useful multistep methods we need to employ a more systematic procedure.

In many cases, methods are constructed by finding interpolating polynomials (through y-values or through y' values), and then differentiating or integrating them to define multistep methods.

For example the Adams-Bashforth schemes are constructed as follows. Consider the k points

$$(t_n, y'_n), (t_{n+1}, y'_{n+1}), \ldots, (t_{n+k-1}, y'_{n+k-1}).$$

Now define a polynomial $\Pi_k^f(t)$ which has degree $k-1$ and passes through these points. Then replace $f(t, y(t))$ in the right hand side of the differential equation $y'(t) = f(t, y(t))$ by the interpolant and integrate both sides from t_{n+k-1} to t_{n+k}; that is, use

$$y_{n+k} - y_{n+k-1} = \int_{t_{n+k-1}}^{t_{n+k}} \Pi_k^f(t)\mathrm{d}t$$

Example: For $k = 1$ we get $\Pi_k^f(t) \equiv y'_n$ and $y_{n+1} = y_n + hy'_n$ which is forward Euler. For $k = 2$, using a linear polynomial through (t_n, y'_n) and (t_{n+1}, y'_{n+1}) we get

$$y_{n+2} - y_{n+1} = h\left[(3/2)y'_{n+1} - (1/2)y'_n\right].$$

This is referred to as the 2-step Adams-Bashforth method, or AB(2). Obviously you could extend this to define AB(3), AB(4), etc.

We can get other families by a similar technique. For example Adams-Moulton methods are constructed by defining the interpolating polynomial $\hat{\Pi}_k^f(t)$ which passes through

$$(t_n, y'_n), (t_{n+1}, y'_{n+1}), \ldots, (t_{n+k-1}, y'_{n+k-1}), (t_{n+k}, y'_{n+k}).$$

It has a higher degree than the corresponding one Π_k^f since it involves an extra point. Note that y_{n+k} is not yet available, but we may carry on and generate a formula that involves y_{n+k} implicitly. We write

$$y_{n+k} - y_{n+k-1} = \int_{t_{n+k-1}}^{t_{n+k}} \hat{\Pi}_k^f(t)\mathrm{d}t.$$

As just the simplest example, we can work out that $\hat{\Pi}_1^f(t) = y'_n + (t - t_n)(y'_{n+1} - y'_n)/h$ and thus we get the method

$$y_{n+1} = y_n + (h/2)[y'_n + y'_{n+1}].$$

This is the 1-step Adams-Moulton method which is just the trapezoidal rule. In a similar way we can derive higher order Adams-Moulton methods, for example the 2-step scheme is

$$y_{n+2} - y_{n+1} = h[(5/12)y'_{n+2} + (2/3)y'_{n+1} - (1/12)y'_n].$$

5.2.1 Algebra of Operators

In coming up with these methods it is very useful to know about the algebra of operators. An operator is like a function of a generalized type, typically a function that has functions as inputs and outputs. We will treat these without the formal presentation that would be needed for a full analysis.

Our main aims are (a) to see some elegant mathematics, and (b) to define the BDF class.

We will consider operators on functions $g(t)$ of one real argument. The simplest operator is 1. It is the operator that leaves the argument alone, i.e.

$$1g(t) = g(t).$$

For an operator L we write L^2 to indicate two applications of the operator. Thus $1^2g(t) = 1(1g(t)) = 1g(t) = g(t)$. OK that was easy. Now consider another operator E_s which is defined by

$$E_s g(t) = g(t + s);$$

it is called the shift operator. If h is a step in t then what we work with in numerical ODEs is the approximation of a differential equation at points

$$\ldots t_n - 2h, t_n - h, t_n, t_n + h, t_n + 2h, \ldots$$

so it is natural to work with E_h which just moves us up and down the sequence $E_h^2 g(t) = g(t + 2h)$, etc., so $E_h^2 g(t_n) = g(t_{n+2})$ and if we want we can write $E_h^2 y_n = y_{n+2}$ etc.

Now one way to build approximations is using the difference operators.

The forward difference operator is Δ_h is defined by

$$\Delta_h g(t) = g(t + h) - g(t).$$

We recognize $g(t + h)$ as $E_h g(t)$ so we could write this relation as

$$\Delta_h g(t) = E_h g(t) - g(t) = (E_h - 1)g(t).$$

This is useful because it represents a relation of the form $\Delta_h = E_h - 1$, which is an algebraic relation.

Now comes a very intriguing observation. Define D as the operator that differentiates a function, then we have $Dg(t) = g'(t)$. Now observe that

$$g(t + h) = g(t) + hg'(t) + \frac{h^2}{2}g''(t) + \ldots = g(t) + hDg(t) + \frac{h^2 D^2}{2}g(t) + \ldots = e^{hD}g(t).$$

That is a very compact representation of the Taylor Series expansion. But it also gives us a useful relation:

$$E_h = e^{hD}.$$

Since h can be anything, we can write

$$g(t) = e^{(t-t_n)D} g(t_n);$$

this is a relation between g at any t and g at some specific point.

Now notice that

$$\Delta_h g(t) = g(t+h) - g(t) = (E_h - 1)g(t) = (e^{hD} - 1)g(t).$$

It means that

$$e^{hD} = 1 + \Delta_h$$

so

$$D = \frac{1}{h} \ln[1 + \Delta_h].$$

Thus we have

$$g(t) = e^{(t-t_n)D} g(t_n) = e^{[(t-t_n)/h] \ln(1+\Delta_h)} g(t_n) = (1+\Delta_h)^{(t-t_n)/h} g(t_n).$$

We can expand in the binomial series:

$$(1+x)^\alpha = 1 + \alpha x + \frac{\alpha(\alpha-1)}{2} x^2 + \frac{\alpha(\alpha-1)(\alpha-2)}{3!} x^3 + \dots$$

so we get

$$g(t) = \left[1 + \frac{t-t_n}{h} \Delta_h + \frac{(t-t_n)(t-t_n-h)}{2h^2} \Delta_h^2 + \dots \right] g(t_n).$$

If we truncate, this defines an approximation to $g(t)$ in terms of forward differences. This truncation actually gives an alternative formula for the interpolating polynomial.

5.2.2 BDF formula

Now come back to the algebra of operators and notice that

$$E_h^{-1} g(t_n) = g(t_{n-1})$$

but we can also use a Taylor expansion to write

$$g(t_{n-1}) = e^{-hD} g(t_n).$$

Thus

$$e^{-hD} = E_h^{-1}.$$

So that means

$$D = -\frac{1}{h}\ln(E_h^{-1}).$$

We can write this as

$$D = -\frac{1}{h}\ln(1-(1-E_h^{-1})). \tag{5.8}$$

The operator $1-E_h^{-1}$, for which $(1-E_h^{-1})g(t_n) = g(t_n)-g(t_n-h)$, is called the **Backward Difference Operator** and we write it $\nabla_h g(t_n)$. Expanding the log expression in (5.8), we have

$$D = \frac{1}{h}[\nabla_h + \frac{1}{2}\nabla_h^2 + \frac{1}{3}\nabla_h^3 + \ldots].$$

The main idea behind the BDF methods is to use a truncation of this series to replace the derivative on the left hand side of the ODE, i.e. replace

$$Dy(t) = y'(t) = f(t, y(t))$$

by

$$\frac{1}{h}[\nabla_h + \frac{1}{2}\nabla_h^2 + \frac{1}{3}\nabla_h^3 + \ldots]y(t) = f(t, y(t)).$$

If we keep just the first few terms this gives approximate relations like

$$\frac{1}{h}\nabla_h y \approx f(t, y(t)) \Rightarrow y(t) - y(t-h) \approx hf(t, y(t))$$

which suggests the Backward Euler method

$$y_n - y_{n-1} = hf(t_n, y_n).$$

[Note that we could shift the indices $n \to n+1$ to get this into the standard form $y_{n+1}-y_n = hy'_{n+1}$ for a one-step method.]

Keeping two terms in the series gives

$$\frac{1}{h}[\nabla_h + \frac{1}{2}\nabla_h^2]y(t) = f(t, y(t))$$

which suggests

$$y_n - y_{n-1} + \frac{1}{2}[y_n - 2y_{n-1} + y_{n-2}] = hf(t_n, y_n)$$

or

$$(3/2)y_n - 2y_{n-1} + (1/2)y_{n-2} = hy'_n$$

or, after a shift of indices and a rescaling,

$$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = h\frac{2}{3}y'_{n+2}$$

This is called the BDF(2) method.

5.3 Order of accuracy

Associated with the linear multistep method (5.1) are the polynomials

$$\rho(\zeta) = \sum_{j=0}^{k} \alpha_j \zeta^j, \qquad \sigma(\zeta) = \sum_{j=0}^{k} \beta_j \zeta^j. \tag{5.9}$$

We refer to these as the **first characteristic polynomial** and **second characteristic polynomial**, respectively.

We will see that these polynomials arise naturally when we analyse convergence and stability.

5.3.1 Linear case

One way to understand where the characteristic polynomials come from is to consider applying the LMM to the scalar, linear, test equation, $y' = \lambda y$. In this case we get

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j y'_{n+j} = h \sum_{j=0}^{k} \beta_j \lambda y_{n+j}.$$

We can view this as a recurrence relation

$$\sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j) y_{n+j} = 0.$$

For a linear recurrence relation like this it is natural to guess a solution of the form $y_n = y_0 \zeta^n$, with $y_0 \neq 0$, $\zeta \neq 0$, and if we introduce this here we find

$$y_0 \sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j) \zeta^{n+j} = 0.$$

Dividing out by $y_0 \zeta^n$ results in

$$\sum_{j=0}^{k} (\alpha_j - h\lambda\beta_j) \zeta^j = 0$$

or

$$\rho(\zeta) - h\lambda\sigma(\zeta) = 0.$$

This is an equation for ζ and clearly there are k roots giving rise to k solutions. If the roots are are distinct, we have the "general solution"

$$y_n = C_1 \zeta_1^n + C_2 \zeta_2^n + \ldots + C_k \zeta_k^n.$$

5.3.2 Residual

The **residual** of a linear multistep method at time t_{n+k} may be defined in a number of ways. Our definition will be given by substituting the exact solution $y(t)$ of (1.3) at times $y(t_{n+j})$, $j = 0, \ldots, k$ into (5.1), i.e.,

$$r_n := \sum_{j=0}^{k} \alpha_j y(t_{n+j}) - h \sum_{j=0}^{k} \beta_j y'(t_{n+j}). \tag{5.10}$$

(This is actually the residual accumulated in the $(n + k - 1)$th step, but for notational convenience we will denote it r_n.) A linear multistep method has **order of consistency** p if $r_n = O(h^{p+1})$ for all sufficiently smooth f.

To study the order of consistency, we write the Taylor series expansions of $y(t_{n+j})$ and $y'(t_{n+j})$ as

$$y(t_{n+j}) = \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i)}(t_n), \qquad y'(t_{n+j}) = \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i+1)}(t_n),$$

where in this chapter $y^{(i)}(t)$ means the ith derivative of $y(t)$. Substituting these into (5.10) and manipulating,

$$\begin{aligned}
r_n &= \sum_{j=0}^{k} \alpha_j \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i)}(t_n) - h \sum_{j=0}^{k} \beta_j \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i+1)}(t_n) \\
&= \sum_{j=0}^{k} \alpha_j y(t_n) + \sum_{i=1}^{\infty} \sum_{j=0}^{k} \alpha_j \frac{(jh)^i}{i!} y^{(i)}(t_n) - \sum_{i=1}^{\infty} \sum_{j=0}^{k} \beta_j \frac{(jh)^i}{j(i-1)!} y^{(i)}(t_n) \\
&= \sum_{j=0}^{k} \alpha_j y(t_n) + \sum_{i=1}^{\infty} \frac{1}{i!} h^i y^{(i)}(t_n) \left[\sum_{j=0}^{k} \alpha_j j^i - i \sum_{j=0}^{k} \beta_j j^{i-1} \right].
\end{aligned}$$

It follows that conditions for a linear multistep method to have order of consistency $p \geq 1$ are that the coefficients α_j and β_j satisfy (where $0^0 = 1$)

$$\sum_{j=0}^{k} \alpha_j = 0 \qquad \text{and} \qquad \sum_{j=0}^{k} \alpha_j j^i = i \sum_{j=0}^{k} \beta_j j^{i-1} \quad \text{for} \quad i = 1, \ldots, p. \tag{5.11}$$

It may be shown that equivalent conditions for a linear multistep method to have order of consistency $p \geq 1$ are:

- The first and second characteristic polynomials $\rho(\zeta)$ and $\sigma(\zeta)$ satisfy

$$\rho(e^z) - z\sigma(e^z) = O(z^{p+1}). \tag{5.12}$$

- The first and second characteristic polynomials $\rho(\zeta)$ and $\sigma(\zeta)$ satisfy

$$\frac{\rho(z)}{\log z} - \sigma(z) = O((z-1)^p). \tag{5.13}$$

For justifications of (5.12) and (5.13), see Hairer, Nørsett and Wanner (1993).
Examples. The method (5.3) has order 2. The methods (5.5) and (5.6) have orders 2 and 3, respectively. The method (5.7) has order 2.

5.4 Convergence of Multistep Methods

In previous chapters, we saw that for Euler's method, convergence follows from the fact that the residual is $O(h^2)$, leading to a global error of $O(h)$ on a fixed interval. More generally, for one-step methods, if the local error is $O(h^{p+1})$ we say the method is pth order consistent, and then under a stability condition, the method will also converge at order p. (Consistency + Stability implies Convergence: see Theorem 5.4.1 below, and Section 5.5.)

For linear multistep methods, consistency is insufficient to ensure convergence. We do not provide a detailed convergence theory in these notes, but we will describe the basic ideas that guarantee convergence.

5.4.1 The Root Condition and Convergence

The simplest differential equation we can think of is $y' = 0$ which has a constant solution. If we apply a Runge-Kutta method to such a differential equation, we obtain $y_{n+1} = y_n$ which is exactly what we expect. However, if we apply a multistep method, we get

$$\sum_j \alpha_j y_{n+j} = 0.$$

This is a recurrence relation with the solution

$$y_n = \sum_{i=1}^{k} C_i \zeta_i^n,$$

where $\zeta_1, \zeta_2, \ldots, \zeta_k$ are the k roots of the first characteristic polynomial ρ. When we apply the multistep method, we don't know the C_i in advance, we effectively determine them from a starting procedure. (A linear multistep method is incomplete without a specified starting procedure to generate the first $k-1$ iterates $y_1, \ldots, y_{k-1}$.)

It is interesting to note that one of the roots of ρ is always one. This comes from the fact that for a first order consistent (residual at least $O(h^2)$) method, we must have $\sum_{j=0}^{k} \alpha_j = 0$. Thus $\rho(1) = 0$. If we let $\zeta_1 = 1$, then the other roots that define the solution of numerical method are somehow extraneous but they cannot be ignored! One thing we

would not like is for any of the roots of ρ to be outside the unit disk since then we would have an exponentially growing extraneous solution component. It can even happen that roots are on the boundary of the unit disk but if one of these has multiplicity > 1 then we again can have a growing solution component. It is important to rule this out. Hence, we make the following definition.

Definition 5.4.1 The LMM (5.1) is said to satisfy the **root condition**, if all roots ζ of

$$\rho(\zeta) = 0,$$

lie in the unit disk ($|\zeta| \leq 1$), and any root of modulus one ($|\zeta| = 1$) has multiplicity one. □

5.4.2 Convergence Theorem

We are now ready to state the classic convergence theorem for multistep methods. The following result applies when the ODE system is sufficiently well-behaved (e.g., f has continuous and bounded partial derivatives).

Theorem 5.4.1 *Suppose a linear multistep method (5.1) is equipped with a starting procedure satisfying* $\lim_{h\to 0} y_j = y(t_0 + jh)$ *for* $j = 1, \ldots, k-1$. *Then the method is guaranteed to converge to the exact solution of (1.3) on a fixed interval as* $h \to 0$ *if and only if it has order of consistency* $p \geq 1$ *and satisfies the root condition.*

We can summarize this as

LMM convergent $\Leftrightarrow$ LMM consistent with $p \geq 1$ and satisfies root condition

or, more simply, as

convergence = consistency + stability.

This powerful result allows us to rule out useless (non-convergent) LMMs by checking algebraic conditions involving the coefficients.

Furthermore, for sufficiently smooth ODEs, if the starting procedure is pth order accurate then it may be shown that the method will converge with order p, i.e., for sufficiently small h the global error will satisfy

$$\max_{0\leq n\leq N} |y_n - y(t_n)| \leq Ch^p$$

where $Nh = T - t_0$.

The proof of these results will not be handled in these notes. See, e.g., the book of Hairer, Nørsett, & Wanner (1993).

To illustrate the necessity of the root condition, consider the method

$$y_{n+3} + y_{n+2} - y_{n+1} - y_n = h\left(\frac{8}{3}f(y_{n+2}) + \frac{2}{3}f(y_{n+1}) + \frac{2}{3}f(y_n)\right). \tag{5.14}$$

Here $k = 3$, $\alpha_3 = 1$, $\alpha_2 = 1$, $\alpha_1 = -1$, $\alpha_0 = -1$, $\beta_3 = 0$, $\beta_2 = 8/3$, $\beta_1 = 2/3$ and $\beta_0 = 2/3$. We will first check the consistency conditions (5.11) (noting that these conditions use the convention $0^0 = 1$). For $p = 1$ we have

$$\sum_{j=0}^{k} \alpha_j = 1 + 1 - 1 - 1 = 0, \quad \sum_{j=0}^{k} \alpha_j j^1 = 3 + 2 - 1 = 4, \quad 1 \sum_{j=0}^{k} \beta_j j^0 = 8/3 + 2/3 + 2/3 = 4.$$

For $p = 2$ we have

$$\sum_{j=0}^{k} \alpha_j j^2 = 9 + 4 - 1 = 12, \quad 2 \sum_{j=0}^{k} \beta_j j^1 = 2((8/3) \times 2 + 2/3 + 0) = 12.$$

For $p = 3$ we have

$$\sum_{j=0}^{k} \alpha_j j^3 = 27 + 8 - 1 = 34, \quad 3 \sum_{j=0}^{k} \beta_j j^2 = 3((8/3) \times 4 + 2/3 + 0) = 34.$$

It is also straightforward to show that the $p = 4$ condition does not hold. Hence this LMM has order of consistency $p = 3$.

Now applying the method to the easiest of all initial value problems,

$$y' = 0, \quad y(0) = 1, \quad t \geq 0,$$

yields the linear difference equation

$$y_{n+3} + y_{n+2} - y_{n+1} - y_n = 0. \tag{5.15}$$

If the roots ζ_1, ζ_2 and ζ_3 of the characteristic polynomial $\rho(\zeta) = 0$ were distinct, the exact solution of such a recursion would be

$$y_n = c_1 \zeta_1^n + c_2 \zeta_2^n + c_3 \zeta_3^n.$$

If any root ζ_i had modulus greater than 1, then the recursion would satisfy $|y_n| \to \infty$ unless the corresponding constant c_i were identically 0. For the current case, $\rho(\zeta) = (\zeta - 1)(\zeta + 1)^2$, and there is a double root at -1. For a double root $\zeta_3 \equiv \zeta_2$, the solution of the recursion (5.15) becomes

$$y_n = c_1 \zeta_1^n + c_2 \zeta_2^n + c_3 n \zeta_2^n.$$

(This is easy to check.) One still has $|y_n| \to \infty$ as $n \to \infty$ (but with linear growth), unless $c_3 = 0$.

The constants c_1, c_2 and c_3 are determined by the initial values necessary to start the multistep method. Suppose we take $y_0 = y_1 = y_2 = 1$, consistent with the exact solution. Then this yields

$$c_1 = 1, \quad c_2 = c_3 = 0,$$

and the solution is $y_n = 1$, for all n. The method is exact.

Suppose, however, that the initial values are perturbed slightly. We take $y_0 = 1 + \varepsilon$, $y_1 = y_2 = 1$. Then we find

$$c_1 = 1 + \frac{1}{4}\varepsilon, \quad c_2 = \frac{3}{4}\varepsilon, \quad c_3 = -\frac{1}{2}\varepsilon.$$

So, for any given $\varepsilon \neq 0$ the solution is unbounded. The numerical solution sequence is unstable to perturbations in the starting values. As a consequence, any errors incurred will destabilize the solution.

The root condition in Definition 5.4.1, which by Theorem 5.4.1 is required for convergence, imposes a constraint on the maximum attainable order of a LMM:

Theorem 5.4.2 *The maximum order of a k-step method satisfying the root condition is $p = k$ for explicit methods and, for implicit methods, $p = k + 1$ for odd k and $p = k + 2$ for even k.*

For a proof of this result, see the book of Hairer, Nørsett, & Wanner (1993).

5.5 Stability

As we have seen earlier in this course, an important criterion for distinguishing between different methods is their ability to preserve the stability of an asymptotically stable equilibrium. To test this in the case of a multistep method, we check under what conditions the numerical solution converges to zero when we apply (5.1) to the scalar linear test problem $y' = \lambda y$:

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h\lambda \sum_{j=0}^{k} \beta_j y_{n+j}.$$

(We keep in mind that λ represents an eigenvalue in the case where we have linearized around a fixed point and then diagonalized, as in Chapter 4.) Letting $z = h\lambda$ we then have

$$\sum_{j=0}^{k} (\alpha_j - z\beta_j) y_{n+j} = 0.$$

For any z this is a linear difference equation with characteristic polynomial

$$\sum_{j=0}^{k} (\alpha_j - z\beta_j)\zeta^j = 0 = \rho(\zeta) - z\sigma(\zeta).$$

The (absolute) **stability region** $\mathcal{S}$ of a linear multistep method is the set of all points $z \in \mathbb{C}$ such that all roots ζ of the polynomial equation $\rho(\zeta) - z\sigma(\zeta) = 0$ satisfy $|\zeta| < 1$. Because λ represents an eigenvalue of an ODE Jacobian, we are interested to know when

$z = h\lambda$ lies in the stability region for z with negative real part (i.e., we are interested to know what stepsizes, h, produce a stable numerical method in the case where the ODE is stable).

On the boundary of the stability region $\mathcal{S}$, there is a root of the polynomial equation $\rho(\zeta) - z\sigma(\zeta) = 0$ with modulus one, say $\zeta = e^{\mathrm{i}\theta}$. Therefore an explicit representation for the boundary of $\mathcal{S}$ is easily derived:

$$\partial\mathcal{S} = \left\{ z = \frac{\rho(e^{\mathrm{i}\theta})}{\sigma(e^{\mathrm{i}\theta})},\ \theta \in [-\pi, \pi] \right\}.$$

Figure 5.1 shows plots of the stability regions for the Adams-Bashforth methods of orders $p = 1$, 2 and 3.

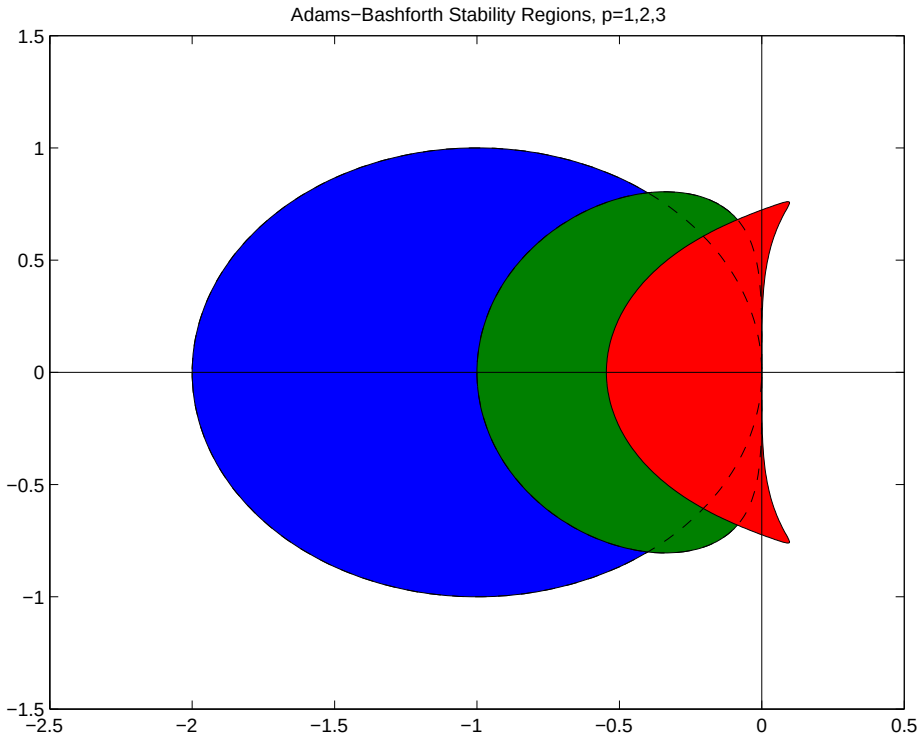

Figure 5.1: Stability regions of the Adams-Bashforth methods of orders $p = 1$ (blue), 2 (green) and 3 (red).

A linear multistep method is called **A-stable** (or **unconditionally stable**) if the stability domain $\mathcal{S}$ contains the entire left half-plane, that is, if $z \in \mathbb{C}$ with $\mathrm{Re}\, z < 0$ implies that z is in the stability region of the LMM. This is a desirable property: it implies that the LMM will be stable, **for any choice of stepsize**, whenever the underlying ODE is stable. However, A-stability imposes a major constraint:

Theorem 5.5.1 *An A-stable linear multistep method has order $p \leq 2$.*

This restriction on the maximum order of a linear multistep method was an important result in numerical analysis, proved by G. Dahlquist.

To get a less stringent condition, we say that the linear multistep method (5.1) is A(α)-stable, for some $\alpha \in (0, \pi/2)$, if the stability region contains an α-wedge in the left half-plane:

$$\{z \in \mathbb{C} : |\arg(z) - \pi| < \alpha\} \subset \mathcal{S}.$$

In this case, for any λ lying within the α-wedge of stability, the method is unconditionally stable (norm-nonincreasing for any h).

Figure 5.2 shows plots of the stability regions for the Backward differentiation formulae (BDF) of orders $p = 1, \ldots, 6$. (These stability regions are unbounded—but we can only plot a finite portion of the complex plane.) The BDF methods have excellent stability properties, and hence are widely used. The first and second order BDF methods are A-stable. The rest are A(α)-stable with the following α:

Order, p	α
3	86.03°
4	73.35°
5	51.84°
6	17.84°

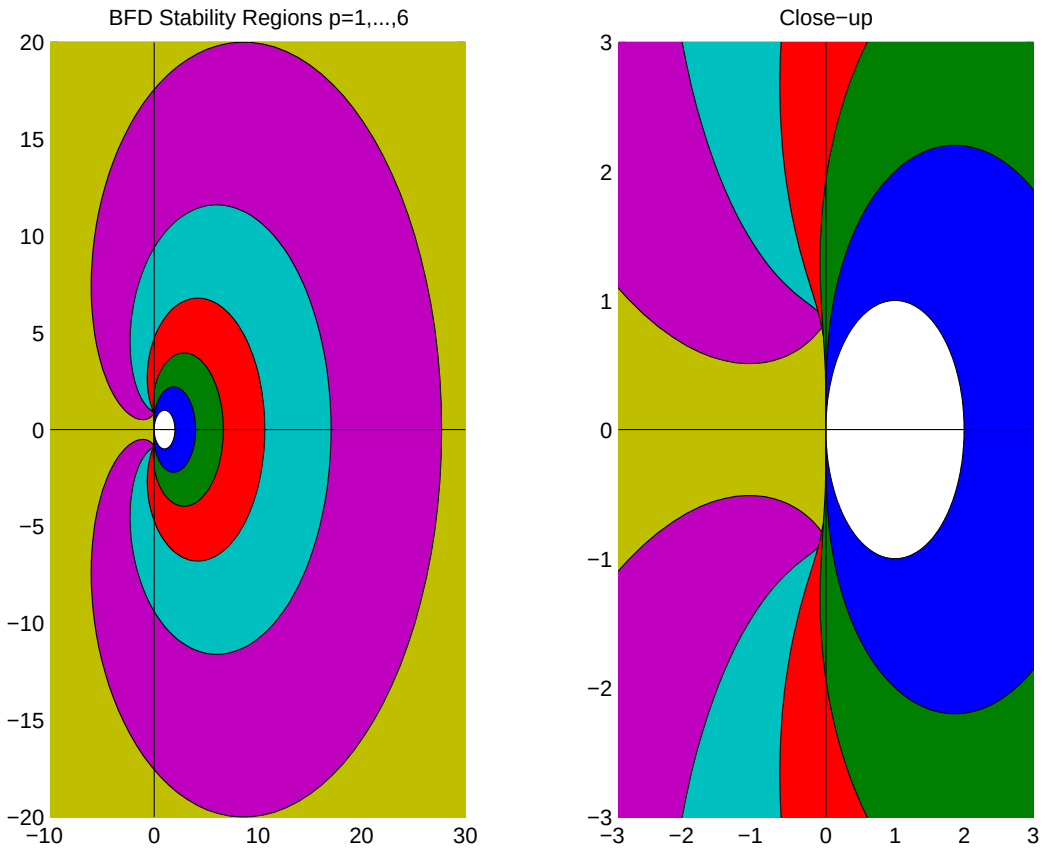

Figure 5.2: Stability regions of the BDF methods of orders $p = 1, \ldots, 6$. For BDF-1 (blue) $\mathcal{S}$ is everything outside the white region; for BDF-2 (green) it is everything outside the blue region; etc.