# Computational Modelling and Dynamical Systems

Jason Frank (CWI, Amsterdam)  Ben Leimkuhler (U. of Edinburgh)

September 11, 2012

# Contents


**1 Modelling** **1**
- 1.1 Modelling with dynamical systems . . . 1
- 1.2 References . . . 8
- 1.3 Exercises . . . 8

**2 Simulation** **9**
- 2.1 Computational modelling . . . 9
- 2.2 Trapezoidal rule . . . 10
- 2.3 Matlab Code . . . 12

**3 Solutions** **13**
- 3.1 Illustrative examples . . . 13
- 3.2 Norms, Gronwall lemmas, Lipschitz continuity . . . 15
- 3.3 Convergence of Euler's method, existence and uniqueness . . . 18
- 3.4 References . . . 21
- 3.5 Exercises . . . 21

**4 The Matrix Exponential** **23**
- 4.1 The general scalar, linear equation in the complex plane. . . . 23
- 4.2 Systems of linear ODEs . . . 24
  - 4.2.1 Diagonalizable systems . . . 24
  - 4.2.2 The matrix exponential . . . 25
  - 4.2.3 Variation of constants formula . . . 25
- 4.3 Computing the exponential of a matrix . . . 26

**5 Flow Maps and Dynamical Systems** **27**
- 5.1 Classical mechanics . . . 28
- 5.2 Phase Space . . . 30
- 5.3 The Group of Flow Maps . . . 30
- 5.4 Dynamical systems . . . 32
- 5.5 Discretization and Approximate Flow Maps . . . 33
- 5.6 References . . . 35
- 5.7 Exercises . . . 35

**6 Convergence of One-Step Methods** **37**
6.1 Approximation . . . 37
6.2 Convergence of generalized one-step methods . . . 38
6.3 Exercises . . . 40

**7 Collocation Methods** **41**
7.1 Construction of one-step methods . . . 41
7.2 Polynomial interpolation . . . 42
7.3 Numerical quadrature . . . 42
7.4 One-step collocation methods . . . 43
7.5 Exercises . . . 45

**8 Runge-Kutta Methods** **47**
8.1 The Family of Runge-Kutta Methods . . . 47
8.1.1 Some Runge-Kutta methods . . . 48
8.1.2 Local accuracy of Runge-Kutta methods . . . 50
8.1.3 Order conditions for Runge-Kutta methods . . . 52
8.2 Exercises . . . 55

**9 Equilibrium Points and Fixed Points** **57**
9.1 Equilibrium Points and Stability for Scalar ODEs . . . 57
9.2 Fixed Points of Numerical Methods . . . 58

**10 Stability of Runge-Kutta Methods** **63**
10.1 Stability of Equilibrium Points . . . 63
10.2 Stability of Fixed Points of Maps . . . 65
10.3 Stability of Numerical Methods: Linear Case . . . 67
10.3.1 Stability functions . . . 68
10.3.2 Stability regions, A-stability and L-stability . . . 71
10.4 Oscillatory Systems . . . 74
10.5 Exercises . . . 76

**11 Interlude: Linear Multistep Methods** **79**
11.1 Examples . . . 79
11.2 Order of accuracy and convergence . . . 80
11.2.1 The root condition, a counter-example . . . 81
11.3 Stability . . . 84
11.4 Implementation issues . . . 85
11.4.1 Starting process. . . . 85
11.4.2 Variable stepsize implementation. . . . 86
11.4.3 Error estimation. . . . 87
11.4.4 Nonlinear systems. . . . 87
11.5 Exercises . . . 88

**12 Reversibility and Composition Methods** **89**
12.1 $\rho$-Reversibility . . . . . 89
12.2 Adjoint of a Method . . . . . 91
12.3 Composition Methods . . . . . 92
12.4 Exercises . . . . . 93

**13 Splitting Methods** **95**
13.1 Introduction to Splitting Methods . . . . . 95
13.1.1 Linear ODEs . . . . . 96
13.1.2 Nonlinear ODEs . . . . . 98
13.2 Exercises . . . . . 101

**14 First Integral** **105**
14.1 First Integrals . . . . . 105
14.2 First Integrals Under Discretization . . . . . 106
14.2.1 Conservation of quadratic first integrals by Runge-Kutta methods . 109
14.3 Discrete Gradient Methods . . . . . 111
14.4 Splitting Methods . . . . . 112
14.5 Exercises . . . . . 113

**15 Constraints and Projection** **115**
15.1 Weak Invariants . . . . . 115
15.2 Projection . . . . . 115
15.3 Numerical Methods with Projection . . . . . 117
15.4 Preserving Weak Invariants Under Discretization . . . . . 118

**16 Symplectic Flows and Maps and Volume Preservation** **121**
16.1 Hamiltonian Systems . . . . . 121
16.2 Divergence-Free Vector Fields . . . . . 123
16.2.1 Volume-preserving flows and Liouville's theorem . . . . . 124
16.2.2 Volume-preserving numerical methods . . . . . 125
16.3 Symplectic Structure . . . . . 125
16.4 Symplectic Integrators . . . . . 127
16.4.1 Symplectic splitting methods . . . . . 127
16.5 Exercises . . . . . 128

**17 Backward Error Analysis** **129**
17.1 A Modified Differential Equation . . . . . 129
17.1.1 A modified equation . . . . . 129
17.1.2 More modified equations . . . . . 131
17.2 Good News and Bad News . . . . . 135

**18 Dissipative Problems** **139**

**19 Gradient flows** **141**
19.1 Lyapunov Functions . . . . . 141

**A Implementation Details** **143**
A.1 Solving Nonlinear Systems . . . . . 143
A.2 Error Control . . . . . 144
A.2.1 Error control . . . . . 144

# Chapter 1

# Modelling



**Main concepts:** In this first chapter we introduce some basic definitions and a series of model problems of increasing complexity. Keywords are: ODE, IVP, solutions

## 1.1 Modelling with dynamical systems

Suppose one day you, as a working mathematician, are asked to model the infestation of a rose farm by aphids, a tiny plant-feeding insect (Fig. 1.1a), and assess the feasibility of controlling the aphid population by introduction of a natural predator.

Being a mathematician, you decide that you need a mathematical model to describe the dynamics of the aphid population. Assuming you know nothing at all about population dynamics and can't be bothered to go to the library or search the internet, you might start reasoning as follows: Let $N(t)$ denote the number of aphids in the rose farm at time $t$. Although we would generally think of $N$ as taking discrete (integer) values, we make the modelling assumption that it can be viewed as a continuous variable, taking values on the (positive) real line $N \in \mathbb{R}$. This assumption is reasonable when the number of aphids is so large that addition or subtraction of a fractional aphid won't matter. If the average



Figure 1.1: *Prey and predator.* a. aphids devouring a rosebud. b. ladybug devouring an aphid. (source: Wikipedia)

life span of an aphid is, say, ten days, then on the average one-tenth of the population will die on any given day. Therefore the death rate is linearly proportional to size of the population: *death rate* $= mN(t)$, $m > 0$. The birth rate works the same way. Although we do not want to delve into the rather exotic reproductive activities of the aphid, suffice it to say that most aphids are females, able to reproduce asexually, giving birth instead of laying eggs, and born pregnant. The birth rate is then some average number $b$ of young per aphid per day: *birth rate* $= bN(t)$, $b > 0$. Based purely on considerations of birth rate and death rate, you could construct a model for the aphid population by setting the rate of change in population equal to the difference in birth and death rates, $dN/dt = bN(t) - mN(t)$. This results in the *exponential growth/decay model*

$$\frac{dN(t)}{dt} = rN(t), \tag{1.1}$$

where $r \in \mathbb{R}$ is the rate of population increase or decrease, depending on whether there are more births or deaths. For aphids, all else being equal, the births far outweigh the deaths.

The model (1.1) is an example of a differential equation, and it is to this type of model—and in particular to its solution on the computer—that this course is devoted.

Formally, a **differential equation** of the general form

$$\frac{dy}{dt} = f(t, y) \tag{1.2}$$

specifies the relation between an unknown differentiable function $y(t)$ and its derivative $dy/dt$. Equation (1.2) is termed an **ordinary differential equation** (ODE) because it involves derivatives with respect to a single independent variable ($t$). 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 the independent variable, which is suggested by the letter $t$. We will often use the shorthand notation $y'(t)$ and occasionally $\dot{y}$ to denote $dy/dt$.

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

$$\frac{\partial \tilde{y}(t)}{\partial t} = f(t, \tilde{y}(t)),$$

at least for all $t$ on some interval of interest. For example, for the aphid population model (1.1), the function

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

is a solution for each constant $c \in \mathbb{R}$, since $d\tilde{N}/dt = cr\exp(rt) = rN(t)$ conform (1.1). We will usually drop the tilde when there is no chance of confusion.

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

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

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

The model (1.1) might be useful to describe the extinction of aphids when treated by a pesticide that reduces the birth rate, or to describe the initial phase of 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. The rose farm can only sustain a certain number of aphids, say $N^*$. A more sophisticated attempt is the model

$$\frac{dN}{dt} = rN\left(1 - \frac{N}{N^*}\right).$$

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*

$$\frac{dn}{dt} = rn(1-n). \tag{1.4}$$

We can generally solve any of a wide variety of differential equations of the form

$$\frac{dx}{dt} = f(x), \qquad x, f : \mathbb{R} \to \mathbb{R},$$

as follows:

$$\frac{dx}{f(x)} = dt,$$

then

$$\int \frac{dx}{f(x)} = t + C,$$

where $C$ is an arbitrary constant. In practice there are some difficulties that can obstruct this procedure. For example it may be difficult to evaluate the indicated integral. Even when an antiderivative $G(x)$ for $1/f(x)$ can be written down, we must solve in some way the (typically nonlinear) equation

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

However, these problems can often be overcome and the solution computed as accurately as required at any point.

Let us solve the logistic model (1.4) using a decomposition into partial fractions:

$$\begin{aligned} r\,dt &= \frac{dn}{n(1-n)} \\ rt + C &= \int \frac{dn}{n(1-n)} = \int \frac{1}{n} + \frac{1}{1-n}\,dn \\ &= \ln n - \ln(1-n). \end{aligned}$$

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 1.2.

Next, suppose that in attempt to control the aphid population, a predator is introduced into the rose farm. One nasty predator is the ladybug (Fig. 1.1b. If you are an aphid, the ladybug is anything but a symbol of nonviolence!). Normally lady bugs will not bother eating aphids unless there is sufficient supply. The logistic model can be extended to incorporate *predation* by lady bugs as follows:

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

The last term gradually "turns on" ladybug 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 for which ladybugs get "interested". Some solutions of this model are shown in the right plot of Figure 1.2. 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 *bifurcation*.

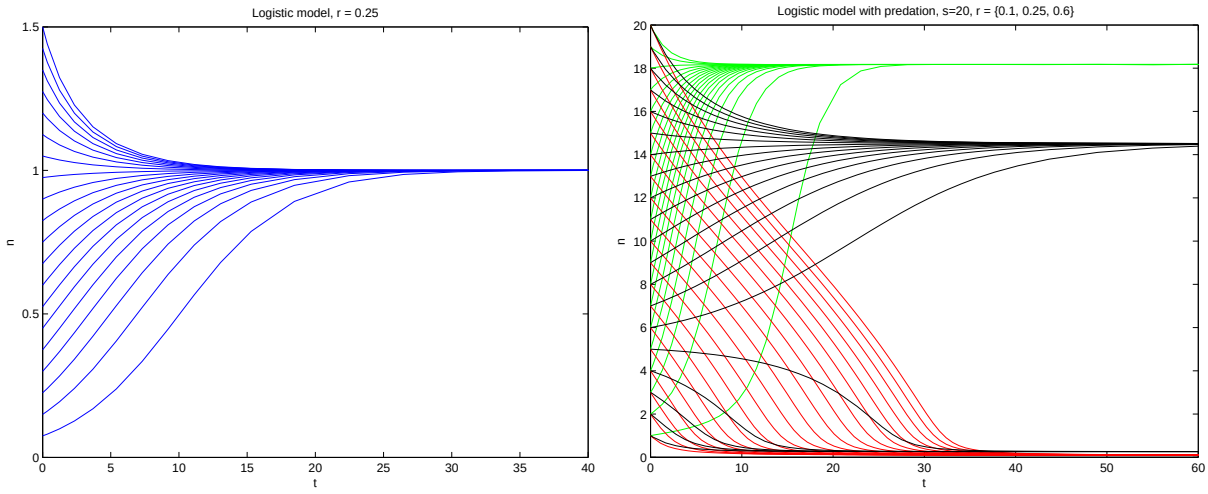



Figure 1.2: Solutions of the logistic model. On the left, solutions of (1.4) with $r = 0.25$ for different initial conditions. On the right, with predation (1.5), $s = 20$ and $r = 0.6$ (green), $r = 0.25$ (black), and $r = 0.1$ (red). For $r = 0.25$, there are two steady states, depending on the initial condition

This model is still rather crude, because it assumes an endless supply of ladybugs. An alternative model might consider an isolated system of aphids and ladybugs, and model the populations of both. Let $p(t)$ denote the (normalized) population of ladybugs. In the absence of aphids, the ladybugs would die out exponentially: $dp/dt = -mp$ for some $m > 0$. But with an increasing number of aphids, the ladybugs will flourish, so the ladybug population should satisfy

$$\frac{dp}{dt} = anp - mp$$

for positive constants $a$ and $m$. Similarly, in the absence of ladybugs, the aphid population would grow exponentially (at least until they saturate the environment). And their death

rate should increase proportional to the number of predators, so

$$\frac{dn}{dt} = bn - cnp,$$

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*

$$\frac{dn}{dt} = rn(1-p) \tag{1.6a}$$

$$\frac{dp}{dt} = p(n-1), \tag{1.6b}$$

where $r > 0$.

The Lotka-Volterra model is an example of a set of two coupled ordinary differential equations. More generally one may have a **system of $d$ differential equations**:

$$\frac{dy^{(1)}}{dt} = f^{(1)}\left(t, y^{(1)}, y^{(2)}, \ldots, y^{(d)}\right) \tag{1.7}$$

$$\frac{dy^{(2)}}{dt} = f^{(2)}\left(t, y^{(1)}, y^{(2)}, \ldots, y^{(d)}\right) \tag{1.8}$$

$$\vdots \tag{1.9}$$

$$\frac{dy^{(d)}}{dt} = f^{(d)}\left(t, y^{(1)}, y^{(2)}, \ldots, y^{(d)}\right) \tag{1.10}$$

$$\tag{1.11}$$

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. 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.

The reproduction rate of aphids is actually not constant, as suggested by the constant parameter $r$ in the above models. In fact 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

$$\frac{dn}{dt} = r(t)\, n \left(1 - \frac{n}{s}\right) - \frac{n^2}{1+n^2}, \tag{1.12}$$

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**

$$\frac{dy}{dt} = f(y), \qquad y, f \in \mathbb{R}^d. \tag{1.13}$$

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 $dt/d\tau = 1$, and

$$\frac{dy}{d\tau} = \frac{dy}{dt}\frac{dt}{d\tau} = \frac{dy}{dt}.$$

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

For notational convenience, it is sometimes useful to write a *nonautonomous* ODE (i.e. one with explicit dependence on $t$) as an autonomous one. This is always possible 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

$$\begin{aligned}
\frac{dy^{(1)}}{ds} &= f^{(1)}\left(y^{(d+1)}, y^{(1)}, y^{(2)}, \ldots, y^{(d)}\right) \\
&\vdots \\
\frac{dy^{(d)}}{ds} &= f^{(d)}\left(y^{(d+1)}, y^{(1)}, y^{(2)}, \ldots, y^{(d)}\right) \\
\frac{dy^{(d+1)}}{ds} &= 1
\end{aligned}$$

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

From a modelling point of view, nonautonomous terms (such as $r(t)$ in the example above) can often be avoided by extending the system considered. For example instead of defining an explicit function $r(t)$, we could couple our predation model to a climate prediction model in which the average temperature $r$ appears as dependent variable. Typically such a climate model will have additional variables as well, and the coupled system may

look like

$$\begin{aligned}\frac{dx}{dt} &= f(x,r)\\ \frac{dr}{dt} &= g(x,r)\\ \frac{dn}{dt} &= rn\left(1-\frac{n}{s}\right)-\frac{n^2}{1+n^2},\end{aligned}$$

where $r$ and $n$ are scalars and $x \in \mathbb{R}^d$.[1] On the other hand, an accurate climate model may have $d = 10^9$, so this approach could constitute overkill.

## 1.2 References

For a thorough introduction to mathematical modelling and population dynamics in ecological systems, see Chapters 1–3 of

> James D. Murray, *Mathematical Biology I: An Introduction*, Third edition, Interdisciplinary Applied Mathematics series, Springer, 2002.

## 1.3 Exercises

1. Consider (1.12) and suppose the aphid reproduction rate is given by

$$r(t) = r_0(\cos \varepsilon t + c).$$

Write a Matlab function to implement the differential equation (see the example function `lv.m` for the Lotka-Volterra model). Solve this with the Matlab toolbox function `ode23` for $r_0 = 0.32$, $c_0 = 1$, $\varepsilon = 2\pi/40$, $s = 20$. On the same axes, plot the solutions on the interval $t \in [0, 160]$ for initial aphid populations of $n = 1$, $n = 2$, $n = 3$ and $n = 10$. Describe what you see. What can you conclude about the numerical solution? about the model? about the phenomenon being modelled? (Use the Matlab help utility to learn about `ode23`. You can use the default tolerances.)

[1] The influence of our aphid population on the climate has been neglected, as indicated by the absence of $n$ dependence in $f$ and $g$.

# Chapter 2

# Simulation

**Main concepts:** In this first chapter we introduce numerical integrators and time stepping. Keywords are: Euler's method, trapezoidal rule, Matlab code

## 2.1 Computational modelling

A solution to a differential equation may exhibit very complex behavior, as we will see later. The complexity increases with the dimension $d$, and problems becomes more and more difficult to treat by purely analytical means. Instead, in this course we will discuss the approximation of solutions of differential equations on a computer.

The most fundamental property of the numerical approximation is that it must be computable in a finite number of operations (unless you are willing to wait an eternity for the solution). The process of replacing the continuous solution by a finite one is called *discretization*.

To do so, we replace the interval $D = [t_0, t_0 + T]$ by a finite number $N$ of discrete times $t_n = t_0 + nh$, $n = 0, 1, \ldots, N$, where $h = T/N$ is the step size. Similarly, we replace the continuous solution $y(t)$ on $D$ with the numerical solution $y_n \approx y(t_n)$, $n = 0, 1, \ldots, N$. The $y_n$ can be thought of as snapshots of the system state at the discrete times, and the sequence $\{y_0, y_1, \ldots, y_N\}$ as a movie. Finally, we need a procedure to generate the $y_n$ for $n > 0$ ($y_0$ is the given initial condition). In this course, we will consider methods that approximate $y_{n+1}$ recursively given $y_n$, $h$, and the explicit form of $f(y)$. Such methods are termed *one-step methods* to distinguish them from *multistep methods* which use information from several preceding time steps. A time-stepping procedure is referred to as a *numerical integrator*or *scheme*.

There are several approaches to the construction of integrators. The simplest and oldest method is based on the rectangle rule for approximation of an integral. Consider the special case of (1.2) with $f(t, y) = f(t)$ ($f$ is independent of $y$). The rectangle rule is just $y_{n+1} = y_n + hf(t_n)$. Generalizing this to $y$-dependent problems is straightforward, and

results in **Euler's method**

$$y_{n+1} = y_n + hf(t_n, y_n). \tag{2.1}$$

Euler's method can be interpreted in several other ways: (1) as the direct extrapolation of the local slope through $(t_n, y_n)$, (2) as the single term truncation of a Taylor expansion, or (3) as a finite difference. For the last case, consider the definition of the derivative:

$$\frac{dy}{dt} = \lim_{h \to 0} \frac{y(t+h) - y(t)}{h} = f(t, y),$$

and to approximate, simply do not take the limit. Different interpretations can be used to generalize Euler's method and construct other methods. We will see some of these in later chapters.

To program this method, we write a loop and compute the successive iterates as follows:

**Algorithm 2.1.1 (Euler's Method) Given:** *initial time $t_0$, initial value $y_0$, stepsize $h$, a vector field $f$, and a number of time steps $N$,*
**Output:** *$y_1, y_2, \ldots, y_N$ approximating the solution of $\frac{dy}{dt} = f(y,t)$ at equally spaced points $t_1 := t_0 + h, t_2 = t_0 + 2h, \ldots$.*
**for** $n = 0, 1, \ldots, N-1$
  $t_{n+1} := t_0 + nh$;
  $y_{n+1} := y_n + hf(y_n, t_n)$;
**end** □

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 2.1. This figure is troublesome because it appears to show a trajectory which spirals out, which seems to suggest a problem with the model (if this trend continues, both populations may grow unbounded, which is not the kind of behavior we would expect from an isolated ecological system.) The cause of this behavior becomes more clear if we consider the effect of decreasing the stepsize, as shown in the right panel of Figure 2.1, 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 solution 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 completely wrong 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.

## 2.2 Trapezoidal rule

Before concluding this chapter with some example codes in Matlab, we introduce a second numerical method. From elementary calculus, you may remember that a better approxima-

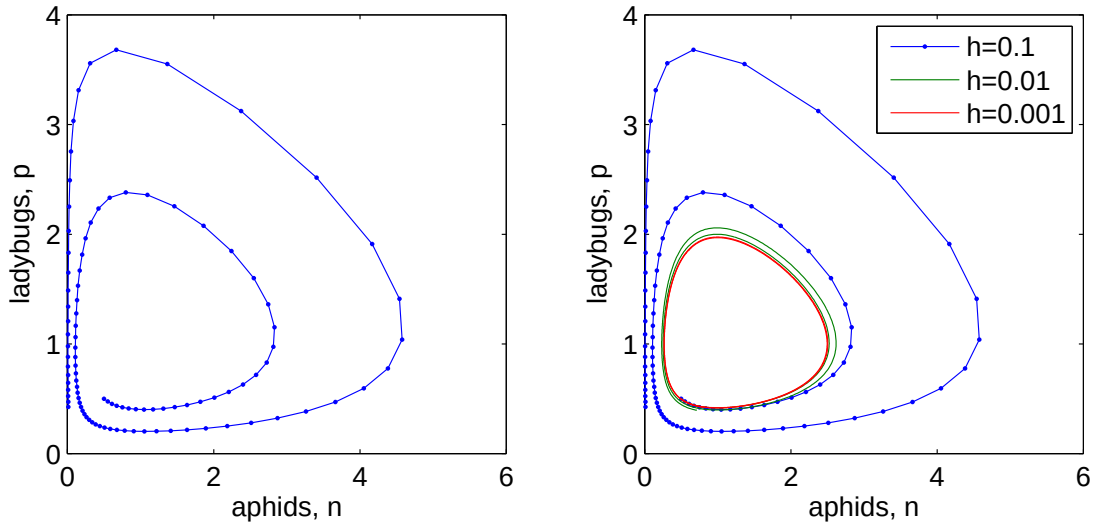



Figure 2.1: The locus curve of predators and prey for the Lotka-Volterra model, left with $h = 0.1$ and right, with $h = 0.1$ (100 steps), $h = 0.01$ (1000 steps) and $h = 0.001$ (10000 steps).

tion than the rectangle rule is the trapezoidal rule. Again consider the equation $\frac{dy}{dt} = f(t)$. The trapezoidal rule approximation is $y_{n+1} = y_n + \frac{h}{2}\left(f(t_{n+1}) + f(t_n)\right)$. Generalizing to the ODE (1.2), the **trapezoidal rule** is

$$y_{n+1} = y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right). \tag{2.2}$$

Notice that this method is of a character very different from Euler's method. It is not possible to evaluate the last term in the equation without knowing $y_{n+1}$, and it is not possible to compute $y_{n+1}$ without evaluating this term. The trapezoidal rule defines $y_{n+1}$ *implicitly* as function of $y_n$. In other words, we must solve the typically nonlinear system of algebraic equations

$$0 = r(y) := y - y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, y)\right) \tag{2.3}$$

for $y$ to determine $y_{n+1}$. Such methods are termed *implicit methods*, and generally demand much more work from the computer per time step than an *explicit method* such as Euler's method. Methods for solving systems of algebraic equations are covered in an appendix to these notes. Obviously an implicit method must be significantly advantageous in some other sense to justify its increased computational cost. We will say more about this in later chapters. Also, it is uncertain if (2.3) will have any real solutions, and if so, how many. For $h = 0$, there is the unique solution $y = y_n$, and for $h$ small enough, the implicit function theorem assures us there is a unique solution. For larger $h$, uniqueness and existence are not guaranteed.

## 2.3 Matlab Code

The following m-file computes the value of the vector field of the Lotka-Volterra model (1.6) at a given point.
lv.m

```
function yprime = lv(t,y)
% lotka volterra vector field
r=2;  %coefficient
yprime =[0;0];   %creates a column vector with two elements
yprime(1) =  r*y(1)*(1  -  y(2));
yprime(2) =  y(2)*(y(1) - 1);
```

The following m-file implements Euler's Method.
euler.m

```
function [ylist,tlist] = euler(y0, t0, m, vf, h, N)
% solves dy/dt = f(y,t) in R^m starting from y(t0)=y0, using N steps of size h
% the initial condition should be an m-dimensional column vector
% vf(y,t) should evaluate the vector field f at the point (y,t)
% results are returned in tlist and ylist, the latter in the form of a matrix, each
% of whose columns represent the numerical solution at a timestep

tlist = zeros(1,N+1);
ylist = zeros(m, N+1);
tlist(1) = t0;
ylist(:,1) = y0;

for nn=1:N,
tlist(nn+1) = t0 + nn*h;
ylist(:,nn+1) = ylist(:,nn) + h*feval(vf, tlist(:,nn), ylist(:,nn));
end
```

A typical run would be performed and plotted as follows.

```
>> t0 = 0;  y0=[0.5; 0.5];
>> [ylist,tlist] = euler(y0,t0,2,'lv',0.1,100);
>> plot(tlist,ylist,'o-');
```

# Chapter 3

# Solutions

**Main concepts:** In this chapter we discuss the well-posedness of initial value problems and the convergence of Euler's method. Keywords are: norms, Lipschitz continuity, convergence of Euler's method, existence and uniqueness

To illustrate the subject of this chapter, we consider two scalar initial value problems for which the solution can be obtained explicitly. These are purely concocted problems, not related to any specific model.

## 3.1 Illustrative examples

The first example looks innocent enough:

$$y' = -2\sqrt{1-y}, \quad y(0) = 1. \tag{3.1}$$

You can check that the solution

$$y(t) = 1 - t^2,$$

is valid for all $t$. However, if you try the constant solution

$$y(t) \equiv 1,$$

you will see that this function also satisfies the differential equation and initial condition, and is also valid for all $t$. This shows that the initial value problem (3.1) has no unique solution. If we attempt to integrate (3.1) using Euler's method with step size $h$, we obtain the iterates

$$\begin{aligned} y_1 &= y_0 + h \cdot (-2)\sqrt{1-y_0} = 1 \\ y_2 &= y_1 + h \cdot (-2)\sqrt{1-y_1} = 1 \\ &\vdots \end{aligned}$$

The trapezoidal rule, on the other hand, yields for the first time step

$$y_1 = y_0 + \frac{h}{2}(-2)\left(\sqrt{1-y_0} + \sqrt{1-y_1}\right) = 1 - h\sqrt{1-y_1}.$$

One can check that the solutions to this nonlinear equation are $y_1 = 1$ and $y_1 = 1 - h^2$. This means that the trapezoidal rule will give a solution without error, but the solution chosen will depend on how one solves the nonlinear equation.

Our second example is the initial value problem

$$y' = y^2, \quad y(0) = y_0. \tag{3.2}$$

This differential equation has the trivial solution $y \equiv 0$ for $y_0 = 0$. For $y_0 \neq 0$, you can directly solve this equation to obtain

$$y(t) = \frac{1}{\frac{1}{y_0} - t}.$$

If $y_0 < 0$, the solution exists for all (positive) time. However, for any $y_0 > 0$, we have

$$\lim_{t \to 1/y_0} y(t) = \infty,$$

so the solution ceases to exist at time $1/y_0$. What happens if you integrate (3.2) with Euler's method? The sequence

$$y_{n+1} = y_n + hy_n^2$$

is well defined for any $y_n \in \mathbb{R}$. Therefore the numerical solution will exist for all $t$ independent of $y_0$! For trapezoidal rule one obtains the quadratic equation

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

You can check that this equation has real roots so long as

$$hy_n < \sqrt{2} - 1 \approx 0.4.$$

For $y_0$ negative the numerical solution always exists. For $y_0$ positive, the numerical solution grows in magnitude until this condition is no longer satisfied. This is reminiscent of the behavior of the exact solution. (Another possibility is to continually reduce the stepsize $h$ such that the condition above is satisfied for each step. One will then find that there is some time $t^* > 0$ which the numerical solution will never reach.)

These examples illustrate the issues of *existence* and *uniqueness* of solutions of differential equations, and of numerical solutions. A related issue is the *convergence* of the numerical solution to that of the differential equation, when a unique solution to the latter exists. We will tackle all of these questions at once, using the following plan:

1. First we will show that, under appropriate assumptions on the function $f$, the numerical solution of Euler's method for the initial value problem (1.3) converges to a continuous function $Y(t)$ on $[t_0, t_0 + T]$ in the limit $h \to 0$.

2. Next we will show that $Y(t)$ is a solution to the initial value problem (1.3) on $[t_0, t_0 + T]$.

3. Finally we will show that $Y(t)$ is the unique solution of (1.3).

This proof is based on the original existence and uniqueness proof of Cauchy, who also used Euler's method. It is somehow satisfying to appreciate that this fundamental result in analysis is grounded in a numerical method.

We begin the analysis with some background concepts.

## 3.2 Norms, Gronwall lemmas, Lipschitz continuity

Recall that a **norm** on a vector space $V \subseteq \mathbb{C}^d$ is a function $\|\cdot\| : V \to \mathbb{R}$ that satisfies, for all $y, x \in V$, $a \in \mathbb{C}$,

1. $\|y\| \geq 0$, with equality holding only if $y = 0$,

2. $\|ay\| = |a|\|y\|$,

3. $\|y + x\| \leq \|y\| + \|x\|$.

The most common examples are the 2-norm $\|y\|_2 = (y^* y)^{1/2}$ and the maximum norm $\|y\|_\infty = \max_{i=1,\ldots,d} |y^{(i)}|$.

On a finite dimensional vector space all norms are equivalent in the sense that if $\|\cdot\|$ and $|||\cdot|||$ are any two norms, there exist positive constants $c$ and $C$ such that for all $y \in V$,

$$c\|y\| \leq |||y||| \leq C\|y\|.$$

For a given vector norm $\|\cdot\|$ there is an associated matrix norm on $\mathbb{C}^{d\times d}$ defined by

$$\|A\| := \max_{\|y\|=1} \|Ay\|.$$

Matrix norms on $\mathbb{C}^{d\times d}$ are also all equivalent.

Consider an open, convex set $D \subseteq \mathbb{R} \times \mathbb{R}^d$. A vector field $f(t, y) : D \to \mathbb{R}^d$ is said to satisfy a **Lipschitz condition on** $D$ if there exists a constant $L > 0$ such that

$$\|f(t, y) - f(t, x)\| \leq L\|y - x\|, \qquad \forall (t, x), (t, y) \in D$$

We also say that $f$ is *Lipschitz continuous* on $D$, and call $L$ the *Lipschitz constant*.

Lipschitz continuity implies uniform continuity, but is a weaker property than continuous differentiability. Therefore, if $f$ is continuously differentiable, it is also Lipschitz

continuous. Intuitively, the Lipschitz constant $L$ is a bound on how much the function $f$ can change along a path from $x$ to $y$. In the scalar ($d = 1$), autonomous case one has

$$|f(y) - f(x)| \leq L|y - x| \quad \iff \quad \left|\frac{f(y) - f(x)}{y - x}\right| < L$$

from which it is obvious that if $f(y)$ satisfies a Lipschitz condition then $L$ places an upper bound on the derivative of $f(y)$. If $f(t, y)$ is differentiable on a compact set $D \subset \mathbb{R} \times \mathbb{R}^d$, then we can take $L$ to be the maximum value obtained by the norm of the Jacobian of $f$ on $D$:

$$L = \max_{(t,y)\in D} \|\frac{\partial f}{\partial y}(t, y)\| < \infty. \tag{3.3}$$

(For $d = 1$, the proof of this statement is by contradiction, using the mean value theorem.)

Next, consider the following

**Lemma 3.2.1 (Gronwall Lemma)** *Let the following differential inequality hold for the scalar function $z(t) : \mathbb{R} \to \mathbb{R}$:*

$$\frac{dz}{dt} \leq az + b, \quad z(0) = z_0 > 0$$

*for $a > 0$ and $b$ constants. Then*

$$z(t) \leq e^{at} z_0 + a^{-1}(e^{at} - 1)b. \tag{3.4}$$

*If $a = 0$, $z(t) \leq z_0 + bt$*

**Proof** Bear in mind that if $f(t) > g(t)$ holds on some interval $[t_0, t_1]$, then $f(t) - g(t)$ is positive on that interval, and therefore

$$\int_{t_0}^{t_1} f(t) - g(t)\, dt = \int_{t_0}^{t_1} f(t)\, dt - \int_{t_0}^{t_1} g(t)\, dt$$

is also positive. The proof of (3.2.1) is

$$\begin{aligned}
\frac{dz}{dt} - az &\leq b \\
e^{-at}\frac{dz}{dt} - ae^{-at}z &\leq e^{-at}b \\
\frac{d}{dt}\left(e^{-at}z(t)\right) &\leq e^{-at}b \\
\int_0^\tau \frac{d}{dt}\left(e^{-at}z(t)\right)\, dt &\leq \int_0^\tau e^{-at}b\, dt \\
e^{-a\tau}z(\tau) - z(0) &\leq a^{-1}(1 - e^{-a\tau})b \\
z(\tau) &\leq e^{a\tau}z_0 + a^{-1}(e^{a\tau} - 1)b.
\end{aligned}$$

The case $a = 0$ is obvious. □

A similar lemma holds for discrete series:

**Lemma 3.2.2 (Discrete Gronwall Lemma)** *Let a positive sequence* $\{z_n\}_{n=0}^N$ *satisfy*

$$z_{n+1} \leq az_n + b, \qquad \forall n = 0, \ldots, N-1$$

*for some constants* $a$ *and* $b$ *with* $a > 0$. *Then for* $a \neq 1$

$$z_n \leq a^n z_0 + \frac{b}{a-1}(a^n - 1), \qquad \forall n = 0, \ldots, N,$$

*and for* $a = 1$

$$z_n \leq nb + z_0, \qquad \forall n = 0, \ldots, N.$$

**Proof** The proof is by induction with the case $a = 1$ obvious. Take $a \neq 1$. The result holds for $n = 0$. Assume it is true for $n = k$. Then

$$z_{k+1} \leq az_k + b \leq \frac{ab}{a-1}(a^k - 1) + a^{k+1}z_0 + b = \frac{b}{a-1}(a^{k+1} - 1) + a^{k+1}z_0.$$

□

Lemma 3.2.1 can be used to bound the divergence of two solutions of a differential equation with Lipshitz continuous $f$. Suppose $y(t)$ and $z(t)$ satisfy

$$\begin{aligned} y'(t) &= f(t, y(t)), \qquad y(0) = y_0, t \in [t_0, t_0 + T] \\ z'(t) &= f(t, z(t)), \qquad z(0) = z_0, t \in [t_0, t_0 + T] \end{aligned}$$

Define $e(t) = \|y(t) - z(t)\|$. Then

$$e'(t) = \|y'(t) - z'(t)\| \leq \|f(t, y) - f(t, z)\| \leq L\|y(t) - z(t)\| = Le(t),$$

and Lemma 3.2.1 tells us that these solutions diverge by at most

$$\|y(t) - z(t)\| \leq e^{Lt}\|y_0 - z_0\| \tag{3.5}$$

so long as $y(t)$ and $z(t)$ remain in $D$.

Similarly, Lemma 3.2.2 can be used to bound the divergence of two numerical solutions. Consider Euler's method applied with stepsize $h = T/N$ to (1.3) with initial conditions $y_0$ and $z_0$

$$\begin{aligned} y_{n+1} &= y_n + hf(t_n, y_n) \\ z_{n+1} &= z_n + hf(t_n, z_n). \end{aligned}$$

Subtracting and applying the triangle inequality:

$$\begin{aligned} y_{n+1} - z_{n+1} &= y_n - z_n + h(f(t_n, y_n) - f(t_n, z_n)) \\ \|y_{n+1} - z_{n+1}\| &\leq (1 + hL)\|y_n - z_n\| \leq (1 + hL)^n\|y_0 - z_0\| \leq e^{nhL}\|y_0 - z_0\| \end{aligned} \tag{3.6}$$

## 3.3 Convergence of Euler's method, existence and uniqueness

The proof in this section is an adaptation of that of Hairer, Nørsett and Wanner. See the references at the end of the chapter.

Introduce the shorthand notation $f_n \equiv f(t_n, y_n)$. For a given integer $N > 0$, let $y_n$, $n = 0, \ldots, N$, denote the iterates of Euler's method applied to the initial value problem (1.3) with step size $h = T/N$. Then the **continuation** of the Euler iterates is defined by the piecewise linear function

$$y_N(t) = y_n + \frac{t - t_n}{h}(y_{n+1} - y_n) = y_n + (t - t_n)f_n, \qquad t_n \le t \le t_{n+1}.$$

In the rest of this chapter we will consider the cylinder

$$D = \{(t, y) \in \mathbb{R} \times \mathbb{R}^d : t_0 \le t \le t_0 + T, \|y - y_0\| \le M\}, \tag{3.7}$$

associated with the IVP (1.3). We assume the vector field $f(t, y)$ is Lipschitz with constant $L$ on $D$. Under this condition we will show that the IVP has a unique solution on $D$ and that Euler's method converges to this solution. However, the result holds for any domain upon which $f$ is Lipschitz, and which can be covered by a finite number of such cylinders (for the autonomous case, see Figure 3.1).

In light of what we have said before, it follows from the theorems below that for any continuously differentiable function $f$, the associated initial value problem has a unique solution on the closure of any bounded open set. If $f$ is Lipschitz on the whole of $\mathbb{R}^d \times \mathbb{R}$, then we say it is *globally Lipschitz*, and a unique soluiton exists everywhere.

First, since $f$ is Lipschitz on $D$, it is continuous. Let $A = \sup_{(t,y)\in D} \|f(t, y)\|$. Choose $T \le M/A$. Then from Lemma 3.2.1 for any $N$, the Euler iterates $(t_n, y_n)$ remain in $D$ for all $n = 0, 1, \ldots, N$; see Figure 3.2.

**Lemma 3.3.1** *If $f$ is Lipschitz with constant $L$ on $D$, and there exists $\varepsilon$ such that $\|f(t_0, y_0) - f(t, y)\| \le \varepsilon$ on $D$, then $\|y_N(t) - (y_0 + (t - t_0)f(t_0, y_0))\| \le \varepsilon(t - t_0)$.*

**Proof** Fix $t < T$ and let $n$ be the largest integer such that $t_n < t$. Then

$$\begin{aligned} y_N(t) - y_n &= (t - t_n)f_n \\ y_n - y_{n-1} &= hf_{n-1} \\ &\vdots \\ y_1 - y_0 &= hf_0 \end{aligned}$$

Summing these up and subtracting $(t - t_0)f_0 \equiv (t - t_n)f_0 + hf_0 + \cdots + hf_0$ gives

$$y_N(t) - y_0 - (t - t_0)f_0 = (t - t_n)(f_n - f_0) + h(f_{n-1} - f_0) + \cdots + h(f_0 - f_0).$$

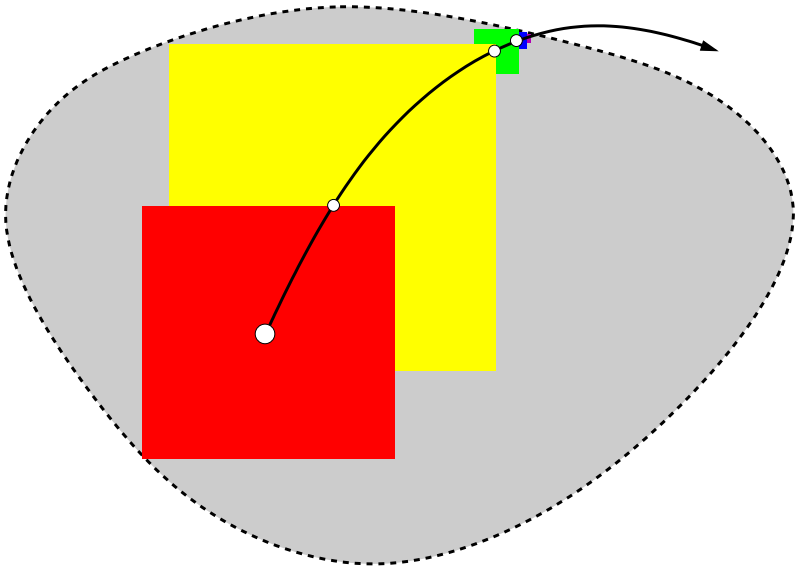


Figure 3.1: Covering of connected open set by compact rectangular sets $D$ (3.7).

Applying the triangle rule and the given bound on $f$ yields

$$\|y_N(t) - (y_0 + (t - t_0)f_0)\| \leq \varepsilon(t - t_0). \tag{3.8}$$

□

**Theorem 3.3.1** *If $f$ is Lipschitz with constant $L$ on $D$, then the continuation of the Euler iterates converges to a continuous function $Y(t)$, i.e. $\lim_{N\to\infty} y_N(t) = Y(t)$.*

**Proof** Since $f(t, y)$ is Lipschitz, it is uniformly continuous on $D$. Therefore, for every $\varepsilon > 0$ there exists $\delta > 0$ such that $|t - t^*| \leq \delta$ and $\|y - y^*\| \leq A\delta$ imply $\|f(t, y) - f(t^*, y^*)\| \leq \varepsilon$. Choose $N$ large enough that $h = T/N \leq \delta$.

Suppose we replace the first Euler step of size $h$ with two steps of size $h/2$. Denote the resulting piecewise linear continuation function by $y_{N,1}(t)$ (i.e. this function is piecewise linear with discontinous first derivative at $h/2, h, 2h, \dots$). The result (3.8) gives $\|y_N(t_1) - y_{N,1}(t_1)\| \leq \varepsilon h$. The rest of the interval is just two Euler processes with different initial conditions, so we can use (3.6) to get

$$\|y_N(t) - y_{N,1}(t)\| \leq e^{L(t-t_1)}\varepsilon h, \qquad t_1 \leq t \leq t_0 + T.$$

(Alghough strictly speaking (3.6) gives the divergence at the discrete times, our linear interpolation ensures the above formula holds at the intermediate times as well.)

Next, refine the second interval: $[t_1, t_2]$ by again taking two steps of size $h/2$. Denote the resulting continuation function (i.e., starting with 4 steps of size $h/2$, followed by steps

of size $h$) by $y_{N,2}(t)$. Note that $y_{N,1}(t)$ and $y_{N,2}(t)$ are identical on $[t_0, t_1]$. Applying (3.8) again yields $\|y_{N,1}(t_2) - y_{N,2}(t_2)\| \le \varepsilon h$. On the rest of the interval we have, due to (3.6),

$$\|y_{N,1}(t) - y_{N,2}(t)\| \le e^{L(t-t_2)}\varepsilon h, \qquad t_2 \le t \le t_0 + T.$$

We continue in this way, refining each subsequent interval until we arrive at a solution obtained by $2N$ steps of size $h/2$, which is denoted $y_{2N}(t)$, and the bound, valid on the interval $t_n \le t \le t_{n+1}$,

$$\begin{aligned}\|y_{2N}(t) - y_N(t)\| &\le \varepsilon \left( h e^{L(t-t_1)} + h e^{L(t-t_2)} + \cdots + h e^{L(t-t_n)} + (t - t_n) \right) \\ &\le \varepsilon \int_{t_0}^{t} e^{L(t-s)}\, ds = \frac{\varepsilon}{L} \left( e^{L(t-t_0)} - 1 \right), \end{aligned} \tag{3.9}$$

where the last inequality is illustrated in Figure 3.3. Since $\varepsilon$ can be made arbitrarily small, the sequence $y_N(t), y_{2N}(t), \ldots$ converges uniformly to a continuous function $Y(t)$. □

**Theorem 3.3.2** *Under the conditions of Theorem 3.3.1, the function $Y(t)$ is the unique solution of the initial value problem (1.3) on $D$.*

**Proof** To show existence, define

$$\varepsilon(\delta) = \sup \left\{ \|f(t,y) - f(t^*,y^*)\| : |t - t^*| \le \delta, \|y - y^*\| \le A\delta; (t,y), (t^*,y^*) \in D \right\}.$$

Then for a given $N$ we have, from item 3 above,

$$\|y_N(t+\delta) - y_N(t) - \delta f(t, y_N(t))\| \le \varepsilon(\delta)\delta.$$

Taking the limit $N \to \infty$ we get

$$\|Y(t+\delta) - Y(t) - \delta f(t, Y(t))\| \le \varepsilon(\delta)\delta.$$

Since $\varepsilon(\delta) \to 0$ for $\delta \to 0$, $Y$ is differentiable and satisfies $Y'(t) = f(t, Y(t))$.

To show uniqueness, suppose $Z(t)$ is a second solution to the initial value problem on $D$, and denote by $y_N^{(n)}(t)$ the piecewise linear continuation of an Euler method: applied to the initial value problem (1.2) on $[t_n, t_0 + T]$ with initial condition $Z(t_n)$ (there are $N$ such functions). Since

$$Z(t) = Z(t_n) + \int_{t_n}^{t} f(s, Z(s))\, ds$$

it follows that

$$\|Z(t) - y_N^{(n)}(t)\| \le \varepsilon |t - t_n|, \qquad t_n \le t \le t_{n+1}.$$

Using the same reasoning as for part 4 it follows that

$$\|Z(t) - y_N(t)\| \le \frac{\varepsilon}{L} \left( e^{L(t-t_0)} - 1 \right).$$

In the limits $N \to \infty$ and $\varepsilon \to 0$, this gives $Z(t) \to Y(t)$. □

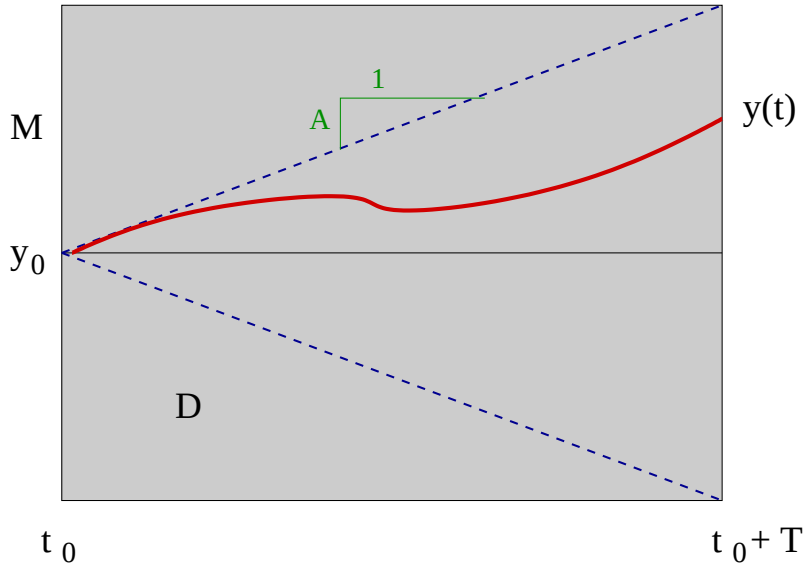



Figure 3.2: A solution with slope bounded by $A$ cannot leave $D$ in time $T \leq M/A$.

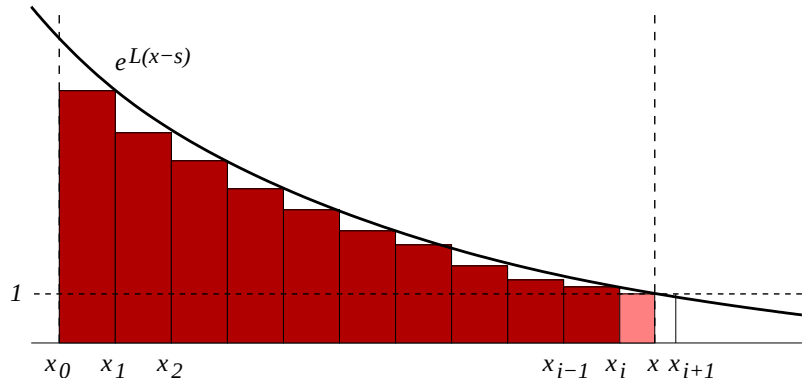



Figure 3.3: The right hand rectangle rule applied to a decreasing function (3.9).

## 3.4 References

The analysis in this chapter is adapted from Cauchy's proof as presented in

> E. Hairer, S.P. Nørsett & G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd Ed., Springer, 1993.

## 3.5 Exercises

1. Discuss the two example problems at the beginning of this Chapter in light of the theory of the last section. In what way do the examples fail to meet the prerequisites

for the theorem. Which of the conclusions of the theorem fail as a result.

2. There are other existence and uniqueness proofs for differential equations. Suppose that you are given the fact that (1.3) with Lipschitz $f$ has a unique solution $Y(t)$ on $D$. Can you construct an easier convergence proof for Euler's method. (Hint: use Taylor's theorem applied to $Y(t)$, define the error iterates $e_n = Y(t_n) - y_n$, $n = 0, \ldots, N$, and show that $\|e_n\| \leq Ch$ and therefore,

$$\lim_{N \to 0} \|e_n\| = 0.$$

3. Choose one of the models in Chapter 1 and use Theorem 3.3.2 to discuss existence/uniqueness of solutions for your model.

# Chapter 4

# The Matrix Exponential

**Main concepts:** In this chapter we solve systems of linear differential equations, introducing the matrix exponential and related functions, and the variation of constants formula.

In general it is possible to exactly solve systems of linear differential equations with constant coefficients using the variation of constant forumula. Because linear systems are of both theoretical and practical importance, will delve into this in some detail. Some numerical techniques for nonlinear problems that we will encounter later require integrating linear systems exactly. To begin, we derive the solution in the scalar case.

## 4.1 The general scalar, linear equation in the complex plane.

Consider the linear, scalar ODE with constant coefficients in $\mathbb{C}$

$$\dot{x} = \lambda x + c, \tag{4.1}$$

$\lambda$, $c \in \mathbb{C}$, $x(t) : \mathbb{R} \to \mathbb{C}$.

We solve this equations as follows:

$$\begin{aligned}
(\dot{x}(s) - \lambda x(s))\, e^{-\lambda s} &= ce^{-\lambda s}, \\
\frac{d}{dt}\left(x(s)e^{-\lambda s}\right) &= ce^{-\lambda s}, \\
\int_0^t x(s)e^{-\lambda s}\, ds &= c\int_0^t e^{-\lambda s}\, ds, \\
x(t)e^{-\lambda t} - x(0) &= -\frac{c}{\lambda}(e^{-\lambda t} - 1),
\end{aligned}$$

yielding

$$x(t) = e^{\lambda t}x(0) + \frac{1}{\lambda}(e^{\lambda t} - 1)c. \tag{4.2}$$

## 4.2 Systems of linear ODEs

Next we consider linear differential equations of the form

$$\dot{y} = Ay + b \tag{4.3}$$

where $y(t) : \mathbb{R} \to \mathbb{C}^d$, $A \in \mathbb{C}^{d\times d}$, $b \in \mathbb{C}^d$.

### 4.2.1 Diagonalizable systems

First, let us suppose that the eigenvectors of $A$ span $\mathbb{C}^d$. Thus, there are $d$ linearly independent eigenvectors $z_i \in \mathbb{C}^d$, $i = 1, \ldots, d$, and corresponding eigenvalues $\lambda_i \in \mathbb{C}$ satisfying

$$Az_i = \lambda_i z_i, \qquad i = 1, \ldots, d \tag{4.4}$$

We can construct a square matrix $Z$ whose columns are the $z_i$:

$$Z = [z_1 | \ldots | z_d]$$

and, since the $z_i$ are linearly independent, $Z$ is invertible. Let $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ be the diagonal matrix with the ordered eigenvalues on the diagonal. Equation (4.4) is equivalent to

$$AZ = Z\Lambda,$$

from which is follows from the invertibility of $Z$ that

$$A = Z\Lambda Z^{-1}.$$

Inserting this into (4.3) gives

$$\dot{y} = Z\Lambda Z^{-1}y + b.$$

Next let us introduce the transformation $x(t) = Z^{-1}y(t)$, $c = Z^{-1}b$. The equation above transforms to

$$\dot{x} = \Lambda x + c.$$

However, since $\Lambda$ is diagonal, the above differential equation can be written as the folowing system of decoupled scalar equations

$$\dot{x}^{(i)} = \lambda_i x^{(i)} + c^{(i)}, \quad i = 1, \ldots, d.$$

These are of the form (4.1) and so we can make use of the exact solution (4.2) to determine $x^{(i)}(t)$. Given the time evolution of the $x^{(i)}$ we can then reconstruct the solution from the relation $y(t) = Zx(t)$.

The case for general $A$ is not much more difficult. But we need some background on the matrix exponential function first.

### 4.2.2 The matrix exponential

The exponential of a square matrix $A \in \mathbb{C}^{d\times d}$ is defined as

$$\exp A := \sum_{k=0}^{\infty} \frac{1}{k!} A^k, \tag{4.5}$$

where $A^0 = I$, the identity matrix on $\mathbb{C}^d$.

The series in (4.5) converges for any matrix $A$. To see this, let $M > 0$ satisfy

$$|A_{ij}| \le M, \quad i, j = 1, \dots, d.$$

Note that for all $i$, $j$,

$$|(A^2)_{ij}| \le dM^2$$

and in fact

$$|(A^k)_{ij}| \le d^{k-1} M^k \le (dM)^k$$

It follows that

$$|\exp A_{ij}| \le \sum_{k=0}^{\infty} (dM)^k = e^{dM} < \infty.$$

We also use the notation $e^A$ for $\exp A$.

You can check that the matrix exponential satisfies the following properties:

- Commutativity with $A$: $Ae^{tA} = e^{tA}A$
- For any $A$, $e^A$ is nonsingular and $(e^A)^{-1} = e^{-A}$.
- For $t \in \mathbb{R}$, $\frac{d}{dt} e^{tA} = Ae^{tA}$.

### 4.2.3 Variation of constants formula

Now we return to (4.3). Equipped with the matrix exponential, we basically construct the solution as in the scalar case. We can also handle the case of time dependent $b = b(t)$

$$\begin{aligned}
(\dot{y} - Ay)e^{-As} &= e^{-As} b, \\
\frac{d}{dt}(e^{-As} y) &= e^{-As} b, \\
\int_0^t \frac{d}{dt}(e^{-As} y)\, ds &= (\int_0^t e^{-As} b(s)\, ds), \\
e^{-At} y(t) - y(0) &= (\int_0^t e^{-As} b(s)\, ds),
\end{aligned}$$

resulting in the *variation of constants formula*

$$y(t) = e^{At} y(0) + (\int_0^t e^{(t-s)A} b(s)\, ds) \tag{4.6}$$

When $b$ is constant, this simplifies to

$$y(t) = e^{At}y(0) + t\theta(At)b, \tag{4.7}$$

where for invertible $A$ and $t \neq 0$ the function $\theta(At)$ can be written

$$\theta(At) = (At)^{-1}(e^{At} - I)$$

For singular $A$, $\theta$ is given by the series expansion

$$\theta(A) = \sum_{k=0}^{\infty} \frac{1}{(k+1)!} A^k.$$

You can check that this series also converges for any matrix $A$.
**Example.** We calculate $\exp(At)$ for the matrix

$$A = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix}.$$

The eigenvectors of $A$ are $z_1 = (1,1)^T$, $z_2 = (1,-1)^T$, with corresponding eigenvalues $\lambda_1 = i\omega$, $\lambda_2 = -i\omega$. Thus,

$$Z = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \qquad Z{-}1 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix},$$

and $\Lambda = \operatorname{diag}\{i\omega, -i\omega\}$. We compute

$$\exp(At) = \exp(Z\Lambda Z^{-1}t) = Z\exp(\Lambda t)Z^{-1} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \cos\omega t + i\sin\omega t & 0 \\ 0 & \cos\omega t - i\sin\omega t \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

, or

$$\exp(At) = \begin{bmatrix} \cos\omega t & \sin\omega t \\ -\sin\omega t & \cos\omega t \end{bmatrix} \tag{4.8}$$

**Exercise**. Consider the system

$$\begin{pmatrix} \frac{dq}{dt} \\ \frac{dp}{dt} \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -k^2 & 0 \end{bmatrix} \begin{pmatrix} q \\ p \end{pmatrix} = A \begin{pmatrix} q \\ p \end{pmatrix}.$$

Show that the exponential of $A$ in this example is

$$\exp tA = \begin{bmatrix} \cos kt & k^{-1}\sin kt \\ -k\sin kt & \cos kt \end{bmatrix}. \tag{4.9}$$

## 4.3 Computing the exponential of a matrix

Scaled and squared Padé approximation, exercise caution, counterexample ...

# Chapter 5

# Flow Maps and Dynamical Systems

**Main concepts:** In this chapter we introduce the concepts of continuous and discrete dynamical systems on phase space. Keywords are: classical mechanics, phase space, vector field, linear systems, flow maps, dynamical systems

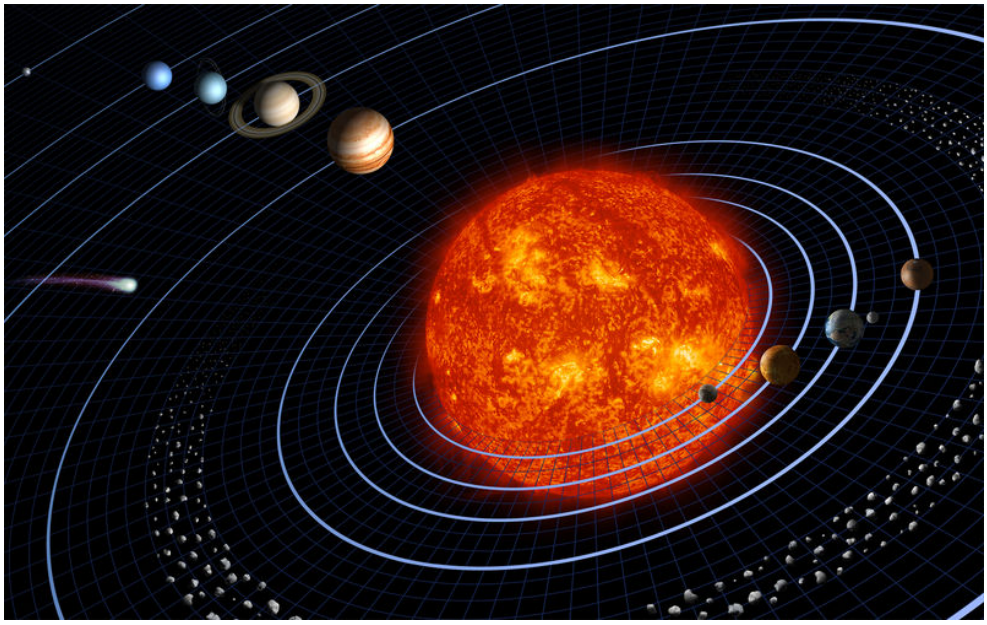


Figure 5.1: A solar system is well modelled by classical mechanics. (source: Wikimedia Commons)

In Chapter 1 we encountered a variety of ODEs in one or several variables. Only in a few cases could we write out the solution in a simple closed form. In most cases, the solutions will only be obtainable in some approximate sense, either from an asymptotic expansion or a numerical computation. Yet, as discussed in Chapter 3, many smooth systems of differential equations are known to have solutions, even globally defined ones,

and so in principle we can suppose the existence of a trajectory through each point of phase space (or through "almost all" initial points). For such systems we will use the existence of such a solution to define a map called the *flow map* that takes points forward $t$ units in time from their initial points. The concept of the flow map is very helpful in exploring the qualitative behavior of numerical methods, since it is often possible to think of numerical methods as approximations of the flow map and to evaluate methods by comparing the properties of the map associated to the numerical scheme to those of the flow map.

In this chapter we will address autonomous ODEs (recall 1.13) only:

$$y' = f(y), \qquad y, f \in \mathbb{R}^d \tag{5.1}$$

## 5.1 Classical mechanics

In Chapter 1 we introduced models from population dynamics. In this chapter we consider another field that is rich with dynamical systems: *classical mechanics*. Classical mechanics consists of those phenomena that can be accurately modelled using Newton's laws of motion, the second of which is:

> *The time rate of change of momentum of a body is equal to the net force applied to it.*

If the position of the body in question is given by $x(t) \in \mathbb{R}^3$, its velocity is $\dot{x}(t) \equiv \frac{dx}{dt}$, and its momentum $p(t)$ is the product of mass $m$ (assumed constant here) and velocity. Assuming the force $F$ is given as a field on $\mathbb{R}^3$, the mathematical statement of the first law is

$$\frac{dp}{dt} = \frac{d}{dt}\left(m\dot{x}(t)\right) = F(x(t)).$$

The momentum can be eliminated in favor of the acceleration $\ddot{x}(t) \equiv \frac{d^2x}{dt^2}$ to give

$$m\ddot{x}(t) = F(x(t)). \tag{5.2}$$

Because of the second derivative, this is an example of a *second order differential equation*. In general we can write any *higher order differential equation* of the form

$$\frac{d^p x}{dt^p} = g(t, x, \dot{x}, \ddot{x}, \ldots, \frac{d^{p-1}x}{dt^{p-1}})$$

in the first order form (1.2) by introducing additional dependent variables. Let $X^{(1)}$ denote $x(t)$, $X^{(2)}$ denote $\dot{x}(t)$, etc. Then

$$\begin{aligned}
\dot{X}^{(1)} &= X^{(2)} \\
\dot{X}^{(2)} &= X^{(3)} \\
&\vdots \\
\dot{X}^{(p)} &= g(t, X^{(1)}, X^{(2)}, \ldots, X^{(p)}).
\end{aligned}$$

Using the momentum $p(t) = m\dot{x}(t)$, Newton's second law can be written as the first order system

$$\dot{x}(t) = \frac{1}{m}p(t) \tag{5.3a}$$
$$\dot{p}(t) = F(x(t)). \tag{5.3b}$$

To define an initial value problem for this system, we need to specify the initial position $x_0$ and momentum $p_0$ at time $t = 0$.

Let us consider the specific example of a mass on a spring. The force exerted by a spring is proportional to its displacement (this is essentially the definition of a "spring" in mechanics).

$$F(x(t)) = -k^2 x(t), \quad x, F \in \mathbb{R}$$

Applying Newton's second law gives

$$m\ddot{x} = -k^2 x$$

Assume $m = 1$. The first order system that arises is also known as the *harmonic oscillator*

$$\dot{x} = p, \qquad \dot{p} = -k^2 x. \tag{5.4}$$

The solution

$$x(t) = x_0 \cos kt + \frac{p_0}{k} \sin kt.$$

is periodic with frequency $k$.

More complex mechanical models have nonlinear forces $F(x(t))$ and may involve the coupling of a number of bodies. For example, in molecular dynamics, one may study the motion of a large number of interacting atomic particles. Atomic motion has several different types of forces. There are long range attracting forces and short range repelling forces. The Lennard-Jones model incorporates both of these. If we consider two particles iteracting on a line via a Lennard-Jones potential, and take one particle fixed in space, the force on the other particle at a distance $x(t) \in \mathbb{R}$ away is $F(x) = 12(x^{-13} - x^{-7})$. This yields the *Lennard-Jones oscillator*

$$\dot{x} = p, \qquad \dot{p} = 12(x^{-13} - x^{-7}). \tag{5.5}$$

The motion $x(t) \in \mathbb{R}^2$ of a satellite of mass $m$ subject to the gravitational pull of a cellestial body positioned at the origin is governed by the *Kepler problem*

$$\dot{x} = \frac{1}{m}p, \qquad \dot{p} = -g\frac{x}{(x^T x)^{3/2}}. \tag{5.6}$$

This system has dimension $d = 4$.

## 5.2 Phase Space

In multidimensional autonomous systems (5.1) we sometimes refer to the underlying space $\mathbb{R}^d$ as the **phase space**. The *phase portrait* for a differential equation is the collective graph of trajectories of the differential equation in the phase plane. Obviously we cannot draw all the trajectories, so in typical parlance, the phase portrait is just a simplified graph showing several trajectories. The phase portrait for the Lennard-Jones oscillator (5.5) in $\mathbb{R}^2$ is shown in the left panel of Figure 5.2. For this system, all of the trajectories are closed curves, indicating that the solutions of the Lennard-Jones oscillator are periodic in time.

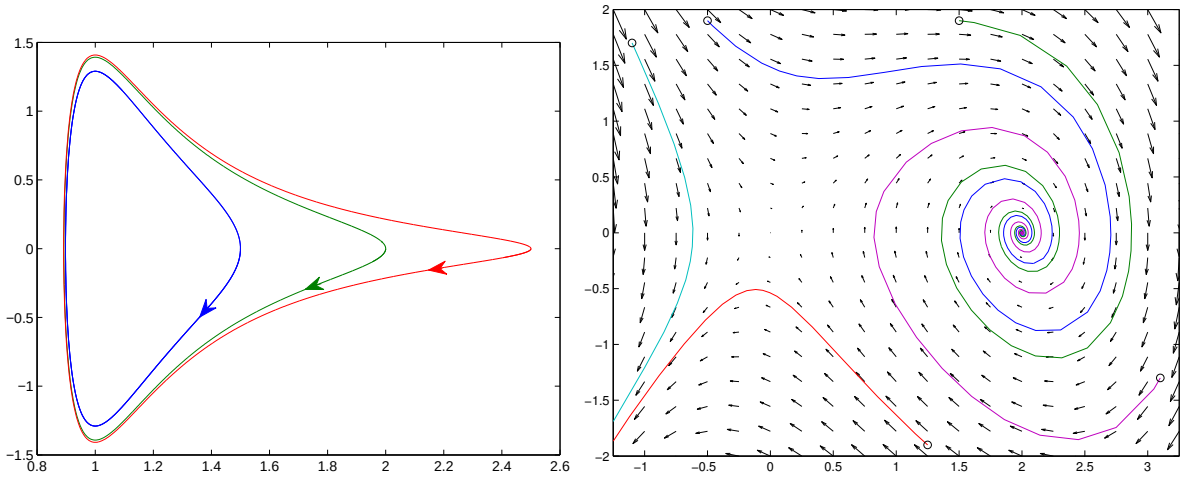



Figure 5.2: On the left, the phase portrait of the Lennard-Jones oscillator (5.5). On the right, vector field for the differential equation $\ddot{x} = 2x - x^2 - \dot{x}/2$ with some trajectories.

Two trajectories in phase space may not intersect, as this would imply that the solution is nonunique at the point of intersection. Furthermore, the solution traverses a curve in phase space in one direction only. It cannot "turn around" as this too would contradict uniqueness.

The function $f(y) : \mathbb{R}^d \to \mathbb{R}^d$ defines a **vector field** on the phase space. That is, to each point $y \in \mathbb{R}^d$ is associated a vector $f(y) \in \mathbb{R}^d$. An example for a representative ODE is shown in the right panel of Figure 5.2. Every solution of (5.1) is everywhere tangent to the vector field $f$, as depicted in the figure. A solution is said to be a *trajectory* or *orbit* or *integral curve* of the vector field.

## 5.3 The Group of Flow Maps

Consider the solution (4.7) of the linear ODE (4.3). One way of using this function is to compute the time evolution of an initial value problem associated with (4.3). In other words, given $y_0$, the function

$$y(t) = \exp(tA)y_0 + t\theta(tA)b, \qquad 0 \le t \le T,$$

maps out a continuous curve in the phase space $\mathbb{R}^d$.

However, we could use the solution in a different way. Specifically, consider a fixed time $t = \tau$ and the function

$$\Phi_\tau(y) = \exp(\tau A)y + \tau\theta(\tau A)b :$$

The function $\Phi_\tau : \mathbb{R}^d \to \mathbb{R}^d$ maps initial conditions of (4.3) to final conditions after an interval $[0, \tau]$.

To generalize this idea, let us take an autonomous ODE system (5.1). Given a fixed time $\tau$, we can consider any point of phase space, $y_0$, as a starting point of a trajectory $(y(t; y_0))$ which is continued up to time $\tau$, assuming the solution exists on the entire interval. We think of the action of solving the differential equation as defining a map from starting points of trajectories to their endpoints. Associated to a differential equation (5.1) we define the **flow map**

$$\Phi_\tau(y_0) = y(\tau)$$

where $y(t)$ is the solution of the initial value problem

$$y'(t) = f(y), \quad y(0) = y_0, \quad t \in [0, \tau].$$

When the system is nonlinear there is usually not a simple formula for the flow map. Nonetheless the concept is valuable.

Consider solving the ODE (5.1) with initial condition $y(0) = y_0$ for a time $t_1$. Denote solution at $t_1$ by $y_1$. Next, solve initial value problem (5.1) with initial condition $y(0) = y_1$ on the interval $[0, t_2]$. Denote the solution at time $t_2$ by $y_2$. Because of the invariance of the autonomous system to a translation of time, it is clear that $y_2$ is also the same solution you would obtain if you would solve (5.1) with initial condition $y(0) = y_0$ over the interval $[0, t_1 + t_2]$. In terms of the flow map,

$$\Phi_{t_2} \circ \Phi_{t_1} = \Phi_{t_1+t_2} = \Phi_{t_1} \circ \Phi_{t_2}, \qquad t_1, t_2 > 0.$$

Thus the composition of the two maps is another map of the same family. If we can solve the differential equations for all positive or negative time, then we can write

$$\Phi_t \circ \Phi_{-t} = \Phi_{t+-t} = \Phi_0,$$

but $\Phi_0$ is evaluated by solving the differential equations over a zero length interval, so is just the identity map. Hence we see that $\Phi_{-t} = \Phi_t^{-1}$. The set of all such maps $\{\Phi_t \,|\, t \in \mathbf{R}\}$ is what is termed a *one-parameter group* with the (commutative) group operation being composition of maps.

The flow map can also be applied to an open set $B \subset \mathbb{R}^d$, and then we view the flow map as a function taking sets on phase space to other sets on phase space, $t$ time units later. In other words, each element of $B$ is taken as the initial condition of (5.1), and is mapped to the solution at the end of the interval of time $t$.

$$\Phi_\tau(B) = \{z \in \mathbb{R}^d : z = \Phi_\tau(y); y \in B\}$$

Implicit in this usage is the assumption that the trajectories through every point of $B$ exist for at least $t$ units of time.

For example, in Figure 5.3 a rectangular set of initial conditions in $\mathbb{R}^2$ is mapped under the flow map $\Phi_\tau$ of the Lennard-Jones oscillator (5.5) for $\tau = 0$, 0.35, 0.9. In general, and especially in higher dimensions, such a set will become stretched and folded in on itself in a very complicated way.

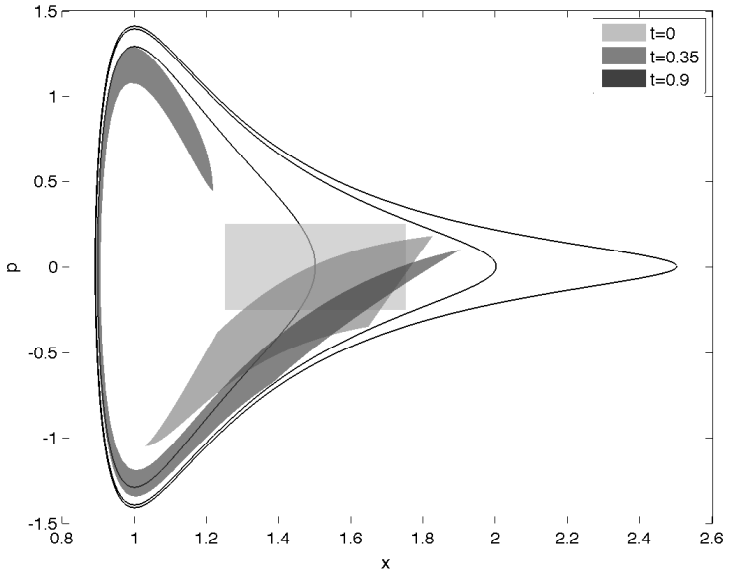



Figure 5.3: Evolution of a rectangular set $B$ under the flow map of the Lennard-Jones oscillator (5.5) at times $t = 0$ (yellow), $t = 0.35$ (red) and $t = 0.9$ (blue).

## 5.4 Dynamical systems

It is important to note that the flow map $\Phi_\tau(y)$ only makes sense if the solution through $y$ exists up to time $t$. In Chapter 3 we showed that a sufficient condition for existence of a solution on a set $D \subseteq \mathbb{R}^d$ is that the vector field $f$ be Lipschitz on $D$. However, this condition alone does not specify to what time $t$ the solution will continue to exist. We will be interested in two special cases in which the solution is known to exist for all time $t > 0$. First, we say a vector field $f$ is **globally Lipschitz** if it is Lipschitz with constant $L < \infty$ on all of $\mathbb{R}^d$. In this case, a solution through any point $y$ exists and is unique for all time. This is often not a very realistic assumption. Instead, consider a set $D$ with the property: A set $D \subset \mathbb{R}^d$ is said to be **positively invariant** under the flow map $\Phi_t$ if

$$\Phi_t(D) \subseteq D \qquad \forall t \geq 0.$$

(The stipulation "for all $t > 0$" is not really restrictive. It is sufficient if $\Phi_t : D \to D$ on any interval $[0, \tau)$, $\tau > 0$, since this implies invariance for all $t > 0$.) If $f$ is Lipschitz on

a set $D$ that is positively invariant under the flow map, then there the unique solution for any initial condition $y_0 \in D$ exists for all time. Given a set $D \subseteq \mathbb{R}^d$, a map $\Phi_t : D \to D$ defines a **dynamical system** on $D$. Clearly the flow map of a differential equation defines a dynamical system whenever the vector field $f$ is globally Lipschitz, or Lipschitz on a positively invariant set.

For some differential equations, the solution may also exist for negative time $t < 0$ and hence for all time $t \in \mathbb{R}$. If $D$ is invariant under the negative time flow of a differential equation, we say $D$ is *negatively invariant*. A set that is both positively and negatively invariant is said to be *invariant*.

The **$\omega$-limit set** of $y$ is the set of points $x$ such that there exists a sequence of times $t_n \to \infty$, such that $\Phi_{t_n}(y) \to x$. This set is denoted $\omega(y)$. Similarly the **$\alpha$-limit set** of $y$ is the set of such points for $t_n \to -\infty$. Examples of such sets are equilibrium points, limit cycles, periodic orbits and attractors. We also use $\omega(B)$ and $\alpha(B)$ to denote the union of the $\omega$ and $\alpha$ limit sets of all initial conditions in an open set $B$.

For example, in the right panel of Figure 1.2, solutions of the predation model (1.5) are shown. For the values $r = 0.1$ and $r = 0.6$, the set $D = \{n_0 > 0\}$ has $\omega(D)$ a single point. For $r = 0.25$, $\omega(D)$ contains two points. For the Lotka-Volterra model (1.6), all solutions are periodic, the $\omega$ and $\alpha$ limit sets coincide, and are precisely the solution curves.

## 5.5 Discretization and Approximate Flow Maps

When we discretize an ODE, we normally replace it by a difference equation which describes in an approximate sense how the solution evolves from timestep to timestep. For example, recall Euler's method (2.1) from Chapter 2

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

Here $y_n$ represents a numerical approximation of $y(t_n)$.

We could regularly sample the exact solution on a succession of time intervals of length $h$, according to the rule

$$y(t_{n+1}) = \Phi_h(y(t_n))$$

If $y(t; y_0)$ represents the solution of the differential equation (5.1) with initial condition $y(0) = y_0$ after $t$ units of time, we can write

$$y(h; y_0) = \Phi_h(y_0).$$

Moreover,

$$y(2h; y_0) = \Phi_h \circ \Phi_h(y_0),$$

and so on. This means that we can view the iteration of $\Phi_h$ as producing snapshots of the solution at equally spaced points in time. Similarly, Euler's method can be seen as iteration of a map

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

which approximates the flow map $\Phi_h$. We refer to $\Psi_h$ as the **numerical flow map** for Euler's method.

Do the mappings $\Psi_h$ form a one-parameter group? The answer is no. Quite simply,

$$\Psi_{h_1} \circ \Psi_{h_2} \neq \Psi_{h_1+h_2}.$$

Indeed, even

$$\Psi_h \circ \Psi_h \neq \Psi_{2h}.$$

This is a fundamental difference between the exact flow map and its numerical approximation.

There are a wide variety of numerical methods available for solving the ODE (1.2) or (5.1). In some cases these are based on using the values of solutions computed at two or more successive points in time (so-called *multistep methods*). While such schemes offer benefits for certain types of computations, only in the case that the numerical method computes the approximation based only on the approximate solution at the previous time point can we directly compare the method's properties to those of the flow map. For this reason, we restrict in this course to *generalized one-step methods* which can always be associated to the recurrence relation

$$y_{n+1} = \Psi_h(y_n). \tag{5.7}$$

The discrete approximation would satisfy

$$y_1 = \Psi_h(y_0), \qquad y_2 = \Psi_h(y_1) = \Psi_h \circ \Psi_h(y_0),$$

and so on. It is useful to have a notation for applying a map $n$ times, recursively. We will use a superscript to indicate this:

$$\Psi_h^2 = \Psi_h \circ \Psi_h, \qquad \Psi_h^3 = \Psi_h \circ \Psi_h \circ \Psi_h,$$

etc.

For explicit methods such as Euler's method, the numerical flow map is obvious. For implicit methods such as the Trapezoidal rule (2.2) introduced in Chapter 2, it is unclear in advance whether such a map is well-defined. It follows from the implicit function theorem that for $h$ small enough and appropriate $f$, there exists a solution to the trapezoidal rule, but it may not be unique. Typically, there is a branch of solutions that converge to $y_n$ as $h \to 0$. If we take this branch, then the flow map (5.7) is well-defined.

As in the case of the continuous flow map, we use the notation $\Psi_h(B)$ to denote the set obtained by applying $\Psi_h$ to every element of an open set $B$. Given a set $D \subseteq \mathbb{R}^d$, if a discrete map $\Psi_h$ satisfies $\Psi_h(D) \subseteq D$, then it defines a **discrete dynamical system** on $D$.

In this course we will study the dynamics of numerical flows $\Psi_h$. This will give us an appreciation of the qualitative behavior of different methods. Generally we will attempt to compare the properties of $\Psi_h$ with those of the exact flow $\Phi_t$. However, since the numerical

map does not satisfy the group property, the size of $h$ plays a special role in numerics. In particular there are practical restrictions on the maximum allowable step size $h$. And in general we will have to compare the iterated map $\Psi_h^n$ with the exact flow $\Phi_{nh}$. In the theory of continuous dynamical systems, one is often concerned with the limit behavior of solutions as $t \to \pm\infty$, and roughly speaking, we would like to recover this behavior in the iterated numerical map. Besides the dynamics of the numerical flow map, however, there is an additional issue—unique to numerics—of *approximation*. Whereas when studying dynamics we are often interested in the *dynamic limit*

$$T \to \infty, \quad h \text{ fixed}, \tag{5.8}$$

when studying approximation we look instead at the *approximation limit*

$$h \to 0, \quad T \text{ fixed}. \tag{5.9}$$

Before turning to the dynamical systems limit, we will first be concerned with the approximation limit in the next few chapters.

## 5.6 References

The definitive work on dynamical systems aspects of numerical integrators is

> A. M. Stuart and A. R. Humphries, *Dynamical Systems and Numerical Analysis*, Cambridge, 1996.

## 5.7 Exercises

1. Can a globally Lipschitz vector field have a solution that grows unbounded? Give an example or counterexample to support your answer.

2. Show that for any $H$ the ellipse $E(H) = \{(x, p) : k^2x^2 + p^2 = H\}$ is invariant under the flow map of the harmonic oscillator (5.4). Is such an ellipse also invariant under the flow map of Euler's method (2.1) applied to this problem? Trapezoidal rule (2.2)?

3. Show that any ellipsoidal disc $D(H) = \{(x, p) : k^2x^2 + p^2 \leq H\}$ is invariant under the flow map of the damped spring (**??**). Is such a disc also invariant under the flow map of Euler's method applied to this problem? For what range of stepsizes?

4. Consider the logistic equation (1.4) and the open interval $X = (0, 1)$. What is $\omega(X)$? What is $\alpha(X)$?

# Chapter 6

# 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. Keywords are: Taylor series, local error, convergence theory for one-step methods.

## 6.1 Approximation

A numerical method is an approximation of the exact flow map of a differential equation. As such, there is always an element of error in the numerically generated solution. If the method is to be useful, we must be able to control this error, at least on short enough time intervals. In this chapter we consider convergence of the numerical flow map $\Psi_h$ to the exact flow map $\Phi_h$ in the approximation limit $h \to 0$.

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

$$e_n := y_n - y(t_n). \tag{6.1}$$

For an approximation we want the error to be small in norm at each step of simulation, that is, we would like to satisfy

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

for some user-specified tolerance $\delta$. For a given vector field $f$, initial value $y_0$ and time interval $T$, we have only one free parameter, the timestep $h = T/N$, which we can vary to make sure the norm of the error meets our tolerance. If the method is going to be useful, we must be able to vary $h$ to meet any tolerance we choose.

Given a Lipschitz vector field $f$, a method is said to be *convergent* if, for every $T$,

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

(Note that this definition considers only discrete values of $h$ which are integral fractions of the time interval. Equivalently we could take the limit as $N \to \infty$ with $h = T/N$.) In this chapter we 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 which the relevant form here is

$$y(t+\tau) = y(t) + \tau y'(t) + \frac{\tau^2}{2} y''(t) + \cdots \tag{6.2}$$

for a perturbation $\tau > 0$ around $t$. For a *scalar function* ($d = 1$), assuming $y(t)$ is $\nu$-times continuously differentiable, Taylor's theorem says there is a point $t^* \in [t, t+\tau]$ such that

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

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

$$\| \frac{\tau^\nu}{\nu!} \frac{d^\nu y}{dt^\nu}(t) \| \leq C\tau^\nu$$

so we write

$$y(t+\tau) = \sum_{i=0}^{\nu-1} \frac{\tau^i}{i!} \frac{d^i y}{dt^i}(t) + \mathcal{O}(\tau^\nu). \tag{6.3}$$

in general.

## 6.2 Convergence of generalized one-step methods

Define the **local error** of a 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), and that it satisfies

$$\|le(y, h)\| \leq Ch^{p+1}, \tag{6.4}$$

where $C$ is a constant that depends on $y(t)$ and its derivatives, and $p \geq 1$. A method that meets this criterion is said to be **consistent**.

We will further suppose that $f$ is Lipschitz with constant $L$.

The error can be viewed as the difference between $n$ iterations of $\Psi_h$ and $n$ iterations of $\Phi_h$, thus we define it to be

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

so

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

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

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

Now we use assumption (6.4) to bound the first term, and the Gronwall Lemma (3.2.1) to bound the second term (cf. 3.5), obtaining

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

Next applying the discrete Gronwall Lemma (3.2.2) yields

$$\|e_n\| \leq e^{Lhn}\|e_0\| + \frac{Ch^{p+1}}{e^{Lh} - 1}(e^{Lhn} - 1).$$

Note that for positive $h$ and $L$,

$$\frac{1}{e^{hL} - 1} \leq \frac{1}{hL}.$$

Furthermore, on our integration interval $0 \leq nh \leq T$ implies $e^{Lnh} \leq e^{LT}$. If we assume the initial condition is exact $e_0 = 0$, we get the uniform bound

$$\|e_n\| \leq h^p \frac{C}{L}(e^{LT} - 1), \tag{6.5}$$

which proves convergence at order $p$. We summarize this result in an important convergence theorem:

**Theorem 6.2.1 (Convergence of One-Step Methods)** *Given a differential equation (1.13) with Lipschitz vector field $f$ and a consistent one-step method $\Psi_h$, the global error satisfies*

$$\max_{n=0,\ldots,N=T/h} \|\Psi_h^n(y_0) - \Phi_{hn}(y_0)\| = \mathcal{O}(h^p).$$

This theorem is powerful. Without specifying anything about the structure of the method, it guarantees the convergence of any consistent one-step method.

As an example, let us use Theorem 6.2.1 to prove the convergence of Euler's method (2.1) 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 $L = \max_D \|\frac{\partial f}{\partial y}\|$).

The exact solution satisfies

$$\Phi_h(y) = y + h\frac{dy}{dt} + \frac{h^2}{2}\frac{d^2y}{dt^2} + \mathcal{O}(h^3) = y + hf(y) + \frac{h^2}{2}\frac{d^2y}{dt^2} + \mathcal{O}(h^3)$$

Therefore the local error is

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

and we can apply Theorem 6.2.1 with $C = \max_D \|\frac{d^2y}{dt^2}\|$ to show that Euler's method is convergent with order $p = 1$.

In the proof of the Theorem 6.2.1, the relation (6.5) 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 halving the stepsize reduces the error by a factor of two. We say for this reason that Euler's method is *1st order accurate*. The error incurred in each time step is $\mathcal{O}(h^{p+1})$, and in fact this bound holds for any fixed number of time steps. 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 $1/h$.

Note the proof suggests that—although in the approximation limit $h \to 0$, $T$ fixed, the error can be made as small as possible—in the dynamics limit $T \to \infty$, $h$ fixed, the global error may grow at an exponential rate.

## 6.3 Exercises

1. Use Theorem 6.2.1 to establish convergence of the trapezoidal rule (2.2) and the explicit trapezoidal rule:

$$y_{n+1} = y_n + \frac{h}{2}\left[ f(y_n) + f\left( y_n + hf(y_n) \right) \right]$$

for a vector field $f$ with Lipschitz constant $L$.

# Chapter 7

# Collocation Methods

**Main concepts:** Polynomial interpolation, quadrature, collocation methods

## 7.1 Construction of one-step methods

One-step methods for (1.3) can be constructed in a variety of ways. A natural approach is to integrate both sides of the differential equation over one timestep,

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

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

$$\int_t^{t+h} f(y(\tau))d\tau \approx hf(y(t)). \tag{7.1}$$

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 (7.1) leads to Euler's method (2.1).

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 approximation to the integral is

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

which results in the trapezoidal rule numerical method (2.2).

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

## 7.2 Polynomial interpolation

Let $\mathbb{P}_s$ denote the space of real polynomials of degree $\leq s$. Given and 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 $\ell_i$, $i = 1, \ldots, s$ for a set of abscissae are defined by

$$\ell_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{s} \frac{x - c_j}{c_i - c_j}. \tag{7.2}$$

The $i$th Lagrange interpolating polynomial $\ell_i$ 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

$$P(x) = \sum_{i=1}^{s} g_i \ell_i(x). \tag{7.3}$$

## 7.3 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

$$b_i = \int_0^1 \ell_i(x)\, dx,$$

the quadrature formula becomes

$$\int_0^1 g(x)\, dx \approx \int_0^1 P(x)\, dx = \sum_{i=1}^{s} b_i g(c_i). \tag{7.4}$$

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 (7.4) becomes

$$\int_{t_0}^{t_0+h} g(t)\, dt = \int_0^1 g(t_0 + hx)\, h dx \approx h \sum_{i=1}^{s} b_i g(t_0 + hc_i). \tag{7.5}$$

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 $p = 2s$.

## 7.4 One-step collocation methods

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

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

$$\begin{aligned} u(t_0) &= y_0 \\ u'(t_0 + c_i h) &= f(u(t_0 + c_i h)), \quad i = 1, \ldots, s \end{aligned} \tag{7.6}$$

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 contruct a polynomial that passes through $y_0$ and agrees with the ODE (1.3) at $s$ nodes on $[t_0, t_1]$. Then we let the numerical solution be the value of this polynomial at $t_1$. See Figure 7.1.

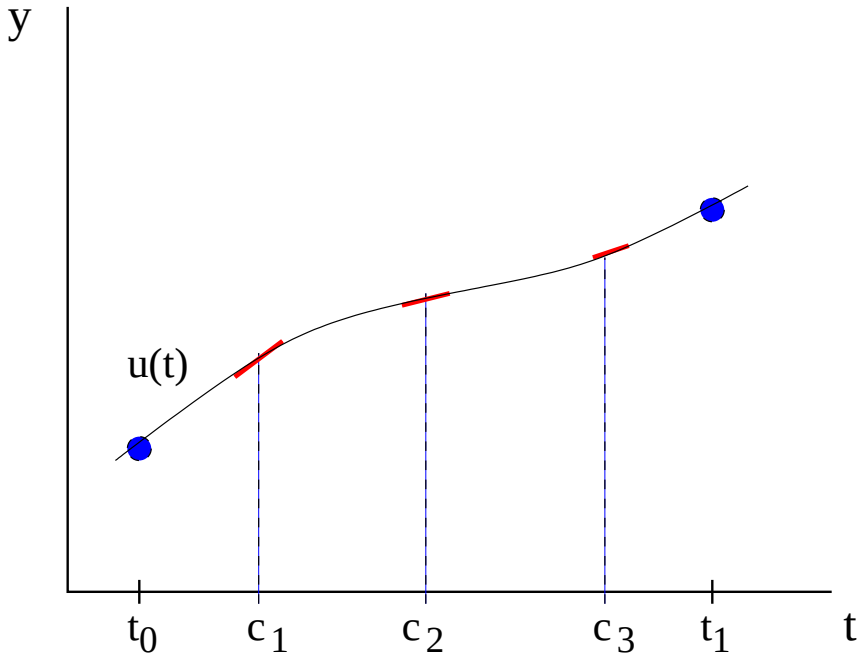



Figure 7.1: The collocation polynomial $u(t)$.

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 (7.3) to define the polynomial $u'(t)$ passing through these points:

$$u'(t) = \sum_{i=1}^{s} F_i \ell_i(\frac{t - t_0}{h}) \tag{7.7}$$

Integrating this equation over the intervals $[0, c_i]$ (and introducing the change of variables $x = (t - t_0)/h$, i.e., $dt = h\,dx$) gives

$$u(t_0 + c_i h) = y_0 + h \sum_{j=1}^{s} F_j \int_0^{c_i} \ell_j(x)\,dx, \quad i = 1, \ldots, s. \tag{7.8}$$

Let us denote

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

Substituting (7.8) into the collocation conditions (7.6) gives

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

Similarly integrating (7.7) over $[0, 1]$ gives

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

To summarize, the collocation method is written

$$F_i = f\left(y_n + h \sum_{j=1}^{s} a_{ij} F_j\right), \quad i = 1, \ldots, s, \tag{7.9a}$$

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i F_i. \tag{7.9b}$$

One first solves the coupled, $sd$-dimensional nonlinear system (7.9a). Subsequently, the update (7.9b) is explicit.

**Remark 7.4.1** *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 7.4.2** *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:*

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

In the next chapter, we consider the class of Runge-Kutta methods, of which the collocation methods presented here are but a small subclass. For this reason, the convergence analysis of collocation methods is postponed to the next chapter.

## 7.5 Exercises

1. Find the coefficients $a_{ij}$, $b_i$ of the two-stage ($s = 2$) Gauss-Legendre collocation method. (Use Maple to evaluate the integrals).

# Chapter 8

# Runge-Kutta Methods

**Main concepts:** Generalized collocation method, consistency, order conditions

In this chapter we introduce the most important class of one-step methods that are generically applicable to ODES (1.2). The formulas describing Runge-Kutta methods look the same as those of the collocation methods of the previous chapter, but are abstracted away from the ideas of quadrature and collocation. In particular, the "quadrature nodes" need no longer be distinct and collocation conditions need not hold at each stage.

## 8.1 The Family of Runge-Kutta Methods

Recall from the previous chapter the definion of a collocation method (7.9)

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

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

For 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 $a_{ij}$, $i = 1, \ldots, s$ 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

[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.

have,

$$Y_i = y_n + h\sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, \ldots, s, \tag{8.1a}$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i f(Y_i). \tag{8.1b}$$

Here, $s$ is termed the *number of stages* of the Runge-Kutta method, the $b_i$, $i = 1, \ldots, s$ are the *weights*, and $a_{ij}$ are the *internal coefficients*.

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 (8.1b)-(8.1a) with $s = 1$, $b_1 = 1$, $a_{11} = 0$. Trapezoidal rule has $s = 1$, $b_1 = b_2 = 1/2$, $a_{11} = a_{12} = 0$, $a_{21} = a_{22} = 1/2$.

Each Runge-Kutta method generates an approximation of the flow map. To see this, note that from the formula (8.1a) 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

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

### 8.1.1 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:

$$\begin{aligned} Y_1 &= y_n, \\ Y_2 &= y_n + \frac{2}{3}hf(Y_1), \\ y_{n+1} &= y_n + h\left(\frac{1}{4}f(Y_1) + \frac{3}{4}f(Y_2)\right). \end{aligned} \tag{8.2}$$

The vectors $Y_1$ and $Y_2$ are called *stage vectors*. They are intermediate values computed within each step and discarded at the end of the step. Another example with $s = 2$ is the popular method due to Heun:

$$\begin{aligned} Y_1 &= y_n, \\ Y_2 &= y_n + hf(Y_1), \\ y_{n+1} &= y_n + \frac{h}{2}(f(Y_1) + f(Y_2)). \end{aligned} \tag{8.3}$$

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

$$\begin{aligned} Y_1 &= y_n, \\ Y_2 &= y_n + \frac{h}{2} f(Y_1), \\ Y_3 &= y_n + \frac{h}{2} f(Y_2), \\ Y_4 &= y_n + h f(Y_3), \\ y_{n+1} &= y_n + h\left(\frac{1}{6} f(Y_1) + \frac{1}{3} f(Y_2) + \frac{1}{3} f(Y_3) + \frac{1}{6} f(Y_4)\right). \end{aligned} \tag{8.4}$$

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 (8.2), (8.3) and (8.4) listed in this section are represented by

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

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. A method for which the matrix $A = (a_{ij})$ is strictly lower triangular is explicit. A simple example of an implicit RK method is the *implicit midpoint rule*

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

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):

$$y_{n+1} = y_n + h f\left(\frac{y_{n+1} + y_n}{2}\right) \tag{8.5}$$

The class of collocation methods from the previous section are a subset of the class of Runge-Kutta methods.

### 8.1.2 Local accuracy of Runge-Kutta methods

We will now investigate the accuracy of the methods introduced above in terms of their approximation of the numerical solution over one timestep.

To do so, we need to work with higher order derivatives of the function $f(y) : \mathbb{R}^d \to \mathbb{R}^d$. We will denote by $f'$ the Jacobian matrix

$$f' = (\frac{\partial f_i}{\partial y_j}),$$

which can be seen as an linear operator: $\mathbb{R}^d \to \mathbb{R}^d$.

The second derivative

$$f'' = (\frac{\partial^2 f_i}{\partial y_j \partial y_k})$$

is a bilinear operator. We denote its contraction by

$$f''(a,b) = \sum_{j,k} \frac{\partial^2 f_i}{\partial y_j \partial y_k} a_j b_k, \qquad a, b \in \mathbb{R}^d.$$

This contraction is symmetric by the equivalence of mixed partial derivatives, so the order of the arguments $a$ and $b$ is irrelevant. The third derivative is similarly trilinear and symmetric in all three arguments, denoted $f'''(\cdot,\cdot,\cdot)$.

In general, we cannot have an exact expression for $le(y;h)$, but we can approximate this by computing its Taylor series in powers of $h$. For the continuous dynamics (i.e. the exact solution of (1.13)) we have the Taylor series expansion

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

The derivatives can be related directly to the solution itself by using the differential equation

$$\begin{aligned} y'(t) &= f(y(t)) \\ y''(t) &= f'(y(t))y'(t) \\ y'''(t) &= f''(y(t))(y'(t), y'(t)) + f'(y(t))y''(t) \\ &\vdots \end{aligned}$$

Then we can recursively use the differential equation again to obtain

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

where we have dropped the arguments of the various expressions. In all cases, $y$ and its derivatives are assumed to be evaluated at $t$ and $f$ and its derivatives at $y$.

Alternatively, we can write

$$\Phi_h(y) = y + hf + \frac{h^2}{2} f'f + \frac{h^3}{6} [f''(f,f) + f'ff] + \mathcal{O}(h^4) \tag{8.6}$$

The same procedure can be carried out for the Runge-Kutta method itself. For example, for Euler's method, we have

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

(that is it!). This means that the discrete and continuous series match to $\mathcal{O}(h^2)$ and the local error expansion can be written

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

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

For 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

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

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

$$z' = dz/dh = \frac{1}{2} (f(y) + f(z)) + \frac{h}{2} f'(z)z', \tag{8.7}$$

(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 (8.7) vanishes and we have

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

Differentiate (8.7) once more with respect to $h$ to obtain

$$z'' = \frac{1}{2} f'(z)z' + \frac{1}{2} f'(z)z' + \frac{h}{2} (f''(z)(z',z') + f'(z)z''), \tag{8.8}$$

for $h = 0$, the last term of (8.8) 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 $\Psi_h(y)$:

$$\Psi_h(y) = z(h) = z(0) + hz'(0) + \frac{h^2}{2} z''(0) + \ldots = y + hf + \frac{h^2}{2} f'f + \ldots \tag{8.9}$$

Comparing this with the expansion for the exact solution (8.6) we see that the first few terms are identical. Thus the local error vanishes to *at least* $\mathcal{O}(h^3)$. To get the next term in the local error expansion, we differentiate (8.8):

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

which evaluates to

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

which means that the expansions (8.9) and (8.6) *do not* match in the term of 3rd order: for trapezoidal rule, we have

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

### 8.1.3 Order conditions for Runge-Kutta methods

The methods discussed above were relatively simple in that they could be written without using the Runge-Kutta formalism. Nonetheless the same procedure can be used to determine order conditions in the general case; 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

$$F_i = f(Y_i), \tag{8.10}$$

$$Y_i = y + h\sum_{j=1}^{s} a_{ij} F_j. \tag{8.11}$$

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

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

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

$$\sum_{i=1}^{s} b_i = 1. \tag{8.12}$$

This is the first of what are termed the "order conditions". Any RK method satisfying (8.12) is consistent, and by Theorem 6.2.1, convergent independent of the $a_{ij}$. To get the next one, we compute the second derivative of $z$:

$$\begin{aligned} z'' &= \sum_{i=1}^{s} b_i F_i' + \sum_{i=1}^{s} b_i F_i' + h \sum_{i=1}^{s} b_i F_i'' \qquad (8.13) \\ &= 2\sum_{i=1}^{s} b_i F_i' + h \sum_{i=1}^{s} b_i F_i''. \qquad (8.14) \end{aligned}$$

From (8.10)-(8.11) we obtain

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

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

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

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

$$\sum_{i=1}^{s} b_i c_i = \frac{1}{2}. \tag{8.15}$$

For $p = 2$, we must therefore have *both* conditions (8.12),(8.15).

To continue this process, we would need to differentiate (8.13). 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

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

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,

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

which means that, for the method to have order $p = 3$, we must have *two* more conditions satisfied (in addition to (8.12) and (8.15)):

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

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 one 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. ($a_{ij} = 0$, $j \geq i$). Obviously, from the previous chapter, it is clear that an implicit $s$-stage method can reach the maximum order $p = 2s$.

## 8.2 Exercises

1. Compute the leading terms of the local error expansions for the methods (8.2) and (8.3).

2. Show that the "traditional Runge-Kutta" method (8.4) satisfies all the conditions mentioned for order 3, and, additionally, the four equations

$$\sum_{i=1}^{s} b_i c_i^3 = \frac{1}{4}, \tag{8.16}$$

$$\sum_{i=1}^{s}\sum_{j=1}^{s} b_i c_i a_{ij} c_j = \frac{1}{8}, \tag{8.17}$$

$$\sum_{i=1}^{s}\sum_{j=1}^{s} b_i a_{ij} c_j^2 = \frac{1}{12}, \tag{8.18}$$

$$\sum_{i=1}^{s}\sum_{j=1}^{s}\sum_{k=1}^{s} b_i a_{ij} a_{jk} c_k = \frac{1}{24}. \tag{8.19}$$

   which are the order 4 conditions.

3. Determine the order of the two-stage Gauss collocation method from Chapter 7.

4. Choose a Runge-Kutta method of order at least two and demonstrate the order by integrating the (nonlinear, nonscalar, smooth) initial value problem of your choice over a fixed interval with successively smaller step sizes until the predicted convergence rate is observed. (In other words, choose an interval $[t_0, t_0+T]$ and a number of time steps $N_k = 2^k N_0$, $k = 0, 1, \ldots$ for some $N_0 > 0$. For each $k$, choose a time step $h_k = T/N_k$ and compute the solution over $N_k$ time steps. Denote the solution at the end of the interval by $\bar{y}^{N_k}$. Approximate the error in the $k$th solution by comparing it with the $(k+1)$th, i.e. $\mathrm{err}_k = \|\bar{y}^{N_{k+1}} - \bar{y}^{N_k}\|$. Continue in this way until you observe the predicted convergence rate.)

# Chapter 9

# Equilibrium Points and Fixed Points

**Main concepts:** Equilibrium points, fixed points for RK methods, convergence of fixed points for one-step methods

Equilibrium points represent the simplest solutions to differential equations. In terms of the solution operator, they are the fixed points of the flow map. In this chapter we address the question of whether the equilibrium points of differential equation are retained as fixed points of the numerical method.

## 9.1 Equilibrium Points and Stability for Scalar ODEs

**Definition 9.1.1** An **equilibrium point** $x^*$ of the scalar differential equation $dx/dt = f(x)$ is a point for which $f(x^*) = 0$. □

It is a simple fact that the equilibria are themselves solutions of the ordinary differential equation. Assume $f$ is a $C'$ function and suppose all the equilibria $x_1 < x_2 < \ldots$ of

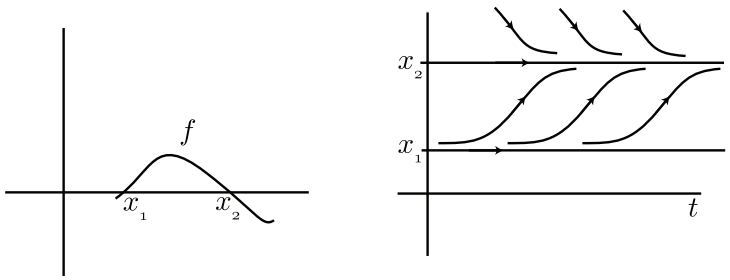



Figure 9.1: The shape of solutions near equilibrium points can be inferred from knowledge of $f$

an autonomous scalar equation $dx/dt = f(x)$ are known. Then, as we shall see in the

following chapter, we can often understand a great deal about the solutions of the system by evaluating the derivative (or derivatives) of $f$ at the equilibrium points. For example, if we know that $f'(x_1) > 0$, then we know that in a small neighborhood of $x_1$, the function $f$ is increasing. For $x$ a little below $x_1$, we must have

$$dx/dt = f(x) < 0$$

and

$$d^2x/dt^2 = f'(x)f(x) < 0$$

This tells us that a little below $x_1$, $x$ is decreasing and concave down. Similarly, a little above $x_1$ the graph of $x$ is increasing and concave up. If, moreover, $f'(x_2) < 0$, 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 9.1.

The arrows on the solution curves sketched in Figure 9.1 indicate the direction of time. Notice that curves tend away from $x_1$ and towards $x_2$. Loosely speaking, we say that an equilibrium point $x^*$ is *asymptotically stable* if, for all values $u$ sufficiently close to $x^*$, the solution through $u$ tends asymptotically to $x^*$ as $t \to \infty$. If instead nearby solutions tend away from the equilibrium point we say it is unstable. Thus $x_1$ in Figure 9.1 would correspond to an unstable equilibrium point and $x_2$ to an asymptotically stable one. We will define this concept more precisely in the next chapter.

Given an autonomous system (1.13) in $\mathbf{R}^d$, we say that a point $y^* \in \mathbf{R}^d$ is an *equilibrium point* if

$$f(y^*) = 0,$$

in which case the solution $y(t)$ with initial condition $y(t_0) = y^*$ satisfies $y(t) \equiv y^*$ (for all $t$).

An equilibrium point of the ODE 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 denote the set of fixed points of $\Phi_t$ 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 equilibrium points:

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

## 9.2 Fixed Points of Numerical Methods

For a numerical method with map $\Psi_h$, the fixed points may depend on $h$ as well as $f$. We denote the set of fixed points of $\Psi_h$ 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.

Consider first Euler's method (2.1)

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

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

Similarly, consider the implicit midpoint rule (8.5)

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

First, suppose $y_n$ and $y_{n+1}$ are such that $(y_{n+1} + y_n)/2$ is an equilibrium. The right hand side evaluates to zero and therefore (for positive $h$) $y_{n+1} = y_n$, i.e. $y_n$ is a fixed point of the method. Thus, $\mathcal{F} \subseteq \mathcal{F}_h$. Conversely, if $y_n$ is a fixed point of the implicit map $\Psi_h$, then the left side evaluates to zero, and we are left with $f(y_{n+1}) = f(y_n) = 0$, so again $\mathcal{F}_h \subseteq \mathcal{F}$ and the sets are equivalent.

Next, consider take 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, as we illustrate next. We apply this method to the logistic equation (1.4)

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

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, $F_h = \{0, 1, 1+h^{-1}, h^{-1}\}$. The fixed points $1+h^{-1}$ and $h^{-1}$, which are not equilibrium points, are termed *extraneous fixed points*. Note that the extraneous fixed points become very large as the time step is refined.

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 9.2.1** *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 an equilibrium point, 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 $\mathcal{F}_h$ includes all the fixed points of $\mathcal{F}$ but it may have some extraneous ones. Notice that the existence of extranous 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 extraneous 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 (8.4) is arguably the most popular method of all for dynamical studies, and yet you can check that it admits extraneous fixed points.

The reason that we can use methods like this is that in general (as in the example above), as $h \to 0$, all the extraneous fixed points tend to infinity. This gives us a simple way to test for extraneous fixed points: we simply vary the timestep and solve the problem repeatedly. 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.

To show this, we define the $\varepsilon$ neighborhoods of the equilibrium and fixed points as follows

$$\mathcal{F}(\varepsilon) := \{y \in \mathbb{R}^d : \|f(y)\| \leq \varepsilon\}, \qquad \mathcal{F}_h(\varepsilon) := \{y \in \mathbb{R}^d : \|\Psi_h(y) - y\| \leq h\epsilon\}.$$

In the following theorem, we assume that for an arbitrary bounded set $B$, there is a continuous interval of time steps $h \in (0, h^*)$, where $h^* > 0$ may depend on $B$, for which the numerical map $\Psi_h(y)$ exists for all $y \in B$.

**Theorem 9.2.2** *Let $\Psi_h$ be a consistent method for the the differential equation (1.13), and suppose for any bounded set $B \subset \mathbb{R}^d$ there exists $h^*(B) > 0$ such that $\Psi_h(B)$ exists for all $h \in (0, h^*)$. Then, for $h \in (0, h^*)$,*

$$\mathcal{F}_h(\varepsilon) \cup B \subseteq \mathcal{F}(\varepsilon + Kh). \tag{9.1}$$

*In particular,*

$$\mathcal{F}_h \cup B \subseteq \mathcal{F}(Kh). \tag{9.2}$$

**Proof** Given $y$ such that $\|\Psi_h(y) - y\| \leq h\varepsilon$, we wish to bound $\|f(y)\|$. Note that[1]

$$\|hf(y)\| \leq \|\Psi_h(y) - y\| + \|\Psi_h(y) - y - hf(y)\|.$$

The last term on the right is the difference between a step with $\Psi_h$ and one with Euler's method, which we denote by $\Psi_h^E$. Since by assumption $\Psi_h$ is a consistent method, we have

$$\|\Psi_h(y) - \Psi_h^E(y)\| = \|\Psi_h(y) - \Phi_h(y) + \Phi_h(y) - \Psi_h^E(y)\| \leq \|le(y,h)\| + \|le_E(y,h)\| \leq Kh^2.$$

for some positive $K$ dependent on $B$.

Inserting this into the previous relation we have

$$h\|f(y)\| \leq \|\Psi_h(y) - y\| + Kh^2,$$

and since $y \in \mathcal{F}_h(\varepsilon)$,

$$h\|f(y)\| \leq h\varepsilon + Kh^2,$$

or,

$$\|f(y)\| \leq \varepsilon + Kh.$$

This proves (9.1). Letting $\varepsilon \to 0$ gives (9.2). □

Let $y^*(h)$ denote a continuous branch of fixed points of $\Psi_h$ parameterized by $h \in (0, h^*)$. Either this set is unbounded as $h \to 0$, or there exists a set $B$ containing $y^*(h)$. In the latter case, the theorem shows that $y^*(h)$ converges to a point in $\mathcal{F}(\varepsilon)$.

[1]Triangle inequality $\|b - a\| \leq \|b\| + \|a\|$. Let $c = b - a$ to get $\|c\| \leq \|b\| + \|b - c\|$. Take $c = hf$ and $b = \Psi_h(y) - y$.

# Chapter 10

# Stability of Runge-Kutta Methods

**Main concepts:** Stability of equilibrium points, stability of maps, Runge-Kutta stability function, stability domain.

In the previous chapter we studied equilibrium points and their discrete couterpart, fixed points. A lot can be said about the qualitative behavior of dynamical systems by looking at the local solution behavior in the neighborhood of equilibrium points. In this chapter we study *stability* of these points and the related stability of fixed points of Runge-Kutta methods.

## 10.1 Stability of Equilibrium Points

We now define the stability of an equilibrium point. In general, the stability concerns the behavior of solutions near an equilibrium point in the long term. Given an autonomous system of differential equations (1.13), denote the solution of the system with the initial condition $y(0) = y_0$ by $y(t; y_0)$.

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

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

(2) The equilibrium point $y^*$ is *asymptotically stable* if 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) The equilibrium point $y^*$ is *unstable* if it is not stable (in the sense of Lyapunov).

Note: any asymptotically stable equilibrium point is also stable.

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

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

The origin is the only equilibrium point in each case. For the first system it is asymptotically stable (and also stable), for the second it is unstable, and for the third it is stable but not asymptotically stable. See Figure 10.1.

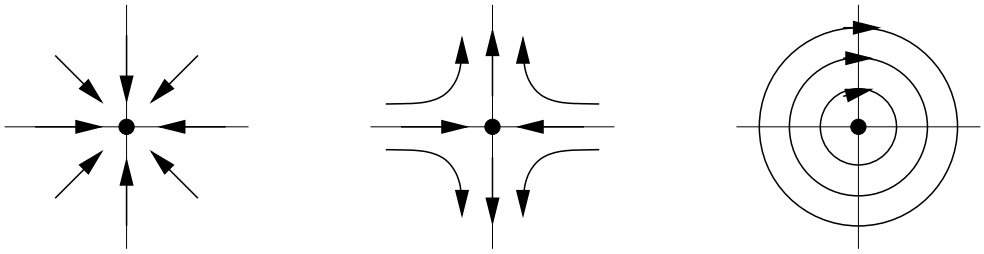


Figure 10.1: Some phase diagrams for linear systems in $\mathbb{R}^2$.

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 equilibrium point of interest is understood, or it may be a different usage of the term entirely. (Some caution is urged.)

There are a number of methods for analyzing stability in general. One of these is the Poincaré-Lyapunov Theorem and 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 10.1.1 (Linearization Theorem)** *Consider the equation in* $\mathbf{R}^d$*:*

$$dy/dt = Ay + F(y) \tag{10.1}$$

*subject to initial condition* $y(0) = y_0$*. Assume* $A$ *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* $y = 0$*, with* $F'(0) = F(0) = 0$*, where* $F'(y)$ *is the Jacobian matrix of* $F$*. Then* $y = 0$ *is an asymptotically stable equilibrium point. If* $A$ *has any eigenvalues with positive real part, then* $y = 0$ *is an unstable equilibrium point.*

Although the conditions on $A$ and $F$ may at first appear restricitve, Theorem 10.1.1 is actually quite powerful. First, if we have an equilibrium point $y^*$ of a differential equation (1.13) 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

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

which is in the form (1.13) but has an equilibrium point at $\tilde{y} = 0$.

Second, consider an arbitrary differential equation (1.13) with $C^2$ vector field and an equilibrium point at the origin. We may proceed as follows to use Theorem 10.1.1. First expand $f$ in a Taylor series expansion about 0:

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

Because 0 is an equilibrium point, we have $f(0) = 0$, and we set $A = f'(0)$. Define $F(y) = R(y)$. We see that we can write our equation in the form

$$\frac{dy}{dt} = Ay + F(y),$$

and it is easy to verify that $F(0) = F'(0) = 0$. If the eigenvalues of $A$ 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 10.1.2 (Linearization Theorem II)** *Suppose that $f$ in (1.13) is $C^2$ and has an equilibrium point $y^*$. If the eigenvalues of*

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

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

## 10.2 Stability of Fixed Points of Maps

We have seen in the introduction two versions of stability for equilibrium points of dynamical systems. These concepts have natural analogues for fixed points of maps.

**Definition 10.2.1** Consider a map $\Psi$ on $\mathbf{R}^d$ and a fixed point $y^*$: $y^* = \Psi(y^*)$. 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 $\forall \epsilon > 0$, $\exists \delta > 0, N > 0$ such that, if $\|y_0 - y^*\| < \delta$ then

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

(2) $y^*$ is *asymptotically stable* if 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 agree with the previous definitions for continuous dynamics. That is, let us take $\Psi \equiv \Phi_t$, the time-$t$ exact flow map of the differential equation. Then, if an equilibrium point of the continuous dynamics is stable, then it is also a stable fixed point for $\Psi$.

Next consider a more general map $\Psi$, not necessarily the flow map. What are the conditions for stability of a fixed point $y^*$? In other words: *when does the sequence of iterates obtained by successively applying $\Psi$ starting from a point near $y^*$ eventually converge to $y^*$?*

First consider a scalar linear iteration, $\Psi(y) = ay$, $a \in \mathbf{R}$, started from a point $y_0 \neq 0$. The iteration of $\Psi$ 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) asymptotical stability, (ii) stability, and (iii) instability.

Now consider a linear iteration in $\mathbf{R}^d$. Then we have $\Psi(y) = Ky$, $K \in \mathbf{R}^{d\times d}$. 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 $y_1 = Ky_0$, $y_2 = Ky_1 = K^2 y_0$, etc. The question 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 an enclosing all eigenvalues of $K$. We have a standard theorem of linear iterations to apply to this case:

**Theorem 10.2.1** *Let $z_n = \|K^n y_0\|$. Then (i) $z_n \to 0$, as $n \to \infty$, if and only if,*

$$\rho(K) < 1$$

*Moreover, (ii) $z_n \to \infty$ for some $y_0$, 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 use the Jordan canonical form for this purpose. Recall that an eigenvalue is semisimple if the dimension of its largest Jordan block is 1.

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

$$y_n = \Psi(y_{n-1}), \tag{10.2}$$

where $\Psi$ is a map on $\mathbf{R}^d$.

Note that

$$y^* = \Psi(y^*). \tag{10.3}$$

Take the difference of (10.2) and (10.3) to obtain

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

Next, we use Taylor's Theorem for functions from $\mathbf{R}^d$ to $\mathbf{R}^d$ 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 $\Psi$. $R(\Delta)$ 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 $\Psi'$. It is possible to show the following theorem:

**Theorem 10.2.2** *Given a smooth ($C^2$) map $\Psi$, (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.

## 10.3 Stability of Numerical Methods: Linear Case

For a linear system of ODEs

$$dy/dt = Ay,$$

where $A$ 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 $dy/dt = Ay$:

$$y_{n+1} = y_n + hAy_n = (I + hA)y_n.$$

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

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

Now, as $u_i$ is an eigenvector of $A$, we have

$$(I+hA)u_i = u_i + hAu_i = u_i + h\lambda_i u_i = (1+h\lambda_i)u_i$$

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 $\lambda$ of $A$, $h\lambda$ must lie inside a disk of radius 1 centred at $z = -1$ in the complex plane. This region is sketched below. We call it the *region of absolute stability* of Euler's method.

Thus, given the set of eigenvalues of $A$, 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 10.2.

### 10.3.1 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 $y_{n+1} = P(\lambda)y_n$, for some polynomial $P(x)$, when applied $y' = \lambda y$. 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 $\mu$, i.e. the ratio of two polynomials

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

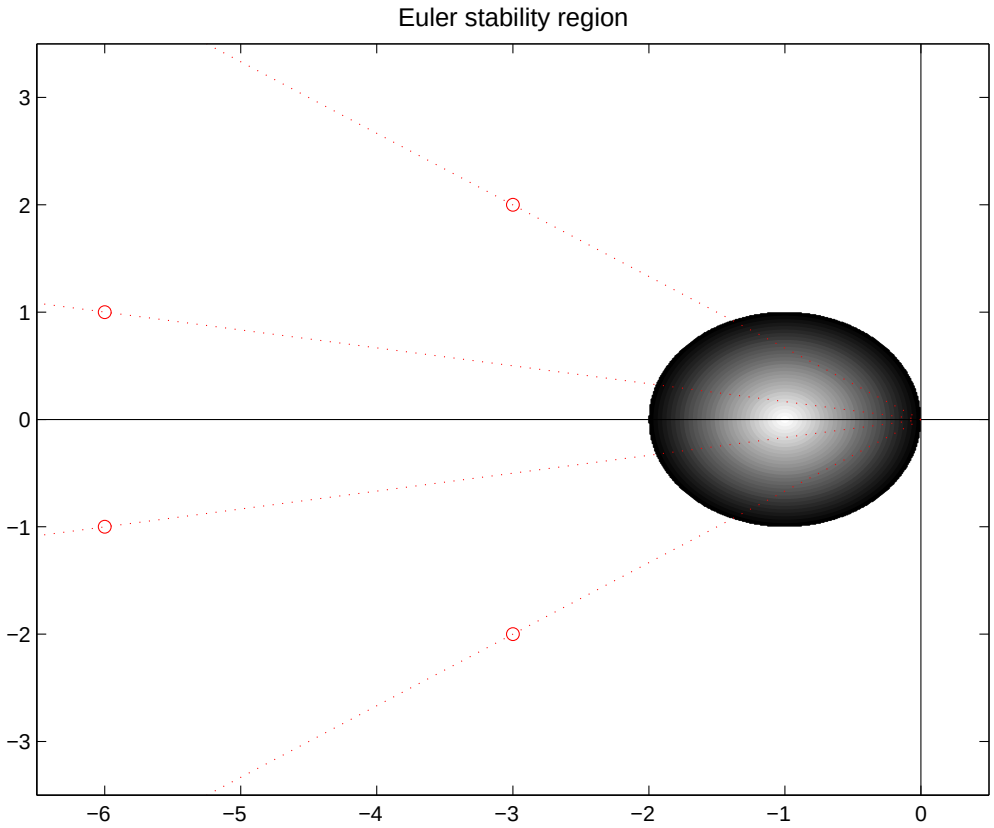



Figure 10.2: The spectrum of $A$ is scaled by $h$. Stability of the origin is recovered if $h\lambda$ is in the region of absolute stability $|1 + z| < 1$ in the complex plane.

where $P$ and $Q$ are two polynomials. $R$ 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 $R$:

$$y_{n+1} = R(\mu) y_n, \qquad R(\mu) = 1 + \mu b^T (I - \mu A)^{-1} \mathbf{1} \tag{10.4}$$

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

Next, Consider applying the general RK method to $y' = Ay$:

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

where

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

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 $A$. For simplicity, we write the eigenvectors as columns of a matrix $U$, so that $AU = U\Lambda$, where $\Lambda$ is the diagonal matrix of eigenvectors. So $U^{-1}AU = \Lambda$ is the diagonalization of $A$. 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

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

or

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

Now for the internal variables, we have

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

or

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

We may observe that, since $\Lambda$ 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

$$\frac{dz^{(i)}}{dt} = \lambda_i z^{(i)},$$

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 $dx/dt = \lambda x$. We state this in a theorem:

**Theorem 10.3.1** *Given the differential equation $y' = Ay$, where we suppose that the $d \times d$ matrix $A$ 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*

$$\frac{dy}{dt} = Ay,$$

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

$$\frac{dx}{dt} = \lambda x,$$

*where $\lambda$ is an eigenvalue of $A$.*

Since we know that a Runge-Kutta method applied to $\frac{dx}{dt} = \lambda x$ can be written

$$x_{n+1} = R(h\lambda)x_n$$

we have the following corollary:

**Corollary 10.3.1** *Consider a linear differential equation $dy/dt = Ay$ with diagonalizable matrix $A$. Let an RK method be given with stability function $R$. The origin is stable for the numerical method applied to $dy/dt = Ay$ (at stepsize $h$) if and only if*

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

*for all $\mu = h\lambda$, where $\lambda \in \sigma(A)$.*

### 10.3.2 Stability regions, A-stability and L-stability

Evidently the key issue for understanding the long-term dynamics of Runge-Kutta methods near fixed points concerns the region where $\hat{R}(\mu) = |R(\mu)| \leq 1$. This is what we call 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 $\hat{R}(\mu) \leq 1$. The condition

$$|1 + \mu| \leq 1$$

means $\mu$ lies inside of a disc of radius 1, centred at the point $-1$.

Trapezoidal rule: the stability region is the region where:

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

This occurs when

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

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*,

$$x_{n+1} = x_n + h\lambda x_{n+1}$$

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

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 10.3.

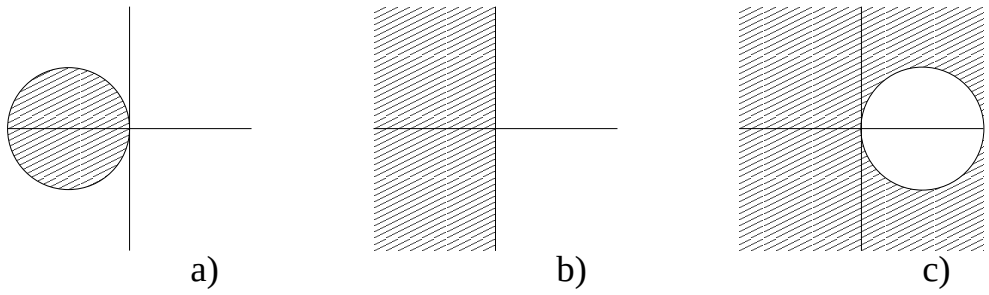



Figure 10.3: Stability Regions: (a) Euler's method, (b) trapezoidal rule, (c) implicit Euler

For the fourth-order Runge-Kutta method (8.4), 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; the latter gives the exact flow map for $dx/dt = \lambda x$. To graph $R$ we could use the following trick. For each value of $\mu$, $R$ is a complex number. The boundary of the stability region is the set of all $\mu$ 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 $\theta$ 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.

A second approach, more suited to MATLAB than Maple, is to just plot some level curves of $\hat{R}$ viewed as a function of $x$ and $y$, the real and imaginary parts of $\mu$. In fact, the single level curve $\hat{R} = 1$ is just precisely the boundary of the stability region! It is useful to have the nearby ones for a range of values near 1. In Matlab, we can achieve this as follows:

```
>> clear i;
>> [X,Y] = meshgrid(-5:0.01:5,-5:0.01:5);
>> Mu = X+i*Y;
>> R = 1 + Mu + .5*Mu.^2 + (1/6)*Mu.^3 + (1/24)*Mu.^4;
>> Rhat = abs(R);
>> contourplot(X,Y,Rhat,[1]);
```

Figure 10.4 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.

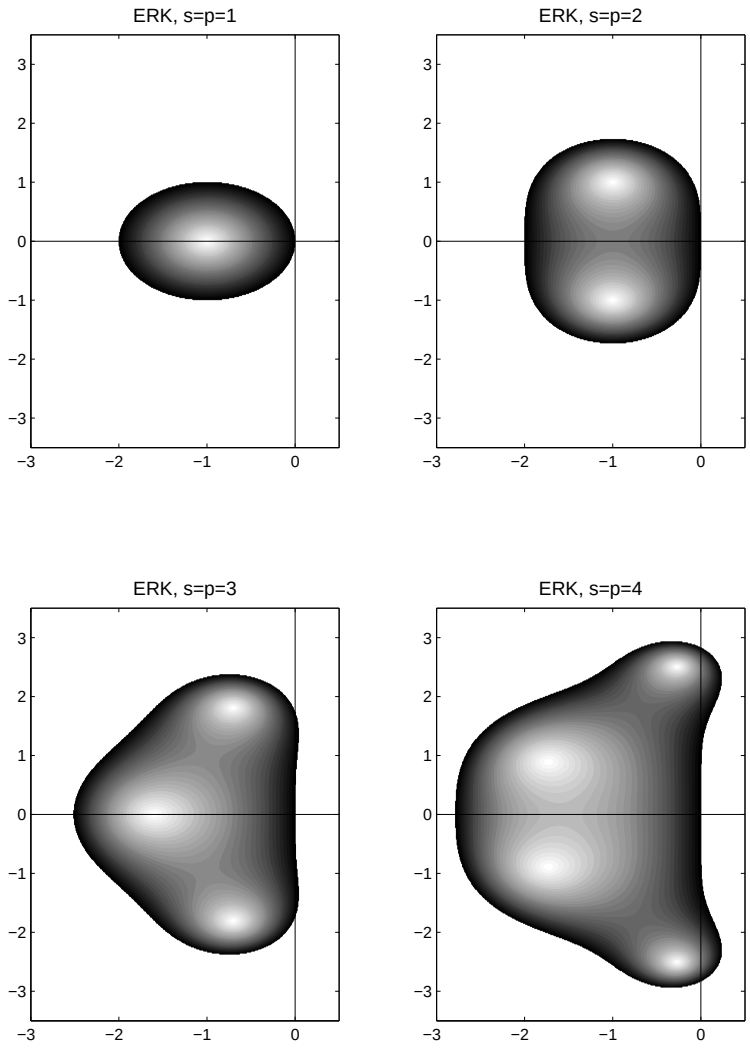



Figure 10.4: Explicit Runge-Kutta Stability Regions

What is the meaning of these funny diagrams? They tell us a huge amount. Consider first a scalar differential equation $dx/dt = \lambda x$, with possibly complex $\lambda$. We know that for the differential equation, the origin is stable for $\lambda$ 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 $\lambda$ 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 independent of $h$, even if $\lambda$ 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 $\lambda$ 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 $\lambda$ 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 $\lambda$ lies on the imaginary axis. Thus Euler's method is a very poor choice for integrating a problem like $dx/dt = i\omega x$, where $\omega$ is real. A very interesting question is whether the 4th order Runge-Kutta method has an unstable point at the origin for $\lambda$ on the imaginary axis and $h \to 0$. The operative question is: does the RK-4 stability region of 10.4 contain a segment on the imaginary axis containing 0? The answer is not obvious from the picture, although it certainly appears to be plausible.

The fact that a linear system can be decomposed into scalar linear problems (involving the eigenvalues as coefficients) means that we can apply the same approach to analyze the stability of a system by applying the scalar argument to each of the eigenvalues.

A very valuable feature if we are interested in preserving the asymptotic stability of the origin under discretization is 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 the origin is stable regardless of the stepsize.

Another important issue is whether the numerical method matches the asymptotic behavior as $\mu \to \infty$. If, in the scalar differential equation $dx/dt = \lambda x$, the coefficient $\lambda$ tends to $-\infty$, then we know that the solution will decay increasingly fast. We could ask that the numerical method have a similar property. We say that a method is L-stable if it has the properties (1) it is A-stable, and (2) $R(\mu) \to 0$, as $\mu \to \infty$.

## 10.4 Oscillatory Systems

An important special case for stability analysis is that of oscillatory linear systems, which have all their eigenvalues on the imaginary axis. For the moment, we will consider just a harmonic oscillator

$$\frac{dx}{dt} = u, \tag{10.5}$$

$$\frac{du}{dt} = -\omega^2 x. \tag{10.6}$$

To understand the asymptotic behavior of, for example, Runge-Kutta methods for the harmonic oscillator we need only compute the eigenvalues of this 2nd order problem (converting first to a 2-dimensional first order system) and we find that these are $\pm i\omega$. Thus

the behavior of an RK method is just determined by the approximation of the stability region to the imaginary axis.

However, it is also of interest to consider some methods constructed in alternate ways and for which the stability region must be defined differently. One such method is the *Störmer-Verlet* or *leapfrog* method which is applicable to problems of the form $dx/dt = u$; $du/dt = f(x)$:

$$x_{n+1} = x_n + hu_{n+1/2}, \tag{10.7}$$

$$u_{n+1/2} = u_n + \frac{h}{2} f(x_n), \tag{10.8}$$

$$u_{n+1} = u_{n+1/2} + \frac{h}{2} f(x_{n+1}). \tag{10.9}$$

We will therefore consider various numerical methods applied directly to (10.5)-(10.6) and analyze their asymptotic behavior directly.

Applying the leapfrog method to the harmonic oscillator results in the propagator

$$\Psi_h \left( \begin{bmatrix} x \\ u \end{bmatrix} \right) = \begin{bmatrix} 1 - \frac{h^2\omega^2}{2} & h \\ -h\omega^2 \left(1 - \frac{h^2\omega^2}{4}\right) & 1 - \frac{h^2\omega^2}{2} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \equiv M(\theta) \begin{bmatrix} x \\ u \end{bmatrix}, \quad \theta = h\omega. \tag{10.10}$$

The matrix $M(\theta)$ has eigenvalues

$$\lambda_\pm = 1 - \frac{1}{2}\theta^2 \pm \sqrt{\theta^2(\frac{1}{4}\theta^2 - 1)},$$

$\lambda_+\lambda_- = 1$ for all $\mu \geq 0$, hence $\det M(\theta)$ is equal to one. For $\theta^2 < 4$, the eigenvalues are complex, and

$$|\lambda_\pm|^2 = (1 - \frac{1}{2}\theta^2)^2 + \theta^2(1 - \frac{1}{4}\theta^2) = 1.$$

When $\theta^2 > 4$, both of the eigenvalues are real and one always has modulus greater than unity. This implies that, for the leapfrog method applied to the harmonic oscillator, the origin is a stable equilibrium point if

$$h^2\omega^2 < 4. \tag{10.11}$$

As $h$ increases from $h = 0$, the eigenvalues of the leapfrog method move around the unit circle until $h = 2/\omega$, at which point the eigenvalues are both at $-1$. As we continue to increase $h$, one eigenvalue tends toward the origin, while the other increases without bound along the negative real axis. The situation is diagrammed in Figure 10.5, below.

This discussion can be extended to a $2d$-dimensional linear systems of differential equations of the form

$$\frac{d}{dt} q = v, \tag{10.12}$$

$$\frac{d}{dt} v = -Kq. \tag{10.13}$$

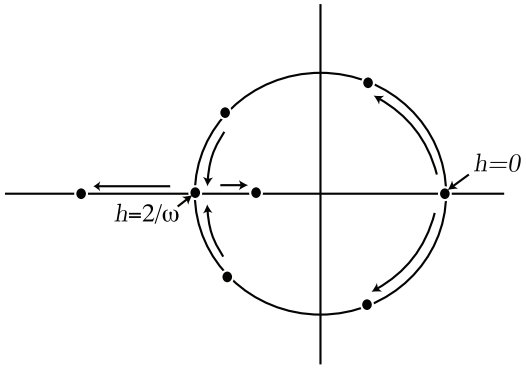



Figure 10.5: Eigenvalues of the leapfrog method for the harmonic oscillator: as $h$ increases, the complex conjugate pair moves around the unit circle, separating into two real eigenvalues at $h\omega = 2$.

Here $K$ is an $d \times d$ symmetric matrix. The stability of the equilibrium point at the origin in a linear system like this depends on the eigenvalue–eigenvector structure of the matrix $K$. When the dimension $d$ is large, there will be many eigenvalues ($2d$ of them, to be precise) but the stability of the origin under discretization is determined by the highest frequency term. Linear systems like (10.12)-(10.13) arise naturally through linearization of nonlinear systems of the form

$$M\frac{d^2x}{dt^2} = -\nabla V(x)$$

near equilibrium points.

## 10.5 Exercises

1. Use Theorem 10.1.1 to show that the pendulum equation with damping

$$\frac{d^2\theta}{dt^2} + \mu\frac{d\theta}{dt} + (g/L)\sin\theta = 0$$

has an asymptotically stable equilibrium at the origin in the phase plane. What does the theorem tell us about the undamped case $\mu \equiv 0$?

2. Consider the following system of differential equations expressing damped mechanical motion in a double-well potential:

$$\dot{q} = p, \qquad \dot{p} = -q(q^2 - 1) - 2p.$$

Determine the equilibrium points and their stability.

3. Compute the stability functions and sketch the stability regions for (a) Heun's Method (8.3), (b) the implicit midpoint method

$$y_{n+1} = y_n + hf((y_n + y_{n+1})/2),$$

(c) the RK method with Butcher diagram:

$$\begin{array}{c|cc} \frac{1}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 0 \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array},$$

Which of these three methods (a), (b), (c) are A-stable? L-stable?

4. Prove that an $A$-stable Runge-Kutta method is necessarily implicit.

5. Why does it not matter in which direction $\mu$ tends to $\infty$ in the definition of L-stability?

6. Show that the following family of Runge-Kutta methods all are second order ($\alpha > 0$ is a free parameter). Compute the stability function and graph the stability regions for $\alpha = 1/2, 3/4$.

$$\begin{array}{c|cc} 0 & & \\ (2\alpha)^{-1} & (2\alpha)^{-1} & \\ \hline & 1-\alpha & \alpha \end{array},$$

# Chapter 11

# Interlude: Linear Multistep Methods

**Main concepts:** Linear multistep methods, accuracy, root condition, stability.

In this chapter we introduce one of the two main classes of numerical integrators for dynamical systems: the linear multistep methods. The chapter contains an overview of the analytical issues relevant to these methods. In subsequent chapters, we will address the other major class: one step methods.

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. A linear $k$-step method is defined as

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f(y_{n+j}). \tag{11.1}$$

For a linear $k$-step method, we require that $\alpha_k \neq 0$ and either $\alpha_0 \neq 0$ or $\beta_0 \neq 0$. Furthermore, the coefficients in (11.1) 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 $\sum_j \beta_j = 1$.

When using (11.1) 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 $\beta_k$ 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 (11.1) can be applied. This is done, for example, by using forward Euler or another one-step method in the first $k - 1$ steps.

## 11.1 Examples

Some examples of linear multistep methods are:

- The $\theta$-method generalizes all linear one-step methods

$$y_{n+1} - y_n = h(1-\theta) f(y_n) + h\theta f(y_{n+1}). \tag{11.2}$$

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 $\beta_1 = 2$:
$$y_{n+2} - y_n = 2hf(y_{n+1}) \tag{11.3}$$

- 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 $f_n \equiv f(y_n)$):
$$y_{n+1} - y_n = hf_n \tag{11.4}$$
$$y_{n+2} - y_{n+1} = h\left(\frac{3}{2}f_{n+1} - \frac{1}{2}f_n\right) \tag{11.5}$$
$$y_{n+3} - y_{n+2} = h\left(\frac{23}{12}f_{n+2} - \frac{4}{3}f_{n+1} + \frac{5}{12}f_n\right) \tag{11.6}$$
*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
$$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = h\frac{2}{3}f(y_{n+2}). \tag{11.7}$$

## 11.2 Order of accuracy and convergence

Associated with the linear multistep method (11.1) are the polynomials

$$\rho(\zeta) = \sum_{j=0}^{k} \alpha_j \zeta^j, \qquad \sigma(\zeta) = \sum_{j=0}^{k} \beta_j \zeta^j. \tag{11.8}$$

These are important for understanding the dynamics of multistep methods, and will be used later.

The *residual* of a linear multistep method at time $t_{n+k}$ may be defined in a number of ways. We obtain it by substituting the exact solution $y(t)$ of (1.3) at times $y(t_{n+j})$, $j = 0, \ldots, k$ into (11.1), i.e.

$$r_n := \sum_{j=0}^{k} \alpha_j y(t_{n+j}) - h\sum_{j=0}^{k} \beta_j y'(t_{n+j}). \tag{11.9}$$

(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 maximal *order of accuracy* $p$ if $r_n = \mathcal{O}(h^{p+1})$ for all sufficiently smooth $f$.

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 $i$th derivative of $y(t)$. Substituting these into (11.9) and manipulating,

$$\begin{aligned} r_n &= \sum_{j=0}^{k} \alpha_j \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i)}(t_n) - h \sum_{j=0}^{k} \beta_j \sum_{i=0}^{\infty} \frac{(jh)^i}{i!} y^{(i+1)}(t_n) \\ &= \sum_{j=0}^{k} \alpha_j y(t_n) + \sum_{i=1}^{\infty} \sum_{j=0}^{k} \alpha_j \frac{(jh)^i}{i!} y^{(i)}(t_n) - \sum_{i=1}^{\infty} \sum_{j=0}^{k} \beta_j \frac{(jh)^i}{j(i-1)!} y^{(i)}(t_n) \\ &= \sum_{j=0}^{k} \alpha_j y(t_n) + \sum_{i=1}^{\infty} \frac{1}{i!} h^i y^{(i)}(t_n) \left[ \sum_{j=0}^{k} \alpha_j j^i - i \sum_{j=0}^{k} \beta_j j^{i-1} \right]. \end{aligned}$$

Equivalent conditions for a linear multistep method to have order of accuracy $p$ are:

- The coefficients $\alpha_j$ and $\beta_j$ satisfy (where $0^0 = 1$)

$$\sum_{j=0}^{k} \alpha_j = 0 \qquad \text{and} \qquad \sum_{j=0}^{k} \alpha_j j^i = i \sum_{j=0}^{k} \beta_j j^{i-1} \quad \text{for} \quad i = 1, \ldots, p. \tag{11.10}$$

- The polynomials $\rho(\zeta)$ and $\sigma(\zeta)$ satisfy

$$\rho(e^z) - z\sigma(e^z) = \mathcal{O}(z^{p+1}). \tag{11.11}$$

- The polynomials $\rho(\zeta)$ and $\sigma(\zeta)$ satisfy

$$\frac{\rho(z)}{\log z} - \sigma(z) = \mathcal{O}((z-1)^p). \tag{11.12}$$

The first of these follows from the considerations above. For proofs of the second and third forms, see Hairer, Nørsett and Wanner (1993).
**Examples.** The method (11.3) has order 2. The methods (11.5) and (11.6) have orders 2 and 3, respectively. The method (11.7) has order 2.

### 11.2.1 The root condition, a counter-example

Earlier we saw that for Euler's method, convergence follows from the fact that the residual is $\mathcal{O}(h^2)$, leading to a global error of $\mathcal{O}(h)$ on a fixed interval. For linear multistep methods, first order accuracy is insufficient to ensure convergence. We will not prove convergence

for this class of methods, but will simply state the convergence theorem and show where it can go wrong.

The method (11.1) is said to satisfy the *root condition*, if all roots $\zeta$ of

$$\rho(\zeta) = 0,$$

lie on the unit disc ($|\zeta| \leq 1$), and any root of modulus one ($|\zeta| = 1$) has multiplicity one.

Furthermore, a linear multistep method is incomplete without a starting procedure to generate the first $k-1$ iterates $y_1, \ldots, y_{k-1}$.

**Theorem 11.2.1** *Suppose a linear multistep method (11.1) 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 converges to the exact solution of (1.3) on a fixed interval as* $h \to 0$ *if and only if it has order of accuracy* $p \geq 1$ *and satisfies the root condition.*

The proof of this theorem will not be handled in these notes. See, e.g., the monograph of Hairer, Nørsett, & Wanner (1993).

To illustrate the necessity of the root condition, consider the method

$$y_{n+3} + y_{n+2} - y_{n+1} - y_n = h\left(\frac{8}{3}f(y_{n+2}) + \frac{2}{3}f(y_{n+1}) + \frac{2}{3}f(y_n)\right). \tag{11.13}$$

Substituting $\rho(\zeta) = \zeta^3 + \zeta^2 - \zeta - 1$ and $\sigma(\zeta) = \frac{8}{3}\zeta^2 + \frac{2}{3}\zeta + \frac{2}{3}$ into (11.11) gives

$$\rho(e^z) - z\sigma(e^z) = \frac{1}{3}z^4 + \mathcal{O}(z^5),$$

so the method is third order accurate. 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

$$y_{n+3} + y_{n+2} - y_{n+1} - y_n = 0. \tag{11.14}$$

If the roots $\zeta_1$, $\zeta_2$ and $\zeta_3$ 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 $\zeta_i$ 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 (11.14) becomes

$$y_n = c_1\zeta_1^n + c_2\zeta_2^n + c_3 n\zeta_2^n.$$

One still has $|y_n| \to \infty$ (but with linear growth), unless $c_3 = 0$.

The constants $c_1$, $c_2$ and $c_3$ are determined by the initial conditions 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 conditions are perturbed slightly. We take $y_0 = 1+\varepsilon$, $y_1 = y_2 = 1$. Then we find

$$c_1 = 1 + \frac{3}{4}\varepsilon, \quad c_2 = \frac{1}{4}\varepsilon, \quad c_3 = -\frac{1}{2}\varepsilon,$$

and the solution is unbounded. The numerical solution sequence is unstable to perturbations in the initial conditions. As a consequence, any errors incurred will destabilize the solution. The method works only for the trivial differential equation, $y' = 0$, and then only if the starting procedure is exact. It fails to converge for all other differential equations.

As an example we compute the solution of $y' = -y$, $y(0) = 1$, $t \in [0, 1.5]$ using (11.13) for $h = 1/10$, $1/100$ and $1/1000$. The instability gets *worse* as $h$ decreases.

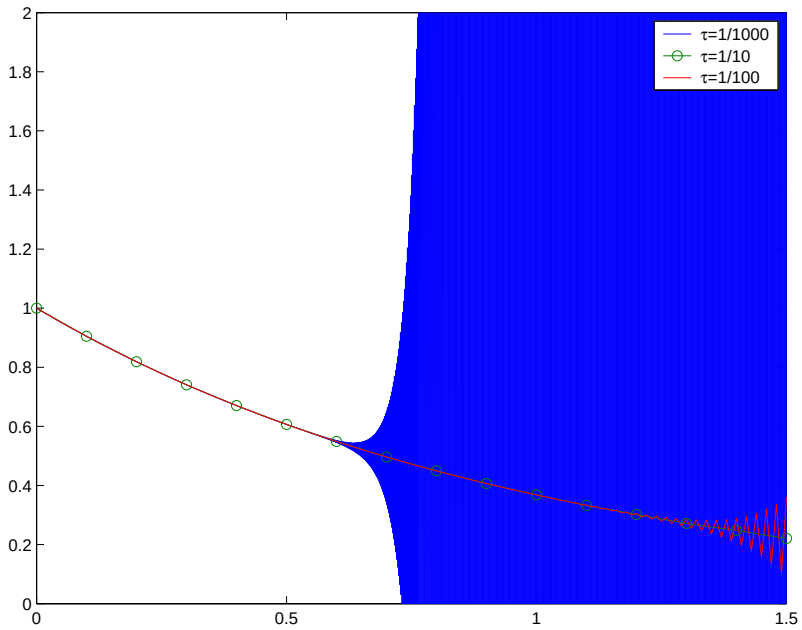



Figure 11.1: Solution of $y' = -y$, $y(0) = 1$ with (11.13) for various $h$.

**Theorem 11.2.2** *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 $k$.*

## 11.3 Stability

As we will see later in the course, an important criterion for distinguishing between different methods is their ability to preserve the stability of a stable equilibrium. To test this, we check under what conditions the numerical solution converges to zero when we apply (11.1) to the scalar linear test problem $y' = \lambda y$:

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h\lambda \sum_{j=0}^{k} \beta_j y_{n+j}.$$

Letting $z = h\lambda$ we write

$$\sum_{j=0}^{k} (\alpha_j - z\beta_j) y_{n+j} = 0.$$

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 *stability region* $\mathcal{S}$ of a linear multistep method is the set of all points $z \in \mathbb{C}$ such that all roots $\zeta$ of the polynomial equation $\rho(\zeta) - z\sigma(\zeta) = 0$ lie on the unit disc $|\zeta| \leq 1$, and those with modulus one are simple.

On the boundary of the stability region $\mathcal{S}$, precisely one root has modulus one, say $\zeta = e^{\mathrm{i}\theta}$. Therefore an explicit representation for the boundary of $\mathcal{S}$ is easily derived:

$$\partial\mathcal{S} = \left\{ z = \frac{\rho(e^{\mathrm{i}\theta})}{\sigma(e^{\mathrm{i}\theta})},\ \theta[-\pi, \pi] \right\}.$$

Figure 11.2 shows plots of the stability regions for the Adams-Bashforth methods of orders $p = 1$, 2 and 3.

A linear multistep method is called *A-stable* (or *unconditionally stable*) if the stability domain $\mathcal{S}$ contains the entire left half-plane

$$\{z \in \mathbb{C} : \operatorname{Re} z \leq 0\} \subset \mathcal{S}.$$

**Theorem 11.3.1** *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.

For stiff problems in which the stiff components have eigenvalues near the real axis, A-stability is too strong a requirement. Instead a weaker concept is introduced:

The linear multistep method (11.1) is A($\alpha$)-stable, for $\alpha \in (0, \pi/2)$, if the stability domain contains a wedge in the left half-plane:

$$\{z \in \mathbb{C} : |\arg(z) - \pi| < \alpha\} \subset \mathcal{S}.$$

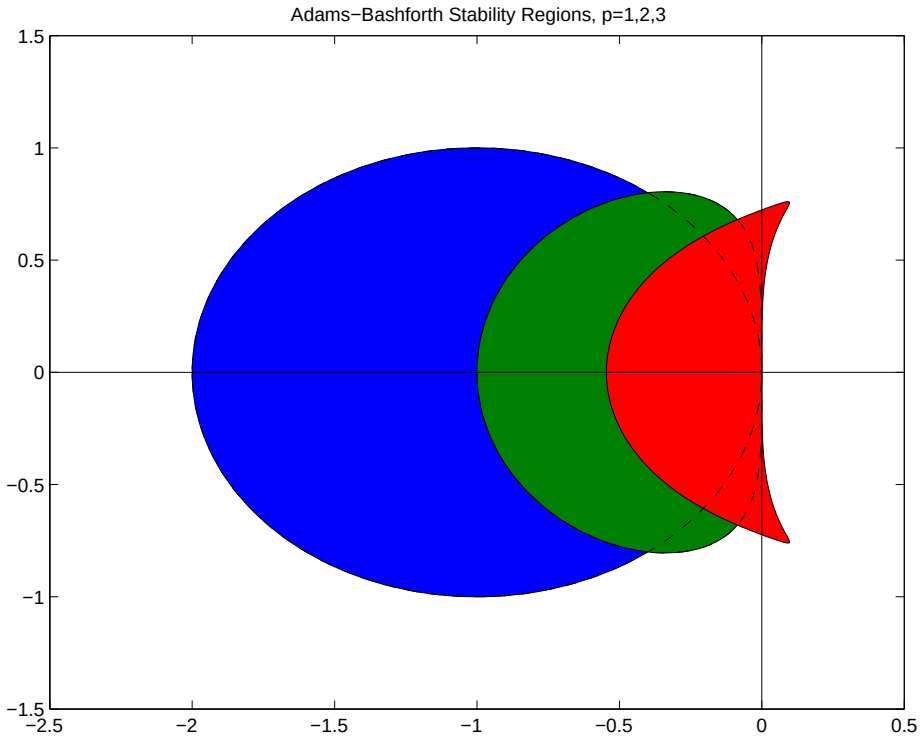



Figure 11.2: Stability regions of the Adams-Bashforth methods of orders $p = 1$ (blue), 2 (green) and 3 (red).

The important point is that for $\lambda$ lying within the wedge of stability, the method is unconditionally stable (norm-nonincreasing for any $h$).

Figure 11.3 shows plots of the stability regions for the Backward differentiation formulae (BDF) of orders $p = 1, \ldots, 6$. The first and second order methods are A-stable. The rest are A($\alpha$)-stable with $\alpha$ (approximately):

| Order, $p$ | $\alpha$ |
|---|---|
| 3 | 86.03° |
| 4 | 73.35° |
| 5 | 51.84° |
| 6 | 17.84° |

## 11.4 Implementation issues

### 11.4.1 Starting process.

As mentioned briefly in section 11.2, a linear $k$-step method needs a starting procedure to generate the first $k-1$ solution values. This is typically done using either $k-1$ steps of a

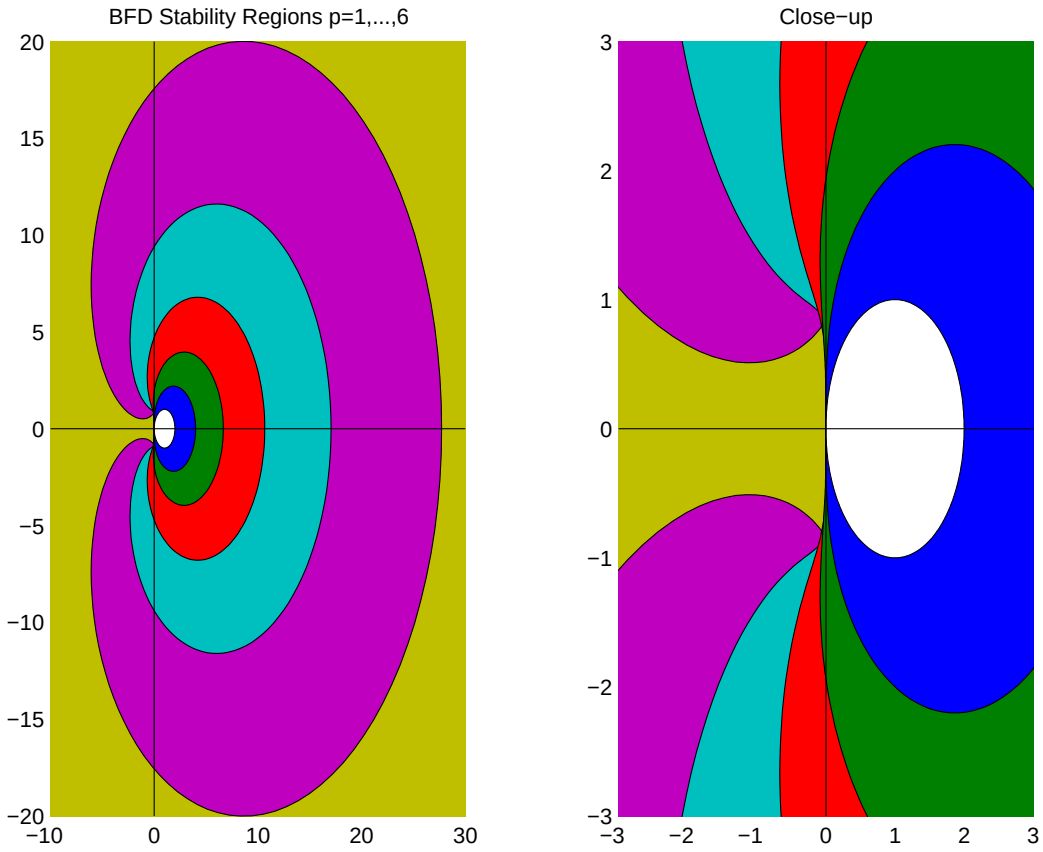



Figure 11.3: Stability regions of the BDF methods of orders $p = 1, \ldots, 6$. For BDF-1 (blue) $\mathcal{S}$ is everything outside the white region; for BDF-2 (green) it is everything outside the blue region; etc.

one-step (Runge-Kutta) method, or applying successively $\ell$-step methods with $\ell = 1, 2, \ldots$. For example, to start a 4-step Adams-Bashforth scheme, the methods (11.4), (11.5), (11.6) could be applied in succession, for $n = 0$. For problems with discontinuous $f$, it is necessary to apply the starting procedure after each discontinuity.

### 11.4.2 Variable stepsize implementation.

The order conditions (11.10)–(11.12) only hold for constant stepsize $h$. If the stepsize varies over the interval $[t_n, t_{n+k}]$, i.e. $h_j = t_{n+j+1} - t_{n+j}$, $j = 0, \ldots, k-1$, with distinct $h_j$, then the coefficients for a given order become dependent on the $h_j$. In general the stepsize ratio $q_j = h_{j+1}/h_j$ will be restricted to some range $0 < q_{\min} < q_j < q_{\max}$, by the root condition. There are various techniques for implementing variable stepsize multistep methods. For more information, see Hairer, Nørsett & Wanner (1993).

In most cases it is not desirable to store all solutions $y_0, \ldots, y_N$, as this eats up computer

memory. Instead only the minimum number of 'back-values' necessary for the implementation is kept. The solution is written to disk as often as desired. For this reason, variable stepsize implementations of multistep methods are problematic. Either the coefficients have to be modified to suit the local stepsize schedule, or the past solution values have to be interpolated onto a uniform grid.

An exception is a variable stepsize implementation in which the stepsize is allowed to change only by a factor of 2. In this case, provided the stepsize is not allowed to double too often, one can get by with storing the $2k$ vectors $y_{n-k}, \ldots, y_{n+k-1}$. When the stepsize is doubled, the values $y_{n+k-2}, y_{n+k-4}, \ldots$ are used in formula (11.1) with $h$ replaced by $2h$. When the stepsize is halved, a $p$th order interpolation can be used to obtain the vectors $y_{n+k-1/2}, y_{n+k-3/2}, \ldots$ needed to apply (11.1) with a stepsize of $h/2$.

### 11.4.3 Error estimation.

To estimate the error for the purpose of adaptivity, one technique makes use of a second linear multistep method, also of order $p$ (but perhaps not a $k$-step method). Denote the coefficients of this process with $\tilde{\alpha}_j$ and $\tilde{\beta}_j$. According to formula (11.11), these methods must satisfy, assuming the values $y_n, \ldots, y_{n+k-1}$ are exact,

$$y(t_{n+k}) - y_{n+k} = ch^{p+1}y^{(p+1)}(t_{n+k}) + \mathcal{O}(h^{p+2}),$$
$$y(t_{n+k}) - \tilde{y}_{n+k} = \tilde{c}h^{p+1}y^{(p+1)}(t_{n+k}) + \mathcal{O}(h^{p+2}),$$

for appropriate $c, \tilde{c} > 0$. Assuming we have chosen the second linear multistep method such that $c \neq \tilde{c}$, we can combine the above and eliminate $y^{(p+1)}(t_{n+k})$ to get the local error estimate

$$y(t_{n+k}) - y_{n+k} = \frac{c}{c - \tilde{c}}(\tilde{y}_{n+k} - y_{n+k}) + \mathcal{O}(h^{p+2}).$$

For efficiency, it is best to use an explicit method such as an Adams-Bashforth for the error estimator $\tilde{y}_{n+k}$. Since this value is thrown away after estimating the error, it cannot alter stability to use an explicit method.

### 11.4.4 Nonlinear systems.

For implicit methods, the nonlinear system to be solved in each timestep is

$$\alpha_k y_{n+k} = h\beta_k f(y_{n+k}) + \left[\sum_{j=0}^{k-1} -\alpha_j y_{n+j} + h\beta_j f_{n+j}\right],$$

where the terms in brackets are explicitly known at time step $(n + k - 1)$. This system has the same form as discussed Appendix A and is of dimension $d$. Furthermore, the back values $y_n, \ldots y_{n+k-1}$ can be used with a high order extrapolation formula to generate a good initial guess for (modified) Newton iteration, so the nonlinear solves are fairly efficient.

## 11.5 Exercises

1. Determine the stability region $\mathcal{S}$ for the Leapfrog method

$$y_{n+1} = y_{n-1} + 2hf(t_n, y_n).$$

2. Derive a 4th order Adams-Bashforth method using one of (11.10)–(11.12). Does it satisfy the root condition? Determine its stability region.

3. Consider the Lorenz equations:

$$\begin{aligned} y_1' &= \sigma(y_2 - y_1) \\ y_2' &= ry_1 - y_2 - y_1y_3 \\ y_3' &= y_1y_2 - by_3 \end{aligned}$$

Choose the parameters to be $\sigma = 10$, $b = 8/3$, $r = 28$ and the initial condition $y_0 = (0, 1, 0)^T$. Integrate this system on the interval $t \in [0, 100]$. Use the 4th order Adams-Bashforth method you derived in the previous problem. Start the integration with 3 steps of Euler's method.

   (a) Plot the solution in 3D using the built-in matlab function `plot3`. Use the rotation tool in the figure window to observe the Lorenz attractor.

   (b) Experiment with different stepsizes. Can you observe 4th order convergence? If not, try integrating over a shorter interval, (say $t \in [0, 10]$). **Hint:** It may be helpful to plot the time history of one component (say $y_2(t)$) of the solution, for several different stepsizes, on the same axes.

   (c) Repeat the computation using forward Euler on the whole interval $t \in [0, 100]$. Do you see any advantage to using the fourth order method for solving this problem on a long time interval?

# Chapter 12

# Reversibility and Composition Methods

**Main concepts:** $\rho$-reversibility, adjoint of a method, self-adjoint methods, composition methods

Many physical processes—e.g. those on the molecular scale and on the cellestial scale—are *reversible*, a symmetry property of the flow map $\Phi_t$. It turns out to be useful in such a case, to use a numerical method whose discrete flow map is also reversible. One reason for doing so is the simple construction of higher order methods by composition, which will be treated in this chapter. Another reason is the near preservation of invariants in integrable problems by reversible methods, which will not be treated in these notes.

## 12.1 $\rho$-Reversibility

Let $\rho$ be a linear, invertible transformation on $\mathbb{R}^d$. The autonomous ODE (1.13) is called *$\rho$-reversible* if

$$\rho f(y) = -f(\rho y) \tag{12.1}$$

for all $y$.

For a $\rho$-reversible ODE, the solution operator $\Phi_t$ satisfies

$$\rho \circ \Phi_t = \Phi_{-t} \circ \rho. \tag{12.2}$$

In words, evolving the system $t$ units in time and applying the reversal transformation, is equivalent to applying the reversal transformation and then evolving the system $t$ units *backward* in time.

As a consequence of (12.2), and since $\Phi_{-t} = \Phi_t^{-1}$,

$$\rho^{-1} \circ \Phi_t \circ \rho \circ \Phi_t = \mathrm{id},$$

the identity map. So if one evolves the system $t$ units in time, then applies the reversal transformation, then evolves it $t$ more units in time and again applies the inverse reversal

transformation, the system is returned to its initial state. This situation is illustrated in Figure 12.1.

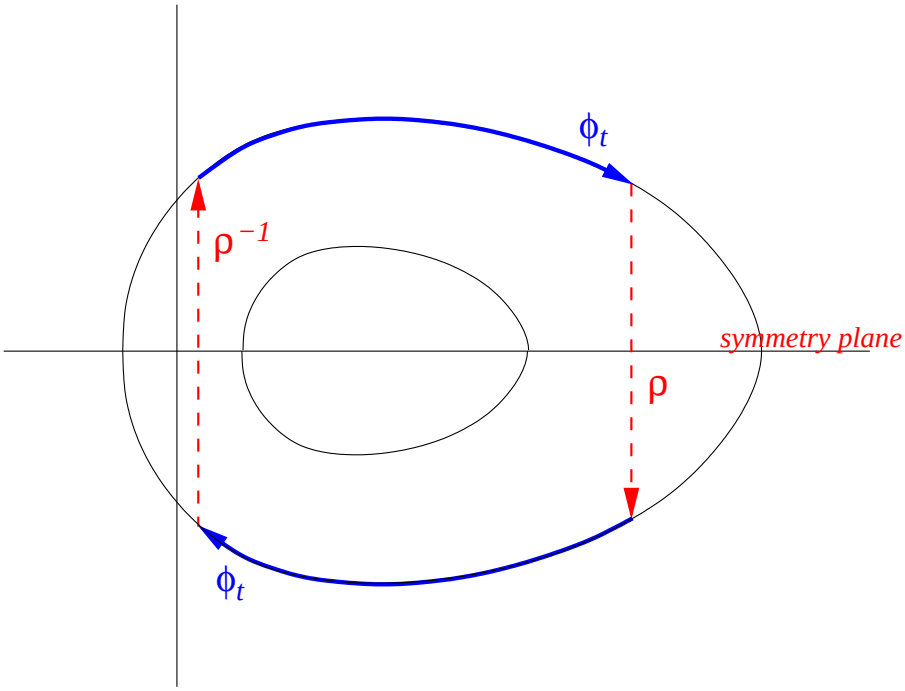



Figure 12.1: Illustration of a $\rho$-reversal symmetry. Here, $\rho$ is a simple reflection about the horizontal axis. The black lines represent phase trajectories of the dynamical system in question.

**Example.** Mechanical systems. A large class of reversible problems are those given by Newton's second law of motion (5.2)

$$M\ddot{q} = F(q), \tag{12.3}$$

where $q \in \mathbb{R}^d$ is a position vector, $F : \mathbb{R}^d \to \mathbb{R}^d$ is a vector of forces, and $M$ is a symmetric, positive definite (mass) matrix. Introducing the momentum $p \equiv M\dot{q}$, this becomes

$$\dot{q} = M^{-1}p, \qquad \dot{p} = F(q).$$

This system can be written as a first order system (1.3) of dimension $2d$:

$$y = \begin{pmatrix} q \\ p \end{pmatrix}, \qquad f(y) = \begin{pmatrix} M^{-1}p \\ F(q) \end{pmatrix}$$

The time-reversal transformation just negates the momentum:

$$\rho = \begin{bmatrix} I & \\ & -I \end{bmatrix}.$$

One can easily check that (12.1) holds. Evolving a mechanical system $t$ units in time, reversing its momentum, evolving it again $t$ units in time, and again reversing its momentum, returns it to its initial state.

## 12.2 Adjoint of a Method

The **adjoint** of a one-step method $\Psi_h$ such as a Runge-Kutta method, which takes the solution of (1.3) from $t_n$ to $t_{n+1}$, i.e.

$$y_{n+1} = \Psi_h(y_n),$$

is the method $\Psi_h^* := \Psi_{-h}^{-1}$, i.e. the inverse of the method with negative time step. We can derive this method by setting $y_{n+1}^* = y_n$ with $y_n^* = y_{n+1}$ and stepping backward with stepsize $-h$,

$$y_n^* = \Psi_{-h}(y_{n+1}^*).$$

Solving for $y_{n+1}^*$:

$$y_{n+1}^* = \Psi_{-h}^{-1}(y_n^*) = \Psi_h^*(y_n^*).$$

**Example.** The adjoint of Euler's method is

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

Rearranging gives:

$$y_{n+1}^* = y_n^* + hf(y_{n+1}^*),$$

an implicit method known as *backward Euler* or *implicit Euler*.

For a general $s$-stage Runge Kutta method with coefficients $A = (a_{ij})$ and $b = (b_i)$, we can calculate

$$y_n^* = y_{n+1}^* - h\sum_{j=1}^{s} b_j f(Y_j),$$

$$Y_i = y_{n+1}^* - h\sum_{j=1}^{s} a_{ij} f(Y_j),$$

or,

$$y_{n+1}^* = y_n^* + h\sum_{j=1}^{s} b_j f(Y_j),$$

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

However, the abscissae $c_i = \sum_j a_{ij}$ have become reversed ($c_i^* = 1 - c_i$), so we replace $Y_i$ by $Y_{s+1-i}$ (to recover the chronological ordering of the stage vectors) and obtain the coefficients $a_{ij}^*$ and $b_i^*$ of the adjoint method:

$$a_{ij}^* = b_{s+1-j} - a_{s+1-i,s+1-j}, \quad b_i^* = b_{s+1-i}.$$

This is again an $s$-stage Runge-Kutta method. The adjoint of an explicit Runge-Kutta method is always implicit.

The order of the adjoint of method is the same as the order of the original method. In fact, the truncation error of the adjoint method satisfies

$$le^*(y,h) = (-1)^p le(y,h),$$

where $p$ is the order of the method (i.e. $le(y,h) = \mathcal{O}(h^{p+1})$.)

A method whose adjoint is equivalent to itself ($\Psi_h^* = \Psi_h$) is called *self-adjoint*. The implicit midpoint rule (8.5) is self-adjoint. For a self-adjoint Runge-Kutta method, the coefficients satisfy

$$a_{s+1-i,s+1-j} + a_{ij} = b_j$$

for all $i, j$. Self-adjoint methods are necessarily implicit and have even order.

A numerical method is called *symmetric* or *time-reversible* if it satisfies $\Psi_h^{-1} = \Psi_{-h}$. Therefore, self-adjoint methods are time-reversible. More generally, if a numerical method applied to a $\rho$-reversible system satisfying (12.1) satisfies

$$\rho \circ \Psi_h = \Psi_{-h} \circ \rho$$

(and mostly they do), then the method is $\rho$-reversible if and only if $\Psi_h$ is symmetric.

## 12.3 Composition Methods

Recall that the group property of flow maps $\Phi_t \circ \Phi_s = \Phi_{s+t}$ does not hold in general for numerical methods. It turns out we can use this to our "advantage" as follows:

Given a one-step method $\Psi_h$ and a set of $s$ real numbers $\gamma_i$, $i = 1, \ldots, s$ that sum to one $\sum_{i=1}^s \gamma_i = 1$, the method obtained by composing $s$ steps of size $\gamma_i h$ is called a *composition method*

$$\bar{\Psi}_h = \Psi_{\gamma_s h} \circ \cdots \circ \Psi_{\gamma_1 h}$$

If $\Psi_h$ is of order $p$, then if

$$\gamma_1^{p+1} + \cdots + \gamma_s^{p+1} = 0,$$

the composition method is at least of order $p+1$.

This relation has no real solution for $p$ odd. For even $p$, there is a solution for $s = 3$. A popular choice is

$$\gamma_1 = \gamma_3 = \frac{1}{2 - 2^{1/(p+1)}}, \quad \gamma_2 = -\frac{2^{1/(p+1)}}{2 - 2^{1/(p+1)}}, \tag{12.4}$$

which is of order $p+1$. If $\Psi_h$ is a symmetric method, then this composition is also symmetric, and the resulting method is actually of order $p+2$.

The approach can be applied recursively. For example, if $\Psi_h$ is of order 2 and symmetric, then the $\bar{\Psi}_h$ is of order 4 and symmetric. Applying the method recursively ($3^2 = 9$ steps of $\Psi_h$) we obtain order 6. Applying it recursively again ($3^3 = 27$ steps of the original method) we obtain order 8. And so on.

However, other, possibly more efficient, compositions may be constructed directly (without using this recursion), and the above is really only intended to illustrate the possibilities.

Instead of composing the same methods with different stepsizes, we could also imagine composing different methods. The composition of two steps of equal size with a method $\Psi_h$ and its adjoint $\Psi_h^*$, yields a self-adjoint method since

$$(\Psi_h \circ \Psi_h^*)^* = \Psi_h \circ \Psi_h^*.$$

For example, the composition of a half-step of backward Euler with a half-step of forward Euler gives either the implicit midpoint rule (8.5) or the trapezoidal rule (2.2) depending on the order of composition. Because the Euler methods are first order, the composition must also be at least first order. But since it is symmetric, it must have even order, so we conclude that it must be at least second order.

Note that (12.4) has $\gamma_2$ negative. This means that the method must regularly take backward steps. For some problems backward steps are ill-defined. For this reason, one would like to avoid them. Unfortunately it has been shown that to obtain order greater than 2, backward steps are unavoidable. For this reason, higher order composition methods are almost exclusively used only with reversible problems.

## 12.4 Exercises

1. Find a $\rho$-reversible symmetry for the Lotka-Volterra model (1.6) in the case $r = 1$.

2. Construct the adjoint of Heun's method (8.3).

3. Show that a self-adjoint Runge-Kutta method must have $|R(iy)| = 1$ on the imaginary axis. Use analyticity of the stability function to prove that a symmetric RK method with Butcher matrix $A$ whose eigenspectrum lies strictly in the positive half of the complex plane is A-stable.

4. Discuss the relationship between a (long) numerical solution computed with implicit midpoint, and one computed with trapezoidal rule.

# Chapter 13

# Splitting Methods

**Main concepts:** Linear systems, nonlinear systems, BCH theorem, order conditions

In this section, we present a natural approach to "method building" which is based on splitting a vector field into integrable parts. This approach is remarkably general and can be used to construct methods with properties very similar to those of the flow map of the dynamical system.

## 13.1 Introduction to Splitting Methods

The idea is simple: *divide et impera* (divide and conquer). It is probably the most important systematic technique for the development of algorithms in general. Suppose we wish to solve the differential equation

$$dz/dt = f(z) \equiv f_1(z) + f_2(z),$$

where each of two differential equations

$$dz/dt = f_1(z)$$

and

$$dz/dt = f_2(z)$$

happen to be completely integrable (i.e. analytically solvable). [Note that $dz/dt = f(z)$ is *not* solvable just because the two pieces are! ]

The way this is done is diagrammed in Figure 13.1, below. We first note that at any point in phase space, the vector field (hence the tangent vector to the solution curve) can be broken up into the two components $f_1$ and $f_2$. A method can then be constructed by stepping forward along first one, then a second, solution curve, each time for a small timestep $h$.

That is, we start from some point $z_n$ and solve first the initial value problem

$$dz/dt = f_1(z), \qquad z(0) = z_n$$

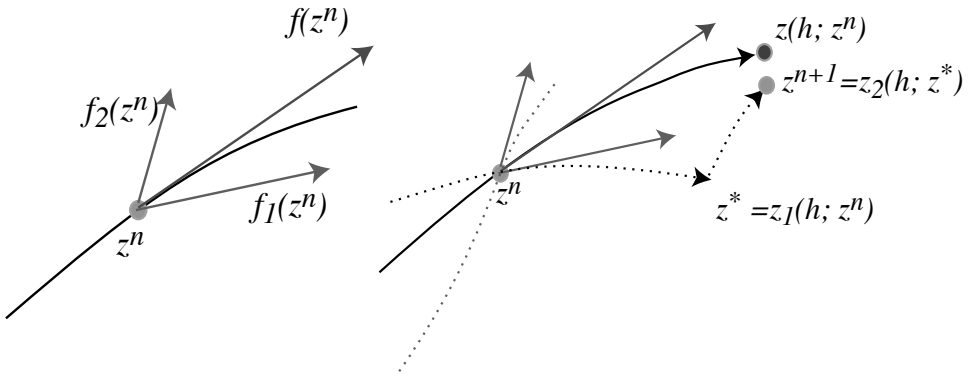



Figure 13.1: Divide and Conquer: An approximation to the solution is constructed by following the solutions of the two parts of the vector field

for a time $h$. This means to compute the exact solution of this part of the system, which is possible, since $f_1$ is assumed integrable. This takes us to a point $z^* = z_1(h; z_n)$, where $z_1$ represents the exact solution of the differential equations on vector field $f_1$.

Next we start from $z^*$ and solve the second differential equation initial value problem

$$dz/dt = f_2(z), \qquad z(0) = z^*$$

for a time $h$. This takes us to the point $z_{n+1} = z_2(h; z^*)$. Denoting the flow maps of the vector fields $f_1$ and $f_2$ as $\Phi_{t,f_1}$ and $\Phi_{t,f_2}$, we have just computed $z_{n+1} = \Phi_{h,f_2}(\Phi_{h,f_1}(z_n)) = (\Phi_{h,f_2} \circ \Phi_{h,f_1})(z_n)$. Such a method is referred to as a *splitting method*.

### 13.1.1 Linear ODEs

As a first step to understanding splitting methods, consider the linear ODE

$$\frac{dy}{dt} = Ay$$

Let $A = B + C$, then the splitting method defined by solving first

$$\frac{dy}{dt} = By$$

followed by

$$\frac{dy}{dt} = Cy$$

can be written

$$\Psi_h(y) = e^{Ch}e^{Bh}$$

The local error is

$$le(y; h) = \Psi_h(y) - \Phi_h(y) = e^{Ch}e^{Bh} - e^{Ah}$$

Let us work out these formulas in order to estimate the local error. Using the exponential series, we have

$$\begin{aligned} le(y;h) &= (I + Ah + \frac{1}{2}A^2h^2 + \ldots) - (I + Ch + \frac{1}{2}C^2h^2 + \ldots)(I + Bh + \frac{1}{2}B^2h^2 + \ldots) \\ &= (I + Ah + \frac{1}{2}A^2h^2) - (I + h(B+C) + h^2(CB + \frac{1}{2}C^2 + \frac{1}{2}B^2)) + O(h^3) \\ &= h^2(\frac{1}{2}(C+B)^2 - (CB + \frac{1}{2}C^2 + \frac{1}{2}B^2) + O(h^3) \\ &= h^2[\frac{1}{2}C^2 + \frac{1}{2}CB + \frac{1}{2}BC + \frac{1}{2}B^2 - CB - \frac{1}{2}C^2 - \frac{1}{2}B^2] + O(h^3) \\ &= \frac{h^2}{2}(BC - CB) + O(h^3) \\ &= \frac{h^2}{2}[B,C] + O(h^3) \end{aligned}$$

the matrix $[B,C] = BC - CB$ is called the "commutator" of the matrices $B$ and $C$. If the commutator is zero, we say that the matrices commute. In that case, it can be shown that not only the 2nd order term in the local error vanishes, but moreover

$$e^{hB}e^{hC} = e^{Ah}$$

i.e. the splitting method is exact. However, this is a rare situation and in general we expect that a splitting method defined in this way is convergent by Theorem 6.2.1 with $p = 1$.

More complicated splitting methods for linear systems can be constructed in several ways. First, we could imagine splitting the into more parts: $A = B + C + D + \ldots$. Second, we could imagine writing

$$B = \sum_{i=1}^{s} \beta_i B$$

$$C = \sum_{i=1}^{s} \gamma_i C$$

then we could define a splitting method that is based on solving successively

$$\frac{dy}{dt} = \beta_1 By$$

$$\frac{dy}{dt} = \gamma_1 Cy$$

$$\frac{dy}{dt} = \beta_2 By$$

$$\frac{dy}{dt} = \gamma_2 Cy$$

etc., each for a step of size $h$.

This is equivalent to approximating the exponential $e^{hA}$ by

$$e^{\gamma_s Ch} e^{\beta_s Bh} e^{\gamma_{s-1} Ch} e^{\beta_{s-1} Bh} \cdots e^{\gamma_1 Ch} e^{\beta_1 Bh}$$

It turns out that this composition gives at least a first order method, but the order can be higher. For example, it can be shown that the so-called Strang splitting (named for the mathematician Gil Strang)

$$\Psi_h(y) = e^{\frac{h}{2}C} e^{hB} e^{\frac{h}{2}C} y$$

defines a 2nd order method.

### 13.1.2 Nonlinear ODEs

Now let $f = f_1 + f_2$ be a real-valued function. We will denote the flow map on $f$ by $\Phi_{h,f}$ and on each of the two parts by $\Phi_{h,f_1}$ and $\Phi_{h,f_2}$. The question we would like to answer is: what is the local error associated to the map

$$\Psi_{h,f} = \Phi_{h,f_2} \circ \Phi_{h,f_1}?$$

As before, let $z(h) = (\Phi_{h,f_2} \circ \Phi_{h,f_1})(y) = \Phi_{h,f_2}(\Phi_{h,f_1}(y))$. Noting that $\Phi_{0,\bullet}(y) = y$, we see that

$$z(0) = y$$

We know that

$$\Phi_{h,f_1}(y) = y + hf_1(y) + \frac{1}{2}h^2 f_1'(y) f_1(y) + \dots$$

and

$$\Phi_{h,f_2}(y) = y + hf_2(y) + \frac{1}{2}h^2 f_2'(y) f_2(y) + \dots$$

Hence,

$$\begin{aligned} \Phi_{h,f_2}(\Phi_{h,f_1}(y)) &= \left[y + hf_1(y) + \mathcal{O}(h^2)\right] + hf_2\left[y + hf_1(y) + \mathcal{O}(h^2)\right] + \mathcal{O}(h^2) \\ &= y + hf_1(y) + hf_2(y) + \mathcal{O}(h^2), \end{aligned}$$

that is,

$$z(h) = y + hf(y) + \mathcal{O}(h^2)$$

This shows that the method is at least first order.

More generally, it is possible to prove the following result:

**Theorem 13.1.1 (Splitting Methods)** *The splitting method $\Phi_{h,f_2} \circ \Phi_{h,f_1}$, where $f = f_1 + f_2$ has local error*

$$le(y;h) = \frac{h^2}{2}[f_1, f_2] + O(h^3)$$

*where $[f, g]$ is the* commutator of the vector fields *$f$ and $g$, which is another vector field*

$$[f, g] = \frac{\partial f}{\partial y} g - \frac{\partial g}{\partial y} f$$

The more general form of this result follows from the Baker-Campbell-Hausdorff theorem, which tells how to compute all the terms in the expansion of the product of the two flow maps. In fact, one can show

**Theorem 13.1.2 (Baker-Campbell-Hausdorff)** *The splitting method $\Phi_{h,f_2} \circ \Phi_{h,f_1}$, where $f = f_1 + f_2$ has local error*

$$le(y;h) = \frac{h^2}{2}[f_2, f_1] + \frac{h^3}{12}([f_2,[f_2,f_1]] - [f_1,[f_2,f_1]]) + \dots$$

If the vector fields commute, i.e. $[f_2, f_1] = 0$, then the splitting is again exact (however, such flow are not usually very interesting).

Consider the vector field obtained by writing the second order equation $\ddot{x} = g(x)$ in first order form:

$$f(x,u) = \begin{bmatrix} u \\ g(x) \end{bmatrix}.$$

Let $f_1$ and $f_2$ be defined by

$$f_1(x,u) = \begin{bmatrix} 0 \\ g(x) \end{bmatrix}, \qquad f_2(x,u) = \begin{bmatrix} u \\ 0 \end{bmatrix},$$

It is easy to see that each of the differential equation systems

$$\frac{d}{dt}\begin{bmatrix} x \\ u \end{bmatrix} = f_1(x,u)$$

and

$$\frac{d}{dt}\begin{bmatrix} x \\ u \end{bmatrix} = f_2(x,u)$$

is integrable (Euler's method gives the exact solution). A first order method arises by composing the two flows.

With the same vector field decomposition, we can construct a second order method based on Strang splitting by solving successively

$$\frac{d}{dt}\begin{bmatrix} x \\ u \end{bmatrix} = \frac{1}{2}f_1(x,u)$$

and, then

$$\frac{d}{dt}\begin{bmatrix} x \\ u \end{bmatrix} = f_2(x,u)$$

and, finally,

$$\frac{d}{dt}\begin{bmatrix} x \\ u \end{bmatrix} = \frac{1}{2}f_1(x,u)$$

This method is referred to as the Leapfrog method. Note that by reversing the order of the flows, we are actually composing a half-step of the first order splitting with its adjoint (the

adjoint of an exact flow is the flow itself), so the Leapfrog method (and Strang splitting in general) is self-adjoint.

**Example.** For $u$ the population of predators and $v$ the population of prey, an alternative Lotka-Volterra model can be written

$$u' = u(v-2), \quad v' = v(1-u)$$

A simple splitting with $y = (u, v)^T$ is

$$f(y) = \begin{pmatrix} u(v-2) \\ 0 \end{pmatrix}, \quad g(y) = \begin{pmatrix} 0 \\ v(1-u) \end{pmatrix}$$

The exact solution of $y' = f(y)$ on $[0, h]$ is

$$u(h) = e^{h(v(0)-2)}u(0), \quad v(h) = v(0).$$

The exact solution of $y' = g(y)$ on $[0, h]$ is

$$u(h) = u(0), \quad v(h) = e^{h(1-u(0))}v(0).$$

Combining these flows gives the first order method

$$\begin{aligned} u_{n+1} &= e^{h(v_n-2)}u_n, \\ v_{n+1} &= e^{h(1-u_{n+1})}v_n \end{aligned}$$

A second order, self-adjoint method can be derived by composing a half step of the above method with its adjoint. Noting that the exact solution of a subproblem is self-adjoint:

$$\begin{aligned} u_{n+1/2} &= e^{\frac{h}{2}(v_n-2)}u_n, \\ v_{n+1} &= e^{h(1-u_{n+1/2})}v_n, \\ u_{n+1} &= e^{\frac{h}{2}(v_{n+1}-2)}u_{n+1/2}. \end{aligned}$$

With initial conditions $u(0) = 6$ and $v(0) = 2$, the solution is periodic with approximate period $T = 6.336$. The solution at time $t = 6.4$ is $u(6.4) \approx 5.89200351$, $v(6.4) \approx 1.45888866$. In Figure 13.2, the solution is illustrated using the 2nd order splitting with $h = 0.1$. Also the error behavior for the first and second order splittings is shown.

Clearly, a multiple term splitting is also possible. For example, for the ODE

$$y' = f_1(y) + \cdots + f_k(y),$$

the splitting method

$$y_{n+1} = \Phi^h_{f_k} \circ \cdots \circ \Phi^h_{f_1}\, y_n \tag{13.1}$$

is first order in general.

It is not necessary that the split subproblems be solved exactly. Since the composition is only first order, the subproblems may also be approximated by a first order method. However, one often looks for an exactly solvable splitting.

The prudent choice of the splitting can make or break a splitting method. For one thing, if the splitting is such that the exact solution of a subproblem is not known, then it must be approximated with, for example, an RK method, and there will be approximation errors in addition to the splitting errors. Furthermore, if two terms are strongly coupled, then splitting them apart could cause large errors. For example, consider a mechanical system with two strong opposing forces that nearly balance each other out. The combined effect of the forces is small. However, each force by itself is large and would cause big accelerations if split apart. Normally one chooses the splitting based on physical or other (structural) considerations.

## 13.2 Exercises

1. Write out the formulas for the splitting method for the differential equation system

$$\frac{dy}{dt} = ay + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} y$$

where $a$ is a scalar, based on solving

$$\frac{dy}{dt} = ay$$

followed by

$$\frac{dy}{dt} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} y$$

Comment on the order of the resulting method. Repeat the exercise for

$$\frac{dy}{dt} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} y + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} y$$

solving first

$$\frac{dy}{dt} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} y$$

followed by

$$\frac{dy}{dt} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} y$$

2. Prove that the Strang splitting gives a 2nd order integrator (for nonlinear problems).

3. Discuss the preservation of equilibrium points by splitting methods. Can you think of a splitting that preserves equilibrium points exactly?

4. Compare the first and second order splittings for the Lotka-Volterra model and and Heun's method (8.3) for a long numerical simulation (say, 10 to 100 periods). Can you discern any qualitative differences in the solutions? Use a range of step sizes.

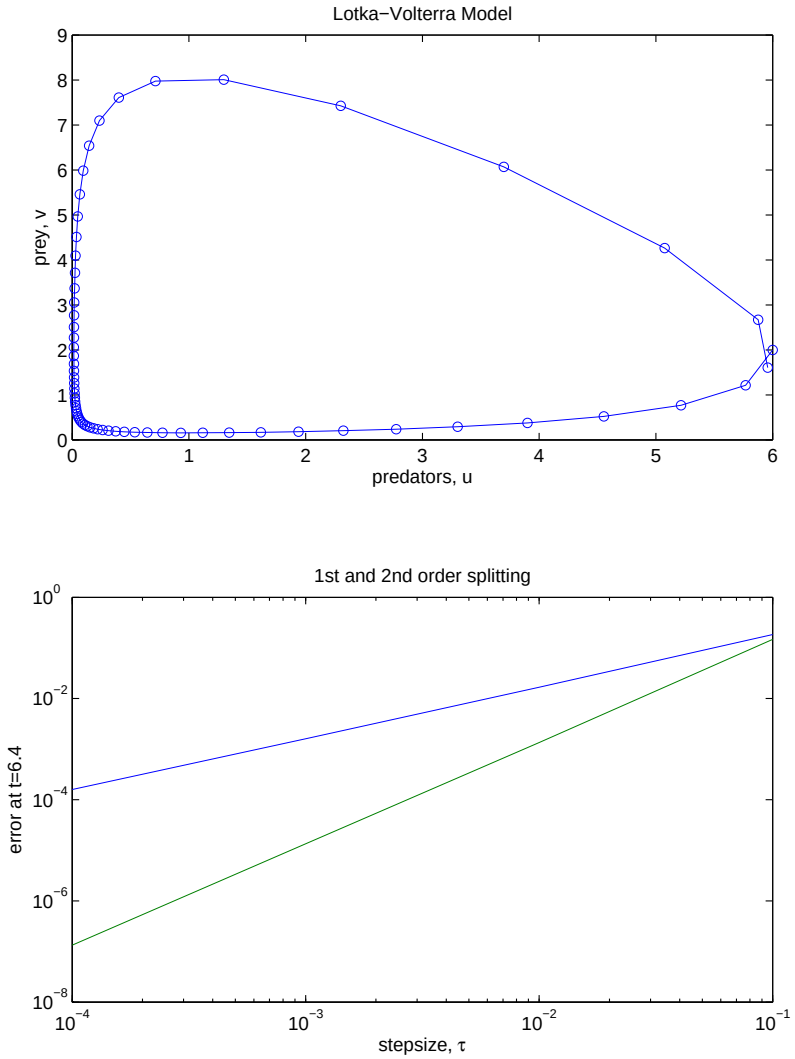



Figure 13.2: One-period of the Lotka-Volterra model: solution (top) and error (bottom).

# Chapter 14

# First Integral

**Main concepts:** First integrals, linear, quadratic, discrete gradients.

## 14.1 First Integrals

In Chapter 5 we said defined the space $\mathbb{R}^d$ to be the phase space of the differential equation (1.13). Each point in $\mathbb{R}^d$ defines a state of the system, and a solution of (1.13) traces a curve in $\mathbb{R}^d$. Given a function $I(y) : \mathbb{R}^d \to \mathbb{R}$, we can evaluate this function along a solution to (1.13), i.e. sampling $I(y(t))$. For some special functions $I$ we may find that that $I$ is constant along solutions to (1.2). Such a function is called a *first integral* of (1.2).

For example, many models from physics are designed to incorporate energy conservation. In the case of the harmonic oscillator (5.4) the energy is just

$$E(x, p) = \frac{1}{2}p^2 + \frac{1}{2}\omega^2 x^2.$$

You can easily check that this is constant along solutions of (5.4) by differentiation of the expression with respect to time, and use of the differential equations:

$$\begin{aligned} \frac{d}{dt}E(x(t), p(t)) &= p\dot{p} + \omega^2 x\dot{x} \\ &= p(-\omega^2 x) + \omega^2 xp = 0. \end{aligned}$$

The energy of a conservative mechanical system is one example of a first integral.

We say that for any differential equation of the form (1.13), a function $I(y)$ is a **first integral** if the function $I$ is constant when evaluated along any solution of the differential equation. Using the equation (1.13), we can easily see that $I$ is a first integral provided, whenever $y = y(t)$ is a solution of (1.13),

$$\frac{d}{dt}I(y(t)) \equiv 0.$$

Using the chain rule, this just means

$$\nabla I(y(t)) \cdot \frac{dy}{dt}(t) \equiv 0,$$

or (since $dy/dt = f(y)$ for any solution of (1.13)), simply

$$\nabla I \cdot f(y) = 0 \quad \text{or,} \quad (\nabla I)^T f(y) = 0.$$

Other names for first integrals include: *invariant*, *constant of motion*, and *conserved quantity*.

First integrals play several crucial roles in studies of dynamical systems. For example it is often the case that first integrals help to define the problem; their values can be viewed as constants that have specific physical significance. First integrals may confine the solution to a bounded region of phase space. In many systems, the first integrals are the only computable measures of the performance of the numerical method – we cannot plot the error in the solution unless the exact solution is available, but we can almost always compute and monitor the energy or another first integral, and use this to assess the quality of the approximation.

**Example.** The rigid body equations describe the motion of a rotating object in the absence of external forces. The equations are

$$\frac{dy}{dt} = y \times \Gamma y, \tag{14.1}$$

where $y(t) = (y_1(t), y_2(t), y_3(t))^T \in \mathbb{R}^3$ and $\Gamma = \text{diag}(I_1^{-1}, I_2^{-1}, I_3^{-1})$ is a diagonal matrix and $I_j > 0$, $j = 1, \ldots, 3$ are the moments of inertia about the principle axes of the object.

For any solution to the rigid body equations the functions:

$$H(y_1, y_2, y_3) = \frac{1}{2}\left(\frac{y_1^2}{I_1} + \frac{y_2^2}{I_2} + \frac{y_3^2}{I_3}\right)$$

and

$$K(y_1, y_2, y_3) = y_1^2 + y_2^2 + y_3^2$$

are constant (and equal to $H(y(0))$ and $K(y(0))$, respectively). Note that the solution is constrained to the intersection of the surfaces of a sphere and an ellipsoid, i.e. a closed one-dimensional submanifold of $\mathbb{R}^3$. The preservation of either of these first integrals guarantees the boundedness of the solution as $t \to \infty$.

In this chapter, we discuss the conservation of first integrals by numerical methods.

## 14.2 First Integrals Under Discretization

Suppose $d = 1$ in (1.13) then preservation of a first integral means that the scalar equation $I(y) = I_0$ is satisfied. Such a nonlinear equation, for smooth $I$, generally has at most just

some discrete set of point solutions there can be no dynamics. Thus it is clear that in order to have interesting dynamics with a first integral we must consider systems of dimension $d \geq 2$. In this case, $\{y|I(y) = I_0\}$ defines a manifold, a point set with a continuous structure. The solution of the differential equations evolves on this manifold. We can write this, compactly, as

$$I \circ \Phi_t = I.$$

Now consider the preservation of such a first integral by a numerical method, which we assume to be represented as a map $\Psi_h$. Under what circumstances do we have

$$I \circ \Psi_h = I?$$

We will discuss several examples to show that it is possible to have exact conservation. First suppose a system has a linear first integral. Such an integral can be represented by

$$I(y) = r + s \cdot y,$$

where $r$ is a scalar, and $s$ a vector in $\mathbf{R}^d$. We assume that $\nabla I \cdot f(y) = 0$ which just means that $s \cdot f(y) = 0$.

If we are given a Runge-Kutta method described by the equations

$$y_{n+1} = y_n + h \sum_i b_i F_i, \quad F_i = f(Y_i) = f(y_n + h \sum_j a_{ij} F_j), \quad i = 1, \ldots, s,$$

we can compute the first integral at time $t_{n+1}$:

$$\begin{aligned} I(y_{n+1}) &= r + s \cdot y_{n+1} \\ &= r + s \cdot (y_n + h \sum_i b_i F_i) \\ &= r + s \cdot y_n + h \sum_i b_i s \cdot F_i \\ &= I(y_n). \end{aligned}$$

Thus, *Runge-Kutta methods preserve arbitrary linear first integrals.*

Next consider the Kepler problem

$$\begin{aligned} \frac{dx}{dt} &= u \\ \frac{dy}{dt} &= v \\ \frac{du}{dt} &= -\frac{x}{(x^2+y^2)^{3/2}} \\ \frac{dv}{dt} &= -\frac{y}{(x^2+y^2)^{3/2}} \end{aligned}$$

We know that the angular momentum

$$L = xv - yu$$

is a conserved quantity. To see this, differentiate $L$ to obtain

$$\frac{dL}{dt} = xv' + x'v - yu' - y'u,$$

where prime is used to denote differentiation with respect to $t$. Using the differential equation, we have

$$\frac{dL}{dt} = -\frac{xy}{(x^2+y^2)^{3/2}} + uv + \frac{yx}{(x^2+y^2)^{3/2}} - vu = 0.$$

The Kepler problem can be written more compactly by introducing vectors $q = (x, y)$, $p = (u, v)$, then the equations of motion become

$$\begin{aligned} \frac{dq}{dt} &= p \\ \frac{dp}{dt} &= -\frac{q}{\|q\|^3} \end{aligned}$$

Define

$$L = q \times p$$

as the cross product of $q$ and $p$, where these vectors are viewed as extended to vectors in $\mathbf{R}^3$. Only the third component here is nonzero, and

$$\frac{d}{dt}L = q \times p' + q' \times p = 0$$

Discretizing with, say, Euler's method, does not result in conservation of $L$. (Check this!) But if we use the one-sided Euler method, we have

$$\begin{aligned} q_{n+1} &= q_n + hp_{n+1} \\ p_{n+1} &= p_n - h\frac{q_n}{\|q_n\|^3} \end{aligned}$$

Now compute

$$\begin{aligned} L_{n+1} &= q_{n+1} \times p_{n+1} \\ &= (q_n + hp_{n+1}) \times p_{n+1} \\ &= q_n \times p_{n+1} + hp_{n+1} \times p_{n+1} \\ &= q_n \times p_{n+1} \\ &= q_n \times (p_n - h\|q_n\|^{-3}q_n) \\ &= q_n \times p_n - h\|q_n\|^{-3}q_n \times q_n \\ &= q_n \times p_n \\ &= L_n \end{aligned}$$

Thus the one-sided Euler method exactly conserves the angular momentum in the Kepler problem. This is quite a remarkable property in itself, but does this hold in more general cases?

### 14.2.1 Conservation of quadratic first integrals by Runge-Kutta methods

The angular momentum above is an example of a quadratic first integral. It is natural to ask if the one-sided Euler method preserves all such integrals. Unfortunately this cannot be the case. For example for the harmonic oscillator with frequency $\omega = 1$, the exact solutions are circles satisfying $x^2 + p^2 = 2E$, where $E$ is the energy. It can be easily checked that the solution using the one-sided Euler method is not a circle, however. Applying the method, we have $x_{n+1} = x_n + hp_{n+1}$, $p_{n+1} = p_n - hx_n$, so

$$\begin{aligned} x_{n+1}^2 + p_{n+1}^2 &= (x_n + hp_n - h^2 x_n)^2 + (p_n - hx_n)^2 \\ &= x_n^2 + 2hx_n p_n + h^2 p_n^2 + h^4 x_n^2 - 2h^2 x_n^2 - 2h^3 x_n p_n + p_n^2 - 2hx_n p_n + h^2 x_n^2 \\ &= x_n^2 + p_n^2 + h^2(p_n^2 - x_n^2) + O(h^3). \end{aligned}$$

The second order term does not vanish identically, so the energy cannot be conserved.

Remarkably, however, the implicit midpoint rule (8.5) *does* conserve all quadratic invariants

$$I(y) = y^T C y, \quad y^T C f(y) = 0, \quad C^T = C$$

This is easily seen by writing (8.5) as

$$\frac{y_{n+1} - y_n}{\tau} = f(\frac{y_{n+1} + y_n}{2})$$

and multiplying both sides with $\frac{1}{2}(y_{n+1} + y_n)^T C$ to get

$$\frac{1}{2\tau}(I(y_{n+1}) - I(y_n)) = (\frac{y_{n+1} + y_n}{2})^T C f(\frac{y_{n+1} + y_n}{2}) = 0.$$

More generally, we prove the following theorem:

**Theorem 14.2.1 (Quadratic First Integrals)** *Let $dy/dt = f(y)$ have a quadratic first integral. A Runge-Kutta method preserves this quadratic first integral if*

$$b_i b_j - b_i a_{ij} - b_j a_{ji} = 0 \tag{14.2}$$

We prove this by assuming that $I(y) = r + s \cdot y + y^T T y$ is such a first integral, where $r$ is a scalar, $s$ is a vector of dimension $d$ and $T$ is a symmetric $d \times d$ matrix. Note that the condition for $I$ to be a first integral is that $\nabla I \cdot f \equiv 0$, or

$$s \cdot f + 2y^T T f \equiv 0.$$

This implies that

$$s \cdot f \equiv -2y^T T f. \tag{14.3}$$

Applying an RK method to the system we have

$$y_{n+1} = y_n + h\sum_i b_i F_i; \quad F_i = f(y_n + h\sum_j a_{ij} F_j)$$

thence

$$I(y_n{+}1) = r + s \cdot (y_n + h\sum_i b_i F_i) + (y_n + h\sum_i b_i F_i)^T T (y_n + h\sum_i b_i F_i)$$

Expanding this out, we have

$$I(y_{n+1}) = I(y_n) + h\sum_i b_i s \cdot F_i + 2h\sum_i b_i y_n^T T F_i + h^2(\sum_i b_i F_i) T (\sum_i b_i F_i)$$

We need to show that, assuming (14.2),

$$J = +\sum_i b_i s \cdot F_i + 2\sum_i b_i y_n^T T F_i + h(\sum_i b_i F_i) T (\sum_i b_i F_i) = 0$$

Now, by virtue of (14.3),

$$s \cdot F_i = s \cdot f(Y_i) = -2Y_i^T T f(Y_i) = -2Y_i^T T F_i$$

hence

$$\begin{aligned} J &= \sum_i b_i(-2) Y_i^T T F_i + 2\sum_i b_i y_n^T T F_i + h(\sum_i b_i F_i) T (\sum_i b_i F_i) \\ &= \sum_i b_i (y_n - Y_i)^T T F_i + h(\sum_i b_i F_i) T (\sum_i b_i F_i) \\ &= -h\sum_i b_i \sum_j a_{ij} F_j^T T F_i + h(\sum_i b_i F_i) T (\sum_i b_i F_i) \end{aligned}$$

Let $F_i^T F_j = B_{ij}$. In general we assume that these terms, which depend on the vector field, are uncorrelated. In that case $J$ will vanish identically provided each of the coefficients multiplying the $B_{ij}$ vanishes. This coefficient is

$$b_i a_{ij} + b_j a_{ji} - b_i b_j$$

which concludes the proof.

One class of method which satisfies the conditions (14.2) is the class of Gauss-Legendre methods, which we have already learned are the optimal order methods for a given number of stages (they are also fully implicit, and therefore, often, too expensive to use in many practical examples).

In terms of finding methods that conserve *all* first integrals of a given class, this is as good as it gets: we do not have broad classes of methods which preserve cubics, quartics, etc. Fortunately, quadratics are an important case.

If we are concerned with preserving a first integral, an alternative approach is to design a method that has this as its specific goal.

## 14.3 Discrete Gradient Methods

The conservation of quadratic invariants by Gauss Runge-Kutta methods is automatic. Any and all such invariants will be conserved whether or not one is aware of them. In this section we describe a method for conserving a known invariant $I(y)$.

Define the exterior product of two vectors $f$ and $v$:

$$f \wedge v = fv^T - vf^T, \qquad (f \wedge v)_{ij} = f_i v_j - v_i f_j.$$

If $I(y)$ is a first integral of (1.3), then, except at points $y^*$ where $\nabla I(y^*) = 0$, $f$ can be written as

$$f(y) = B(y)\,\nabla I(y) = \frac{1}{\|\nabla I(y)\|^2}(f \wedge \nabla I)\,\nabla I(y),$$

where $B(y)$ is skew-symmetric: $B^T = -B$.

Let $\bar{B}(y, z)$ satisfy:

- skew-symmetric structure: $\bar{B}(y, z)^T = -\bar{B}(y, z)$,
- symmetry with respect to its arguments: $\bar{B}(y, z) = \bar{B}(z, y)$,
- consistency $\bar{B}(y, y) = B(y)$.

Similarly, define $v(y) := \nabla I(y)$ and let $\bar{v}(y, z)$ be a *discrete gradient* of $I$, i.e. satisfying

$$\bar{v}(y, z)^T(y - z) = I(y) - I(z),$$

and consistent $\bar{v}(y, y) = v(y)$.

Then the method

$$y_{n+1} = y_n + \tau \bar{B}(y_{n+1}, y_n)\,\bar{v}(y_{n+1}, y_n)$$

satisfies

$$I(y_{n+1}) = I(y_n),$$

since

$$\begin{aligned} I(y_{n+1}) - I(y_n) &= \bar{v}(y_{n+1}, y_n)^T(y_{n+1} - y_n) \\ &= \tau \bar{v}(y_{n+1}, y_n)^T \bar{B}(y_{n+1}, y_n)\bar{v}(y_{n+1}, y_n) = 0. \end{aligned}$$

One discrete gradient is the *midpoint discrete gradient*. Denote $y_{n+1/2} := (y_{n+1} + y_n)/2$ and $\Delta y := y_{n+1} - y_n$. Then

$$\bar{v}(y_{n+1}, y_n) := v(y_{n+1/2}) + \frac{I(y_{n+1}) + I(y_n) - v(y_{n+1/2})^T \Delta y}{\|\Delta y\|^2}\,\Delta y.$$

A possible realization of the matrix $\bar{B}$ is given by

$$\bar{B}(y, z) = B(\frac{y + z}{2}).$$

**Example.** Mechanical systems (12.3) can be written (using a change in variables to eliminate the mass matrix $M$) in the form

$$\dot{q} = p, \qquad \dot{p} = \nabla V(q),$$

where $F(q) = -\nabla V(q)$. For $y = (q^T, p^T)^T$, this defines a gradient system with skew-symmetric, constant matrix

$$B = \begin{bmatrix} 0 & id \\ -id & 0 \end{bmatrix},$$

and first integral

$$I(q,p) = \frac{p \cdot p}{2} + V(q).$$

Define the discrete gradient $\bar{v}(\bar{q}, p)$ by

$$\bar{v}_q(q_{n+1}, p_{n+1}, q_n, p_n) = \frac{I(q_{n+1}, p_n) - I(q_n, p_n)}{q_{n+1} - q_n},$$

and

$$\bar{v}_p(q_{n+1}, p_{n+1}, q_n, p_n) = \frac{I(q_{n+1}, p_{n+1}) - I(q_{n+1}, p_n)}{p_{n+1} - p_n},$$

where in both cases the division is understood component-wise.

## 14.4 Splitting Methods

Consider a skew-gradient system

$$y' = B(y)\nabla I(y),$$

where $B(y)^T = -B(y)$, and suppose we can find a splitting of the form

$$y' = (B_1(y) + \cdots + B_k(y))\nabla I(y),$$

where $B_i(y)^T = -B_i(y)$, and each subproblem

$$y' = B_i(y)\nabla I(y), \quad i = 1, \ldots, k$$

is exactly integrable.

Then the splitting method (13.1) with

$$f_i = B_i(y)\nabla I(y), \quad i = 1, \ldots, k$$

exactly conserves $I$.

**Example.** Consider the rigid body equations (14.1). Let us derive a splitting method that exactly conserves the energy

$$H(y_1, y_2, y_3) = \frac{1}{2}\left(\frac{y_1^2}{I_1} + \frac{y_2^2}{I_2} + \frac{y_3^2}{I_3}\right)$$

The rigid body equations can be written in the form

$$y' = B(y)\nabla H(y),$$

with skew-symmetric $B(y)$ split as

$$B(y) = \begin{bmatrix} 0 & -y_3 & y_2 \\ y_3 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -y_1 \\ 0 & y_1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & y_2 \\ 0 & 0 & 0 \\ -y_2 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -y_3 & 0 \\ y_3 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = B_1(y)+B_2(y)+B_3(y).$$

Each of the split sub-systems

$$y' = B_i(y)\nabla H(y), \quad i = 1, \ldots, 3$$

is exactly solvable as a harmonic oscillator, and preserves the manifold $H = const$.

## 14.5 Exercises

1. What is the energy function for the pendulum? Use the fact that energy is conserved (and Matlab or a graphing calculator) to sketch the phase portrait for the pendulum.

2. Prove that the Lotka-Volterra system has a first integral:

$$K(Y, R) = C \log Y - DY + A \log R - BR.$$

3. Assume an autonomous differential equation system is given (**??**)and is known to have a first integral. Show that any time-rescaling of this differential equation has the same first integral. To rescale time, we introduce a time-transformation, such as

$$\frac{dt}{d\tau} = g(y),$$

where $g$ is a scalar-valued function.

4. Show that the implicit midpoint method and the 2-stage Gauss-Legendre scheme satisfy the condition (14.2).

5. Can you think of a splitting that conserves the other first integral of the rigid body, i.e. the total angular momentum $I = \frac{1}{2}y^T y$? Can you recall a method that conserves both $H$ and $I$?

# Chapter 15

# Constraints and Projection

**Main concepts:** Constrained dynamics, projection, first integrals via projection

## 15.1 Weak Invariants

In the previous sections we have looked at the case where $\nabla I(y) \cdot f(y) = 0$, so that $I(y)$ is a conserved quantity associated to any initial condition. Another case of conservation is one in which there exists an invariant submanifold $\mathcal{M}$ of dimension $d - m$ defined by $g(y) = 0$, where $g : \mathbb{R}^d \to \mathbb{R}^m$, such that if the initial condition satisfies $y_0 \in \mathcal{M}$, then $y(t) \in \mathcal{M}$ for all $t$.

The elements $g_i(y)$, $i = 1, \ldots, m$ are weak invariants, satisfying $\nabla g(y) \cdot f(y) = 0$ for all $y \in \mathcal{M}$ (as opposed to all $y \in \mathbb{R}^d$ for strong invariants.)

Weak invariants can be conserved by projection.

## 15.2 Projection

The straightforward way to obtain methods to preserve a given first integral is by projection. To motivate this, consider a vector $u \in \mathbf{R}^d$ and a linear subspace (a hyperplane $P$) defined by the condition

$$s \cdot y = 0$$

for some vector $s \in \mathbb{R}^d$. We can in general project $u$ into the hyperplane by solving the following system involving a *Lagrange multiplier* $\lambda$:

$$\begin{aligned} y &= u + \lambda s, \\ 0 &= s \cdot y. \end{aligned}$$

Plugging the first equation into the second, we have

$$0 = s \cdot (u + \lambda s)$$

so that

$$\lambda = -(s \cdot s)^{-1} s \cdot u,$$

where we have assumed that $s$ is not the zero vector so that $s \cdot s = \|s\|^2 \neq 0$. Now

$$y = u - \|s\|^{-2} s \cdot u.$$

$y$ is called the orthogonal projection of $u$ onto the hyperplane $P$.

This technique can be extended to the case where the surface of interest is not a hyperplane but instead some more general manifold of dimension $R^{d-1}$ defined by the condition

$$I(y) = 0$$

Then, as before, we introduce a Lagrange multipler, $\lambda$ and solve the (now nonlinear) equation system

$$\begin{aligned} y &= u + \lambda \nabla_y I, \\ 0 &= I(y). \end{aligned}$$

Solving this system corresponds to minimizing the functional

$$K(y) = \|y - u\|^2$$

subject to the condition $I(y) = 0$. Thus, what we are doing is finding the nearest point to $u$ on the constraint set $\mathcal{M} = \{y | I(y) = 0\}$. It can be shown that this projection is orthogonal in the sense diagrammed in Figure 15.1a.

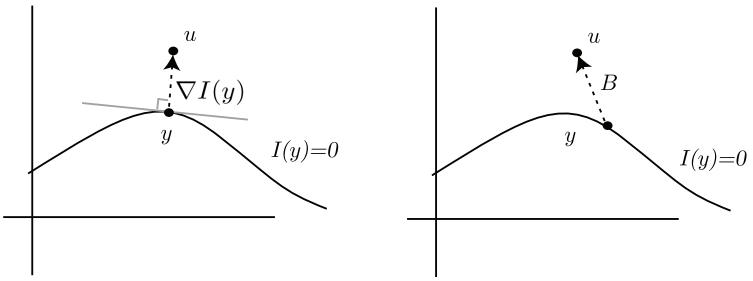



Figure 15.1: Orthogonal (a) and Oblique (b) projections onto a manifold.

There is a broader class of projections which do not necessarily find the nearest point (which is not always the essential requirement), but some other point on the constraint manifold. For example, if we know that $B$ is *any* vector such that $s \cdot B \neq 0$, then we can solve the hyperplane projection problem by finding $y$ and $\lambda$ (which will be different than before) such that

$$\begin{aligned} y &= u + \lambda B, \\ 0 &= s \cdot y \end{aligned}$$

This type of *oblique projection* can also be of use for nonlinear manifolds. We can then use the equations

$$\begin{aligned} y &= u + \lambda B, \\ 0 &= I(y). \end{aligned}$$

(See Figure 15.1b.)

## 15.3 Numerical Methods with Projection

Now consider a point in space $y_0$ satisfying $I(y_0) = 0$. We take a step using any $p$th order numerical method

$$\hat{y}_1 = \Psi_h(y_0)$$

How much error can be introduced in the first integral $I(y) = 0$ by this step? Since the numerical method has local error $p$, we know that $\hat{y}_1$ differs from $\Phi_h(y_0)$ by at most $O(h^{p+1})$. Assuming $I$ is a smooth function, and since we know that $I \circ \Phi_h = I$, we must have

$$I(\hat{y}_1) = I(\hat{y}_1) - I(\Phi_h(y_0)) = O(h^{p+1}).$$

The idea in *projection methods* is that we try to eliminate the small integral error by calculating the projection of the numerical solution using $\Psi$ onto the integral manifold.

In an orthogonal projection method, we solve for $y_1$ so that

$$\begin{aligned} y_1 &= \hat{y}_1 + \lambda \nabla I(y_1) \\ 0 &= I(y_1) \end{aligned}$$

Here we can see already a problem: even if our original numerical method was explicit, the method obtained in this way will be implicit, since we must solve this nonlinear equation system to obtain $y_1$.

There is no way around solving some equations, but we can simplify our lives in many cases by using an oblique projection. The natural way to define this is to use the direction $\nabla I(\hat{y}_1)$ which is available to us after the numerical method step. Thus we think of solving for $\lambda$ and $y_1$ from the equations

$$\begin{aligned} y_1 &= \hat{y}_1 + \lambda B(\hat{y}_1) && (15.1) \\ 0 &= I(y_1) && (15.2) \end{aligned}$$

where $B$ is a suitable vector valued function of $y$. The advantage here is that we are able to compute the step by solving a single scalar nonlinear equation instead of a system of such equations (as would be the case if we used the orthogonal projection). Once $y_1$ is found, we use this as the starting point for the new step.

To explain, we use the example of the Lotka-Volterra model (1.6)

$$\begin{aligned} du/dt &= u(1 - v) \\ dv/dt &= v(u - 1) \end{aligned}$$

which has the first integral (as we know)

$$K(u,v) = \ln u + \ln v - u - v.$$

An Euler step gives $\hat{u}_{n+1}$, $\hat{v}_{n+1}$. We then may project this onto the constraint manifold $K(u,v) - K_0 = 0$ by solving

$$\begin{aligned} u_{n+1} &= \hat{u}_{n+1} - \lambda[1 - \hat{u}_{n+1}^{-1}] \\ v_{n+1} &= \hat{v}_{n+1} - \lambda[1 - \hat{v}_{n+1}^{-1}] \\ K_0 &= \ln u_{n+1} + \ln v_{n+1} - u_{n+1} - v_{n+1}. \end{aligned}$$

After introducing the first two equations into the third, we are left with a nonlinear equation to solve for $\lambda$. This could be done by Newton iteration.

Define a map $\Omega_h$ from $y_n$ to $y_{n+1}$ by the equations

$$\begin{aligned} \hat{y}_{n+1} &= \Psi_h(y_n) \\ y_{n+1} &= \hat{y}_{n+1} + \lambda B(\hat{y}_{n+1}) \\ I(y_{n+1}) &= 0. \end{aligned}$$

It is then a substantial exercise to analyse the conditions under which this map is well-defined and the convergence and order of accuracy of the resulting iteration scheme. However, this can be done and under mild assumptions on $I$ and $B$, it can be shown that $\Omega_h$ generally retains the order of accuracy of $\Psi_h$.

## 15.4 Preserving Weak Invariants Under Discretization

We introduce the Lagrange multiplier $\lambda \in \mathbb{R}^m$ and augment (1.3)

$$\begin{aligned} y' &= f(y) + g'(y)^T\lambda \\ 0 &= g(y) \end{aligned}$$

.

Let $\Psi_\tau$ be any one-step method for (1.3), and let $y^*_{n+1}$ be the output of a single step of the method with input $y_n$:

$$y^*_{n+1} = \Psi_\tau y_n$$

In general $y^*_{n+1} \notin \mathcal{M}$ and so must be projected onto $\mathcal{M}$ by solving a nonlinear problem for $\lambda$:

$$g(y^*_{n+1} + g'(y^*_{n+1})^T\lambda) = 0$$

and correcting:

$$y_{n+1} = y^*_{n+1} + g'(y^*_{n+1})^T\lambda.$$

For a $p$th order method, the distance from $y^*_{n+1}$ to $\mathcal{M}$ is $\mathcal{O}(\tau^{p+1})$, so convergence of Newton iteration is quite fast.

**Example.** Consider a particle of unit mass in an electric field $V(x)$, constrained to a sphere. The equations of motion of the free particle are:

$$\ddot{x} = -\nabla V(x)$$

The constraint is

$$g(x) = \|x\|^2 - 1 = 0.$$

The constrained equations are

$$\begin{aligned} \dot{x} &= v + 2x\lambda \\ \dot{v} &= -\nabla V(x) \\ 0 &= g(x) \end{aligned}$$

The constrained version of Euler's method is

$$\begin{aligned} x^*_{n+1} &= x_n + \tau v_n \\ v_{n+1} &= v^*_{n+1} = v_n - \tau \nabla V(x_n) \\ x_{n+1} &= x^*_{n+1} + 2x^*_{n+1}\lambda = (1+2\lambda)x^*_{n+1} \\ 0 &= g(x_{n+1}) \end{aligned}$$

where the last two equations are solved simultaneously to give $\lambda$ and $x_{n+1}$, i.e.

$$\|x_{n+1}\|^2 = 1 = (1+2\lambda)^2\|x^*_{n+1}\|^2 \Rightarrow \lambda = \frac{1}{2}\left(\frac{1}{\|x^*_{n+1}\|} - 1\right),$$

and $x_{n+1}$ follows by substitution.

**Remark 15.4.1** *An important point should be made regarding such projection methods: we have mentioned in the introduction to the section that a first integral is often our only way of monitoring the quality of a solution. If we design a numerical method to exactly conserve the first integral or first integrals of a system, we may destroy our only way of knowing whether the solution is correct.*

# Chapter 16

# Symplectic Flows and Maps and Volume Preservation

**Main concepts:** Liouville's equation, cat wash cycles, volume preservation, symplectic maps, symplectic RK methods, symplectic splitting methods

## 16.1 Hamiltonian Systems

Hamiltonian systems are pervasive in chemical and physical simulation. These systems arise at very large scales (celestial mechanics, cosmology) and at small scales (atomic and molecular simulation).

Suppose that the *configuration* of some (mechanical) system with $d$ degrees of freedom can be represented by a set of $d$ real numbers $q = (q_1, \ldots, q_d)^T$. (Think of the positions of all nine planets and the sun in our solar system. In some reference frame, the positions are specified by three real numbers for each body, so $d = 3 \times 10 = 30$.)

The motion of the system is given by $q(t)$, a curve in $\mathbb{R}^d$.

Lagrange's theory of mechanical systems is based on two real (scalar) valued functions, the kinetic and potential energies

$$T(q(t), \dot{q}(t)), \quad U(q(t)).$$

The Lagrangian $L$ is the difference of these

$$L(q, \dot{q}) = T - U,$$

and Lagrange's equations of motion are

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}(q, \dot{q})\right) = \frac{\partial L}{\partial q}(q, \dot{q}).$$

Hamilton's equations of motion are derived from Lagrange's equations by introducing a new variable

$$p = \frac{\partial L}{\partial \dot{q}}(q, \dot{q}), \quad \text{i.e.,} \quad p_i = \frac{\partial L}{\partial \dot{q}_i}$$

and the *Hamiltonian*

$$H(q, p) = p^T \dot{q} - L(q, \dot{q}), \tag{16.1}$$

where it is implied that the defining relation for $p$ above is invertible for $\dot{q}$ for every $q$, so that we can express

$$\dot{q} = \dot{q}(q, p).$$

Hamilton's equations of motion have the antisymmetric form

$$\dot{q}_i = \frac{\partial H(q,p)}{\partial p_i}, \quad \dot{p}_i = -\frac{\partial H(q,p)}{\partial q_i}, \tag{16.2}$$

or simply

$$\dot{q} = \frac{\partial H(q,p)}{\partial p}, \quad \dot{p} = -\frac{\partial H(q,p)}{\partial q}.$$

We can check that these equations are consistent with Lagrange's equations by differentiating (16.1)

$$\frac{\partial H}{\partial p} = \dot{q} + p^T \frac{\partial \dot{q}}{\partial p} - \frac{\partial L}{\partial \dot{q}} \frac{\partial \dot{q}}{\partial p} = \dot{q}, \qquad \frac{\partial H}{\partial q} = p \frac{\partial \dot{q}}{\partial q} - \frac{\partial L}{\partial q} - \frac{\partial L}{\partial \dot{q}} \frac{\partial \dot{q}}{\partial q} = -\frac{\partial L}{\partial q} = -\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) = -\dot{p}$$

**Example.** The equation for the mathematical pendulum is $\ddot{q} = -\sin(q)$. The kinetic energy is given by $T(\dot{q}) = \frac{\dot{q}^2}{2}$ and the potential energy by $U(q) = -\cos q$. The Lagrangian is $L(q, \dot{q}) = \frac{\dot{q}^2}{2} + \cos q$, and Lagrange's equation is

$$\frac{d}{dt}\dot{q} = -\sin q.$$

For Hamilton's formulation we find

$$p = \dot{q}, \qquad H = p\dot{q} - L = p^2 - (\frac{p^2}{2} + \cos q) = \frac{p^2}{2} - \cos q$$

with equations of motion

$$\dot{q} = \frac{\partial H}{\partial p} = p, \qquad \dot{p} = -\frac{\partial H}{\partial q} = -\sin q.$$

**Example.** A particularly important class of Hamiltonians is the *N-body problem* (here formulated with $q_i \in \mathbb{R}^d$, i.e. the full configuration space is of dimension $Nd$):

$$H(q, p) = \sum_{i=1}^{N} \frac{\|p_i\|^2}{2m_i} + V(q_1, q_2, \ldots, q_N).$$

$m_1, m_2, \ldots, m_N$ are the masses of the $N$ bodies, and $V$ is a potential energy function that describes their interactions. This type of model is routinely used in drug design and materials modelling.

The Hamiltonian is a first integral of (16.2). This is easily checked

$$\frac{dH}{dt} = \sum_i \frac{\partial H}{\partial q_i}\dot{q}_i + \frac{\partial H}{\partial p_i}\dot{p}_i = \sum_i \frac{\partial H}{\partial q_i}\frac{\partial H}{\partial p_i} + \frac{\partial H}{\partial p_i}(-\frac{\partial H}{\partial q_i}) = 0.$$

In most cases, this first integral can be identified with the energy of the system, so that its invariance corresponds to the physical principle of conservation of total energy. Indeed, one often finds $H = T(q,p) + U(q)$.

Sometimes we write (16.2) in the form (1.13) by defining $y = (q_1, \ldots, q_d, p_1, \ldots p_d)^T \in \mathbb{R}^{2d}$, whence

$$\frac{dy}{dt} = J\nabla H(y),$$

with the skew-symmetric matrix $J$ defined by

$$J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}, \tag{16.3}$$

where $I$ is the identity matrix of dimension $d$. Here, the conservation of $H$ is immediate from the skew-gradient form introduced in Chapter 14.

## 16.2 Divergence-Free Vector Fields

Another category of vector field with interesting dynamical features are those with zero divergence. Let $f = f(y)$, $f : \mathbf{R}^d \to \mathbf{R}^d$ be a vector field. Recall that the divergence of $f$ is

$$\mathrm{div} f = \frac{\partial}{\partial y_1} f_1 + \frac{\partial}{\partial y_2} f_2 + \ldots + \frac{\partial}{\partial y_d} f_d.$$

It happens in certain situations that $\mathrm{div} f \equiv 0$. As an illustration, the vector field of the system

$$\begin{aligned} \frac{dx}{dt} &= h_1(y) \\ \frac{dy}{dt} &= h_2(x) \end{aligned}$$

is clearly divergence free, as is the following one:

$$\begin{aligned} \frac{dx}{dt} &= x + h_1(y) \\ \frac{dy}{dt} &= h_2(x) - y \end{aligned}$$

A very important special case of divergence-free vector fields are those associated to Hamiltonian systems, since, from (16.2), we find

$$\mathrm{div} f = \frac{\partial}{\partial q_1}\frac{\partial H}{\partial p_1} + \frac{\partial}{\partial q_2}\frac{\partial H}{\partial p_2} + \ldots + \frac{\partial}{\partial q_k}\frac{\partial H}{\partial p_k} - \frac{\partial}{\partial p_1}\frac{\partial H}{\partial q_1} - \frac{\partial}{\partial p_2}\frac{\partial H}{\partial q_2} - \ldots - \frac{\partial}{\partial p_k}\frac{\partial H}{\partial q_k}$$

which vanishes due to equality of mixed partial derivatives.

The remarkable feature of divergence-free vector fields is that the flow maps associated to these vector fields are volume preserving.

### 16.2.1 Volume-preserving flows and Liouville's theorem

Recall the change of variables theorem tells us that, if we are given a map $\Phi : \mathbf{R}^d \to \mathbf{R}^d$, and a suitable domain $V \subset \mathbf{R}^d$, then

$$\mathrm{vol}\ V = \int_V d\mathbf{x}, \qquad \mathrm{vol}\ \Phi(V) = \int_V \left|\det \frac{\partial \Phi}{\partial \mathbf{x}}\right| d\mathbf{x} = \int_V |\det M|\ d\mathbf{x}$$

where $M = \Phi'$ is the Jacobian of $\Phi$. It follows that the map $\Phi$ preserves volume provided $|\det M| = 1$. We would like to examine the Jacobian determinant of the flow map of a system. The flow map itself satisfies

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

On each side of this equation, we have a vector valued function of $y$ and $t$; compute the Jacobian matrix of each side, and swap the order of differentiation with respect to $t$ and $y$:

$$\frac{d}{dt}\Phi_t'(y) = f'(\Phi_t(y))\Phi_t'(y)$$

Or $dM/dt = f'(\Phi_t(y))M$. Assuming $M$ is invertible, multiply on the right on both sides by $M^{-1}$. Compute the trace of each side:

$$\mathrm{tr}(\dot{M}M^{-1}) = \mathrm{tr}[f'(\Phi_t(y))]$$

It is a short exercise to show that $\mathrm{tr} f' = \mathrm{div} f$. *Jacobi's formula* for the derivative of a determinant gives

$$\mathrm{tr}(\dot{M}M^{-1}) = \frac{\frac{d}{dt}\det M}{\det M}$$

Since $M(0) = \Phi_0'(y) = I$, it follows that

$$\det \Phi_t'(y) \equiv 1$$

Thus we have proved the following theorem:

**Theorem 16.2.1 (Liouville's Theorem)** *Let a vector field $f$ be divergence free. Then $\Phi_t$ is a volume preserving map (for all $t$).*

In particular, all Hamiltonian flow maps preserve volume in phase space. This is a sort of (qualitative) invariant property, but not a first integral.

### 16.2.2 Volume-preserving numerical methods

We now ask if there are numerical methods that preserve volume in phase space, i.e. mimicking the corresponding property for the flow map.

The obvious requirement for such a numerical method $\Psi_h$ is that

$$\mathrm{div} f = 0 \Rightarrow \det(\Psi_h') = 1.$$

Let us start with Euler's method, $y_{n+1} = y_n + hf(y_n)$. The Jacobian matrix of the flow map is

$$\Psi_h' = I + hf'(y_n)$$

which only has unit determinant in extraordinary situations. For example, the determinant is one for Euler's method applied to the following example:

$$\begin{aligned} \frac{dx}{dt} &= 0 \\ \frac{du}{dt} &= f(x) \end{aligned}$$

However, it is *not* one when Euler's method is applied to the harmonic oscillator:

$$\begin{aligned} \frac{dx}{dt} &= u \\ \frac{du}{dt} &= -x \end{aligned}$$

On the other hand, there are certain methods that do conserve volume, sometimes under special conditions.

## 16.3 Symplectic Structure

It can be shown that Hamiltonian flows possess an even deeper property than volume preservation: the flow map is a symplectic map. Given any two vectors $\xi$ and $\eta$ in $\mathbb{R}^{2d}$, define the quantity

$$\omega_i(\xi, \eta) = \xi_i \eta_{d+i} - \xi_{d+i} \eta_i.$$

This quantity can be given a geometric interpretation. The vectors $\xi$ and $\eta$ span a 2-dimensional parallelogram in $\mathbb{R}^{2d}$. The quantity $\omega_i$ is the *oriented area of the orthogonal projection of this parallelogram on the* $(q_i, p_i)$ *plane*. Here, the word *oriented* means that the sign of $\omega_i$ may be either positive or negative, based on the 'right-hand rule' convention for vector multiplication.

Next, define the bilinear form

$$\omega(\xi, \eta) = \sum_{i=1}^{d} \omega_i(\xi, \eta).$$

which is the sum of the projected oriented areas of $\xi$ and $\eta$. This form can also be written, making use of the matrix $J$ from (16.3), as

$$\omega(\xi, \eta) = \xi^T J \eta$$

A *symplectic matrix* $A$ is one that preserves the bilinear form $\omega$ in the sense that

$$\omega(A\xi, A\omega) = \omega(\xi, \omega).$$

In other words,

$$\xi^T A^T J A \eta = \xi^T J \eta.$$

Since this must hold for arbitrary vectors $\xi$ and $\eta$ in $\mathbb{R}^2 d$, it follows that the condition for a matrix to be symplectic is

$$A^T J A = J.$$

Similarly, a map $g(y) : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ is said to be a *symplectic transformation* if its Jacobian $g'(y)$ satisfies

$$g'(y)^T J g'(y) = J.$$

The fundamental property of Hamiltonian systems is that the flow map is a symplectic transformation:

$$(\Phi_t')^T J \Phi_t' = J$$

Such a map is said to be *symplectic.* To see that this holds, recall that, with $M = \Phi_t'$,

$$\dot{M} = f'(\Phi_t(y))M$$

In the case of a Hamiltonian system, we have

$$f(y) = J \nabla H(y)$$

hence

$$f'(y) = J H_{yy},$$

with $H_{yy}$ the (symmetric) Hessian matrix of second derivatives $\frac{\partial^2 H}{\partial y_i \partial y_j}$, and

$$\dot{M} = J H_{yy} M$$

Differentiate the matrix

$$R = M^T J M$$

with respect to time:

$$\dot{R} = \dot{M}^T J M + M^T J \dot{M}$$

so

$$\dot{R} = M^T H_{yy} J^T J M + M^T J J H_{yy} M$$

Noting that $J^2 = -I$ and $J^T J = I$, we have

$$\dot{R} = M^T H_{yy} M - M^T H_{yy} M = 0.$$

Since $M(0) = \Phi_0'(y) = I$, we must have

$$R(0) = M(0)^T J M(0) = J$$

Since the matrix $R$ is constant for all time, we must have the desired identity

$$M^T J M = J.$$

## 16.4 Symplectic Integrators

It is possible to verify by direct computation that certain numerical methods are symplectic whenever the differential equation being solved is Hamiltonian; in general this means checking that

$$(\Psi_h')^T J \Psi_h' = J.$$

Within the class of Runge-Kutta methods, it can be shown that the subclass of symplectic methods is equivalent to the subsclass of methods that conserve arbitrary quadratic invariants. That is, the methods which satisfy (14.2). In particular, the implicit midpoint rule is symplectic.

An especially fruitful means of constructing symplectic integrators is considered next.

### 16.4.1 Symplectic splitting methods

The idea of splitting has many practical uses. In some instances it allows us to construct methods with some desirable property. In this section we illustrate this for Hamiltonian systems, where splitting may be used to construct symplectic integrators.

Consider a Hamiltonian system

$$\frac{dy}{dt} = J\nabla H(y), \qquad H(y) = H_1(y) + H_2(y)$$

and suppose the flows

$$\frac{dy}{dt} = J\nabla H_1(y), \qquad \frac{dy}{dt} = J\nabla H_2(y)$$

can be exactly integrated. Define the corresponding flow maps $\phi_{t,1}(y)$ and $\phi_{t,2}(y)$. Since the exact solution of a Hamiltonian system defines a symplectic map, we have

$$\phi_{t,1}'(y)^T J \phi_{t,1}'(y) = J, \qquad \phi_{t,2}'(y)^T J \phi_{t,2}'(y) = J.$$

Next consider the numerical method defined by composing these two exact flow maps:

$$\psi_h := \phi_{t,2} \circ \phi_{t,1}$$

This map is also symplectic, since

$$\begin{aligned}\psi_h'(y)^T J \psi_h'(y)^T &= (\phi_{t,2}'(y^*)\phi_{t,1}'(y))^T J \phi_{t,2}'(y^*)\phi_{t,1}'(y) \\ &= \phi_{t,1}'(y)^T \phi_{t,2}'(y^*)^T J \phi_{t,2}'(y^*)\phi_{t,1}'(y) \\ &= \phi_{t,1}'(y)^T J \phi_{t,1}'(y) = J.\end{aligned}$$

That is, the composition of symplectic maps is again a symplectic map.
**Example.** A *separable* Hamiltonian system is one for which $H(q,p) = T(p) + U(q)$. For this case, the splitting method based on splitting the Hamiltonian into kinetic and potential energy terms is given by

$$q_{n+1} = q_n + hT'(p_n), \qquad p_{n+1} = p_n + hU'(q_{n+1}).$$

This scheme is referred to as *symplectic Euler*.

The second order, symmetric variant is called the *Störmer-Verlet method*

$$q_{n+1/2} = q_n + \frac{h}{2}T'(p_n), \quad p_{n+1} = p_n + hU'(q_{n+1/2}), \quad q_{n+1} = q_{n+1/2} + \frac{h}{2}T'(p_{n+1}).$$

## 16.5 Exercises

1. Show that the Harmonic oscillator and the Kepler problem are all expressible as Hamiltonian systems. [ Hint: in each case, start by reformulating the differential equations in terms of positions and momenta, by using the relation $p = mv$ where $m$ is mass and $v$ is velocity.]

2. Write the differential equations for the N-body problem.

# Chapter 17

# Backward Error Analysis

**Main concepts:** Example with forward euler and arbitrary f. Forward, backward, symplectic Euler and pendulum. Symplectic integrators.

Up to this point, the error analysis in this course has considered the difference between the numerical sequence $y_n$, $n = 0, 1, \ldots$ and the exact solution $y(t_n)$ of the differential equation (1.3) passing through either the initial point $y(0) = y_0$ (global error) or through an earlier point on the numerical solution, $y(t_m) = y_m$, $m < n$ (residual or local error). This approach to error analysis yields quantitative statements about the solution, but does not tell us much about the qualitative behavior of a method.

## 17.1 A Modified Differential Equation

In this section we introduce an alternative approach to error analysis, in which we attempt to find a different differential equation

$$\tilde{y}' = \tilde{f}(\tilde{y}) \tag{17.1}$$

for which the numerical solution is exact: $y_n = \tilde{y}(t_n)$. This approach is called *backward error analysis*, and the differential equation (17.1) is called the *modified equation*.

### 17.1.1 A modified equation

Let us procede with an example. Since for any convergent method, we should have $\tilde{f} \to f$ as $h \to 0$, let us try to find a vector function $f_1(y)$ such that

$$\tilde{f}(\tilde{y}) := f(\tilde{y}) + h f_1(\tilde{y}) \tag{17.2}$$

more accurately 'approximates' the numerical solution generated by forward Euler(2.1).

The solution to the modified equation (17.1) for this $\tilde{f}$ over a step of size $h$, can be expanded in a Taylor series. Denoting $\tilde{y}_n \equiv \tilde{y}(t_n)$ and the Jacobian of $f$ by $f'(y)$,

$$\tilde{y}_{n+1} = \tilde{y}_n + h\left(f(\tilde{y}_n) + hf_1(\tilde{y}_n)\right) + \frac{h^2}{2}\left(f'(\tilde{y}_n) + hf_1'(\tilde{y}_n)\right)\tilde{f}(\tilde{y}_n) + \mathcal{O}(h^3)$$

$$\tilde{y}_{n+1} = \tilde{y}_n + h\left(f(\tilde{y}_n) + hf_1(\tilde{y}_n)\right) + \frac{h^2}{2}\left(f'(\tilde{y}_n) + hf_1'(\tilde{y}_n)\right)\left(f(\tilde{y}_n) + hf_1(\tilde{y}_n)\right) + \mathcal{O}(h^3)$$

$$\tilde{y}_{n+1} = \tilde{y}_n + hf(\tilde{y}_n) + h^2\left[f_1(\tilde{y}_n) + \frac{1}{2}f'(\tilde{y}_n)f(\tilde{y}_n)\right] + \mathcal{O}(h^3) \tag{17.3}$$

For forward Euler, we have

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

so, by choosing the term in brackets in (17.3) to be zero, i.e.

$$f_1(\tilde{y}) = -\frac{1}{2}f'(\tilde{y})f(\tilde{y}), \tag{17.4}$$

we see that forward Euler applied to (1.3), while being only a first order accurate approximation to the original ODE, is actually producing a *second order* accurate approximation to the modified system (17.1), (17.2), (17.4).

We can expect therefore, that statements made about the modified equation (17.1) with this choice of $\tilde{f}$ will give us additional insight into the qualitative behavior of the forward Euler method, more so than statements made about the original problem (1.3).

**Example.** Consider the initial value problem

$$y' = y^2, \quad y(0) = 1$$

with exact solution

$$y(t) = \frac{1}{1-t}.$$

The exact solution exists only for $t < 1$. On the other hand, if we apply forward Euler to this problem, the sequence

$$y_{n+1} = y_n + hy_n^2$$

does exist for all time (On a computer, one would eventually get an overflow error).

The modified differential equation is

$$\tilde{y}' = f(\tilde{y}) - \frac{h}{2}f'(\tilde{y})f(\tilde{y}) = \tilde{y}^2 - h\tilde{y}^3.$$

This system has an unstable equilibrium at $y = 0$ and an asymptotically stable equilibrium at $y = 1/h$. In particular, a solution exists for all time.

Figure 17.1 shows the exact solution, the forward Euler solution and the modified equation solution for $h = 0.1$. The modified equation is not much closer to the numerical solution than the exact solution is, but it does exist for all time.

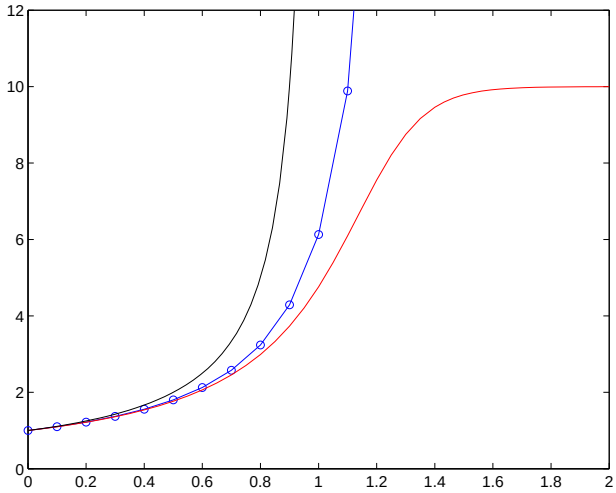



Figure 17.1: Exact (black), forward Euler (blue), and 1-term modified equation solutions to $y' = y^2$, $y(0) = 1$.

### 17.1.2 More modified equations

To get even more accurate modified equations, we can include higher order perturbations in (17.2) such that the numerical method becomes a better and better approximation. The $m$-term modified equation is

$$\tilde{f}(\tilde{y}) := f(\tilde{y}) + hf_1(\tilde{y}) + h^2 f_2(\tilde{y}) + \cdots + h^2 f_m(\tilde{y}) \tag{17.5}$$

We may also talk about the modified equation for $m \to \infty$, formally, without any specification of whether or not the series converges. In fact, we have a hierarchy of modified equations, for $m = 1, 2, \ldots$.

Let us introduce some notation. We denote the Jacobian of $f(y)$ by $f'$, the (symmetric) bilinear form obtained by differentiating $f'(y)$ with respect to $y$ by $f''(\cdot, \cdot)$, etc. Then,

$$\begin{aligned}
\frac{d}{dt} f(y(t)) &= f'f \\
\frac{d^2}{dt^2} f(y(t)) &= f''(f, f) + f'f'f \\
\frac{d^3}{dt^3} f(y(t)) &= f'''(f, f, f) + 3f''(f'f, f) + f'f''(f, f) + f'f'f'f. \\
&\vdots
\end{aligned}$$

We procede to determine $f_2$. We write

$$\tilde{y}(t_{n+1}) = \tilde{y}(t_n) + h\tilde{f} + \frac{h^2}{2}\frac{d}{dt}\tilde{f} + \frac{h^3}{3!}\frac{d^2}{dt^2}\tilde{f}, + \mathcal{O}(h^4), \tag{17.6}$$

where all terms on the right side are evaluated at $t_n$. Then, since

$$\begin{aligned}\frac{d}{dt}\tilde{f} &= f'\tilde{f} + hf_1'\tilde{f} + \mathcal{O}(h^2) \\ &= f'f + hf'f_1 + hf_1'f + \mathcal{O}(h^2)\end{aligned}$$

and

$$\begin{aligned}\frac{d^2}{dt^2}\tilde{f} &= f''(\tilde{f},\tilde{f}) + f'\tilde{f}'\tilde{f} + \mathcal{O}(h) \\ &= f''(f,f) + f'f'f + \mathcal{O}(h),\end{aligned}$$

(17.6) becomes

$$\begin{aligned}\tilde{y}(t_{n+1}) &= \tilde{y}(t_n) + h\left[f + hf_1 + h^2 f_2\right] + \frac{h^2}{2}\left[f'f + hf'f_1 + hf_1'f\right] + \frac{h^3}{3!}\left[f''(f,f) + f'f'f\right] + \mathcal{O}(h^4) \\ &= \tilde{y}(t_n) + hf + h^2\left[f_1 + \frac{1}{2}f'f\right] + h^3\left[f_2 + \frac{1}{2}f'f_1 + \frac{1}{2}f_1'f + \frac{1}{6}f''(f,f) + \frac{1}{6}f'f'f\right] + \mathcal{O}(h^4).\end{aligned} \tag{17.7}$$

The modified equation agrees with the iterates of forward Euler if both terms in brackets above are equal to zero. The first term just gives (17.4). The second term can be solved for $f_2$ to give

$$f_2 = -\frac{1}{2}(f'f_1 + f_1'f) - \frac{1}{6}\left[f''(f,f) + f'f'f\right]. \tag{17.8}$$

**Example.** Continuing the previous example, by substituing $f = \tilde{y}^2$ and $f_1 = -\tilde{y}^3$ into (17.8) the second term is

$$f_2 = \frac{3}{2}\tilde{y}^4.$$

Continuing the procedure outlined above to determine higher order terms (and making use of symbolic mathematics software) gives the five-term modified equation

$$\tilde{y}' = \tilde{y}^2 - h\tilde{y}^3 + h^2\frac{3}{2}\tilde{y}^4 - h^3\frac{8}{3}\tilde{y}^5 + h^4\frac{31}{6}\tilde{y}^6 - h^4\frac{157}{15}\tilde{y}^7.$$

The $m$-term modified equations for $m = 1, \ldots, 5$ are plotted in Figure 17.2 for $h = 0.05$ and $h = 0.02$. The modified equations with $m$ odd each have an attracting fixed point proportional to $1/h$. Note in the figure that the modified equation solutions 'converge' to the numerical solution very quickly as $h$ become smaller.

**Example.** As a second example consider the nonlinear pendulum. Let $q$ be the angle, measured clockwise from the vertical at the lowest point, and let $p = \dot{q}$. We assume a scaling of variables such that mass and gravitational acceleration are equal to one. The equations of motion are

$$\begin{aligned}\dot{q} &= p \\ \dot{p} &= -\sin q.\end{aligned}$$

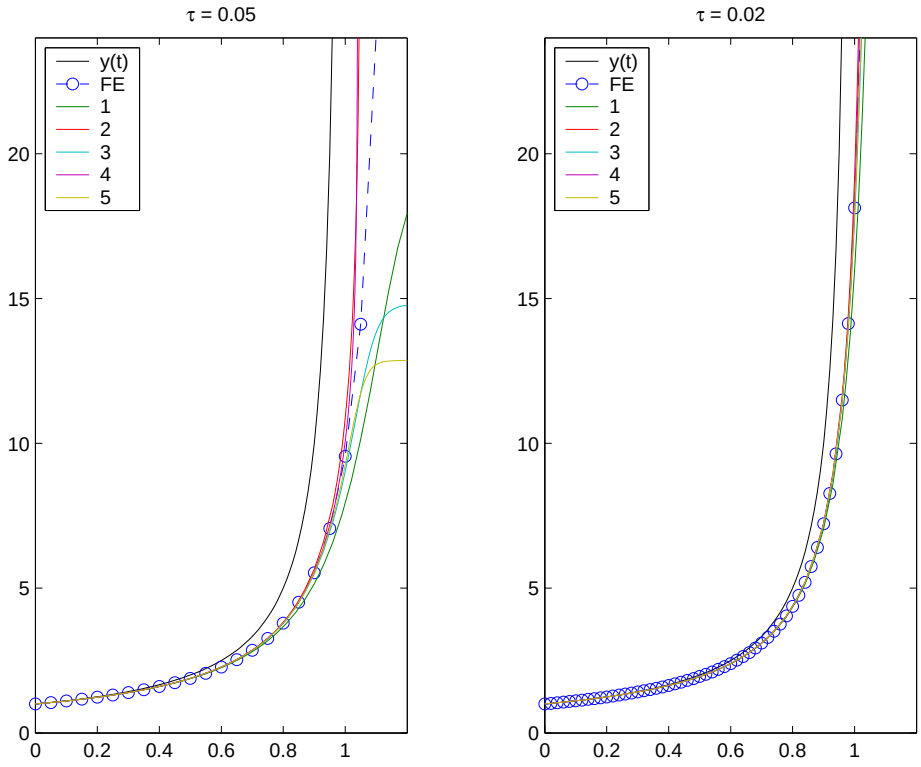



Figure 17.2: Forward Euler and $m$-term modified equation solutions to $y' = y^2$, $y(0) = 1$. On the left $h = 0.05$ and on the right $h = 0.02$.

And the system conserves the total energy

$$H = \frac{p^2}{2} - \cos q. \tag{17.9}$$

We will compare three methods for this problem, forward Euler, backward Euler, and the following method, which we refer to as *symplectic Euler*

$$\begin{aligned} q_{n+1} &= q_n + hp_n \\ p_{n+1} &= p_n - h \sin q_{n+1}. \end{aligned}$$

Note that this method can be computed explicitly even though $q$ is evaluated at time level $n+1$ on the right. The method is first order accurate.

The solutions obtained with all three methods are plotted in Figure 17.3. The black curves are level curves of the energy (17.9). Most solutions are periodic; the pendulum either swings back and forth or around and around. There is a center equilibrium point at the bottom of its arc and a saddle point at the top. The saddle points are connected by separatrices.

The numerical solutions with all three methods were computing using stepsize $h = 0.1$ for 200 steps, with initial condition $q(0) = -3\pi/4$, $p(0) = 0$. The exact solution through

this is point is a closed energy level set, the pendulum swings back and forth. The forward Euler solution is seen to spiral outwards, gaining speed until it crosses the separatrix. The backward Euler solution spirals inward towards the center equilibrium point. The symplectic Euler solution appears to be a closed preiodic orbit. What does backward error analysis tell us about this problem?

The Jacobian for the pendulum is

$$f'(q,p) = \begin{bmatrix} 0 & 1 \\ -\cos q & 0 \end{bmatrix}.$$

The $\mathcal{O}(h)$ perturbation in the modified vector field for the forward Euler method is

$$f_1 = -\frac{1}{2} f' f = \begin{pmatrix} -\frac{1}{2}\sin q \\ -\frac{1}{2} p \cos q \end{pmatrix}.$$

This vector field is plotted in the leftmost frame of Figure 17.4. Note that the component of the vector field that is normal to the level sets of constant energy is everywhere pointing outwards. The center point becomes an unstable equilbrium. The effect of this perturbation is an increase in energy along each trajectory, which is consistent with the behavior exibited by forward Euler.

It may come as no surprise that the $\mathcal{O}(h)$ perturbation of backward Euler is just the negative of that of forward Euler, as was also the case for the principle term in the local error of backward Euler. We have

$$f_1 = \frac{1}{2} f' f = \begin{pmatrix} \frac{1}{2}\sin q \\ \frac{1}{2} p \cos q \end{pmatrix},$$

and this vector field is plotted in the second frame of Figure 17.4. In this case the perturbation causes a decrease in energy along each trajectory, and the center point becomes a stable equilibrium.

For the symplectic method, we expand the numerical solution for $p_{n+1}$ about time $t_n$

$$\begin{aligned} q_{n+1} &= q_n = h p_n \\ p_{n+1} &= p_n - h(\sin q_n + h\dot{q}_n \cos q_n + \mathcal{O}(h^2)) \\ &= p_n - h \sin q_n - h^2 p_n \cos q_n + \mathcal{O}(h^3). \end{aligned}$$

The term $f_1$ must to be chosen such that the first term in brackets in (17.7) agrees with this:

$$f_1(q_n, p_n) + \frac{1}{2} f'(q_n, p_n) f(q_n, p_n) = \begin{pmatrix} 0 \\ -p_n \cos q_n \end{pmatrix},$$

that is,

$$f_1(q,p) = \begin{pmatrix} \frac{1}{2}\sin q_n \\ -\frac{1}{2} p_n \cos q_n \end{pmatrix}.$$

The pendulum equations can be written in the form of a skew-symmetric matrix times a gradient, as in the previous section, to make it more explicit that (17.9) is conserved. Let $y = (q, p)^T$; then

$$y' = J\nabla H(y), \quad J := \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}.$$

It is interesting that the modified vector field $f_1$ can also be written as a product of $J$ and a gradient, namely,

$$f_1 = J\nabla H_1(y), \quad H_1(q,p) = \frac{1}{2} p \sin q$$

It follows that the 1-term modified equation is

$$\tilde{y}' = J\nabla \left( H(\tilde{y}) + hH_1(\tilde{y}) \right),$$

which also has a conserved quantity $\tilde{H} = H + hH_1$. In particular, this means that the 1-term modified equation also has periodic solutions. The level sets of $\tilde{H}$ are plotted in the third frame of Figure 17.4, using a larger stepsize perturbation $h = 0.2$ to exaggerate the effect.

Note that $\tilde{H}$ is *not* a conserved quantitity of the numerical method, since we have only used a single term in the modified equation expansion. However, since the symplectic Euler method is a second order accurate approximation to this modified equation, we would expect that $\tilde{H}$ to be better preserved than $H$.

We could continue to develop higher and higher order modified equations for the nonlinear pendulum, and we would find that *each of them can be written as a product of $J$ and a gradient.* In other words, the symplectic Euler method has a modified equation of the form

$$\tilde{y}' = J\nabla \left( H(\tilde{y}) + hH_1(\tilde{y}) + h^2 H_2(\tilde{y}) + \cdots \right).$$

For this problem, this result suggests that the numerical solution lies on a closed loop in phase space.

## 17.2 Good News and Bad News

In this section we give some of the theory of backward error analysis, without proofs.

First the bad news: the sequence of modified equations (17.5) for $m = 1, 2, \ldots$ *diverges* in general. This is suggested by the left frame in Figure 17.2, where for $m$ odd, the modified equations have asymptotic fixed points which are decreasing in magnitude.

Next the good news: there exists an optimal value of $m$, dependent on $h$ and denoted $m_h^*$, for which the difference between the $m$-term modified equation and the numerical solution is minimized. $m_h^*$ increases like $1/h$ as $h$ tends to zero, and is usually much larger than the order $p$ of the numerical method. In other words, the modified equations are indeed a useful tool in understanding numerical methods.

Specifically, let $B_R(y_0) = \{y \in \mathbb{C}^d : \|y - y_0\| \leq r\}$.

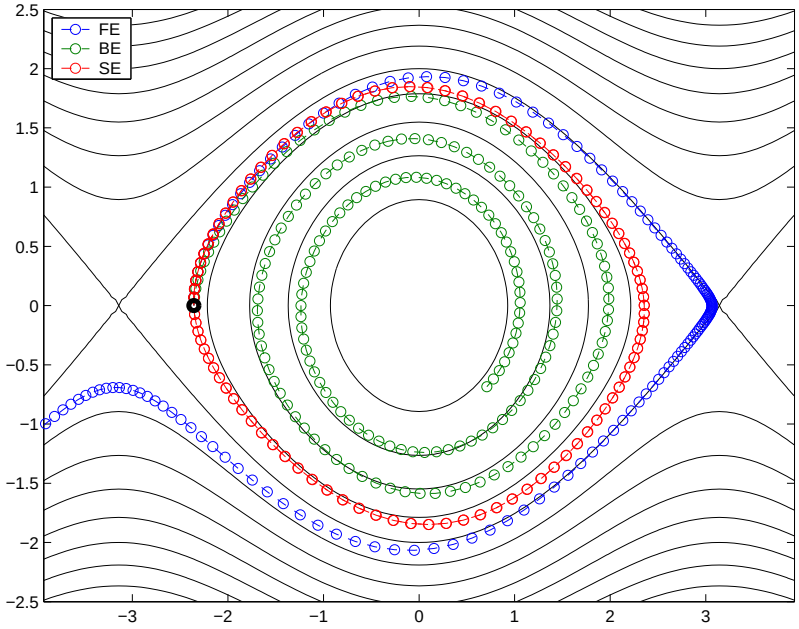



Figure 17.3: Numerical solutions of the nonlinear pendulum superimposed over the energy level sets.

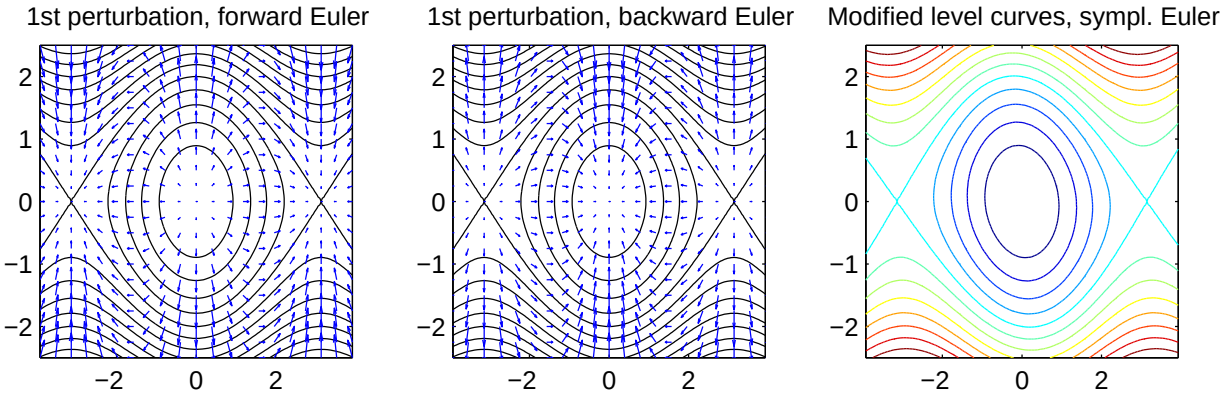



Figure 17.4: Vector field perturbation $f_1$ for forward Euler (left) and backward Euler (center). On the right are the energy level sets of the 1-term modified equation for the symplectic Euler method.

**Theorem 17.2.1** *Suppose $f$ is analytic in $B_{2R}(y_0)$ and satisfies $\|f(y)\| < M$ there. Also suppose the coeffecients of the expansion of the numerical method about $t_n$ is analytic and bounded on $B_R$. Then there exists a constant $\gamma$ dependent only on the method, and a timestep $h_0 \propto R/M$ such that, for $h < h_o/4$, there is $m_h^*$ equal to the largest integer $m$*

*satisfying* $hm \le h_0$*, satisfying*

$$\|\Psi_h y_0 - \Phi_{m_h^*}(h) y_0\| \le h\gamma m_h^* e^{-h_0/h},$$

*where* $\Psi_h$ *is the numerical solution map for a step of size* $h$*, and* $\Phi_{m_h^*}(h)$ *is the exact solution of the* $m_h^*$*-term modified equation (17.5) over an interval of length* $h$*.*

Backward error analysis has been applied successfully to numerical methods that preserve a Lie group, in which case it can be shown by induction that the modified equations preserve the Lie group as well.

A different version of backward error analysis introduces a non-autonomous modified differential equation (effectively increasing the problem dimension by one). This approach has been used to show under what conditions methods applied to dissipative problems and gradient flows will preserve the properties of the analogous continuous solutions. See the book by Stuart & Humphries (Cambridge, 1996).

# Chapter 18

# Dissipative Problems

**Main concepts:**

# Chapter 19

# Gradient flows

**Main concepts:**

## 19.1 Lyapunov Functions

Another type of function we sometimes encounter is termed a *Lyapunov function*, named for its inventor who used it to probe stability in dynamical systems. The Lyapunov function is not a first integral but a function with known growth/decay along solutions. Let $V = V(y)$ be a smooth function defined on $\mathbf{R}^d$ and satisfying $V(0) = 0$. If $V(y) > 0$, $y \neq 0$, we say that $V$ is positive definite. If $V(y) < 0$, $y \neq 0$, we say it is negative definite. If we only have that $V(y) \geq 0$ ($V(y) \leq 0$), we say that $V$ is positive semidefinite (negative semidefinite).

The usefulness of the function $V$ occurs when we know not only something about the definiteness of $V$ but also something about it's monotonicity along solutions. Here is an example of a theorem regarding stability that can be proved using a Lyapunov function.

**Theorem 19.1.1 (Lyapunov Stability Theorem)** *Let $V(y)$ be a positive definite smooth function on $\mathbf{R}^d$. Suppose the origin is an isolated equilibrium point for (??). (a) Suppose $\frac{d}{dt}V(y(t)) \leq 0$ where $y(t)$ is any solution of (??) then the origin is a stable equilibrium point for (??). (b) If, moreover, $\frac{d}{dt}V(y(t)) < 0$ (except possibly at the origin) then the origin is an asymptotically stable equilibrium point.*

**Exercise:** Consider the system

$$\frac{dx}{dt} = u, \tag{19.1}$$

$$\frac{du}{dt} = -x - u^3, \tag{19.2}$$

(a) Sketch the phase portrait
(b) By considering the quantity

$$V(x, u) = x^2 + u^2,$$

can you prove that all solutions tend to the origin by use of Theorem 19.1.1?

# Appendix A

# Implementation Details

## A.1 Solving Nonlinear Systems

There are basically two approaches that are used to solve the nonlinear algebraic equations arising in the implementation of implicit methods. These are *functional iteration* and *Newton iteration*. We discuss both of these below.

**Functional iteration.** The nonlinear systems encountered in implicit methods usually have the form

$$\mathbf{y} = \tau \mathbf{g}(\mathbf{y}) + \mathbf{c} \tag{A.1}$$

for some function $\mathbf{g}$ and constant vector $\mathbf{c}$. For backward Euler, for example, to determine $\mathbf{y}_{n+1}$ we have such a relation with $\mathbf{g} = \mathbf{f}$ and $\mathbf{c} = \mathbf{y}_n$.

Functional iteration for (A.1) is given by

$$\mathbf{y}^{(\nu+1)} := \tau \mathbf{g}(\mathbf{y}^{(\nu)}) + \mathbf{c}. \tag{A.2}$$

It is clear that for $\tau = 0$ this iteration converges in one step to $\mathbf{c}$. Although the nonlinear problem (A.1) may have multiple solutions, if the Jacobian of (A.1), i.e. $I - \tau\, \partial \mathbf{g}/\partial \mathbf{y}$, is nonsingular as $\tau \to 0$, then there is a unique solution for $\tau$ small enough (by the implicit function theorem.)

Convergence of functional iteration requires $\tau \|\partial \mathbf{g}/\partial \mathbf{y}\| < 1$. For stiff problems, $\|\partial \mathbf{g}/\partial \mathbf{y}\|$ is large, so this criterion severely restricts the stepsize in a similar manner as does stability. We have gained nothing. However, for nonstiff problems functional iteration may be acceptable, if one has another reason for using implicit methods.

**Newton iteration.** Given a nonlinear system $\mathbf{G}(\mathbf{y}) = 0$, Newton iteration is

$$\mathbf{y}^{(\nu+1)} := \mathbf{y}^{(\nu)} - \left( \frac{\partial \mathbf{G}}{\partial \mathbf{y}}(\mathbf{y}^{(\nu)}) \right)^{-1} \mathbf{G}(\mathbf{y}^{(\nu)}). \tag{A.3}$$

To solve the system (A.1), we take

$$\mathbf{G}(\mathbf{y}) = \mathbf{y} - \tau \mathbf{g}(\mathbf{y}) - \mathbf{c}, \qquad \partial \mathbf{G}/\partial \mathbf{y} = I - \tau \partial \mathbf{g}/\partial \mathbf{y}.$$

in (A.3). For $\tau$ sufficiently small in magnitude, Newton iteration converges quadratically: if the initial guess $\mathbf{y}^{(0)}$ is close enough to a stationary point $\bar{\mathbf{y}}$ of the iteration (A.3), then there exists a constant $C$ such that

$$\|\mathbf{y}^{(\nu+1)} - \bar{\mathbf{y}}\| \leq C\|\mathbf{y}^{(\nu)} - \bar{\mathbf{y}}\|^2.$$

Note that the 'smallness' of $\tau$ here is much larger than that for functional iteration just discussed. In particular, for linear $\mathbf{G}$, Newton iteration converges in one iteration, regardless of stiffness.

In practice, the evaluation of the Jacobian and its inverse in the iteration (A.3) is prohibitively expensive. Instead, the Jacobian is usually evaluated once per timestep

$$J = \frac{\partial \mathbf{g}}{\partial \mathbf{y}}(\mathbf{y}^{(0)}),$$

where the initial guess $\mathbf{y}^{(0)}$ is, for example, taken to be the solution from the previous time step $\mathbf{y}^{(0)} = \mathbf{y}_n$. The Jacobian can be factored using Gaussian elimination, $I - \tau J = LU$, and used repeatedly in (A.3). The resulting method is sometime called 'modified' Newton iteration. The computational cost of Gaussian elimination is $\mathcal{O}(d^3)$ whereas the solution of the systems with triangular matrices $L$ and $U$ can be done with a cost of $\mathcal{O}(d^2)$ for each system. So re-using the Jacobian results in a factor $d$ savings in computational effort in succeeding iterations, assuming the convergence is unaffected.

Unfortunately the convergence *is* affected. In fact, the the modified Newton method is simply a functional iteration (A.2), with $\mathbf{g}(\mathbf{y})$ and $\mathbf{c}$ replaced with $(I - \tau J)^{-1}(\mathbf{g}(\mathbf{y}) - J\mathbf{y})$ and $(I - \tau J)^{-1}\mathbf{c}$, respectively. Fortunately the reduction in computational expense by not evaluating and factoring the Jacobian in each step usually outweighs the increase in expense due to slower convergence rate, and furthermore the inverse of the Jacobian tends to cancel the effect of stiffness.

Since the convergence of (modified) Newton iteration depends on $\tau$, it may be faster to take more steps with a smaller stepsize, in each of which Newton converges in a few iterations, than to take a fewer, large steps in which convergence is sluggish. For this reason, practical codes usually perform a small number (say 10) modified Newton iterations, and if the convergence is not sufficient, throw out the whole step and reduce the stepsize. Clearly this adds additional complexity to the error control procedure we defined earlier.

## A.2 Error Control

### A.2.1 Error control

In this section we follow the approach to extrapolation and error control in Hairer, Nørsett & Wanner (1993).

A constant stepsize $\tau$ may be sufficient for academic studies. However, real industrial-strength ODE software makes use of variable steps to keep the error below a user-specified

tolerance. In this section we develop one approach to stepsize selection based on local error estimation. We define $\tau_n := t_{n+1} - t_n$.

As mentioned before, the residual (**??**) obtained upon substitution of the Taylor expansion of the exact solution into the difference equation (**??**) is the incremental error introduced in the $n$th time step. Put another way, this is the error incurred in a single step of Euler's method for the initial value problem

$$\mathbf{y}' = \mathbf{f}(\mathbf{y}), \quad \mathbf{y}(0) = \mathbf{y}_n, \tag{A.4}$$

The first term of the residual is a good approximation of the incremental error (improving as $\tau \to 0$).

To control the error we first need a good estimate. The first term of the residual, while accurate, may not be available as computer code, and indeed may be a very complicated expression, since it involves higher derivatives of the function $\mathbf{f}$. Instead we will determine a better approximation of the solution of the local problem (A.4).

### Richardson extrapolation

For the problem (A.4), a single step of size $\tau$ with a method of order $p$ leads to a residual

$$\mathbf{r}_1 = \Phi^\tau \mathbf{y}_n - \mathbf{y}_{n+1} = \tau^{p+1}\boldsymbol{\kappa} + \mathcal{O}(\tau^{p+2}) \tag{A.5}$$

Consider instead taking two steps of size $\tau/2$. For the first of these we again have

$$\hat{\mathbf{r}}_{1/2} = \Phi^{\tau/2} \mathbf{y}_n - \hat{\mathbf{y}}_{n+1/2} = \left(\frac{\tau}{2}\right)^{p+1} \boldsymbol{\kappa} + \mathcal{O}(\tau^{p+2}),$$

where the hatted vectors refer to the second process with stepsize $\tau/2$. For the second step we get

$$\hat{\mathbf{r}}_1 = \Phi^{\tau/2} \hat{\mathbf{y}}_{n+1/2} - \hat{\mathbf{y}}_{n+1} = \left(\frac{\tau}{2}\right)^{p+1} \hat{\boldsymbol{\kappa}} + \mathcal{O}(\tau^{p+2}) \tag{A.6}$$

where $\hat{\boldsymbol{\kappa}} = \boldsymbol{\kappa} + \mathcal{O}(\tau)$ is the residual evaluated at $t_{n+1/2}$ instead of $t_n$. Now the total error made in the two steps is due to the propagation of $\hat{\mathbf{r}}_{1/2}$ under the exact solution through $\hat{\mathbf{y}}_{n+1/2}$ plus $\hat{\mathbf{r}}_1$. It can be shown that

$$\Phi^\tau \mathbf{y}_n - \Phi^{\tau/2} \hat{\mathbf{y}}_{n+1/2} = \left(\frac{\tau}{2}\right)^{p+1} \boldsymbol{\kappa} + \mathcal{O}(\tau^{p+2}) \tag{A.7}$$

Adding (A.6) and (A.7) gives

$$\Phi^\tau \mathbf{y}_n - \hat{\mathbf{y}}_{n+1} = 2\left(\frac{\tau}{2}\right)^{p+1} \boldsymbol{\kappa} + \mathcal{O}(\tau^{p+2})$$

The constant $\boldsymbol{\kappa}$ can be eliminated between the above relation and (A.5) to give an $\mathcal{O}(\tau^{p+2})$ approximation to $\mathbf{y}(t_{n+1})$:

$$\mathbf{y}(t_{n+1}) = \frac{2^p \hat{\mathbf{y}}_{n+1} - \mathbf{y}_{n+1}}{2^p - 1} + \mathcal{O}(\tau^{p+2}).$$

### A stepsize selection algorithm

Given the approximate solution $\mathbf{y}_{n+1}$ and a more accurate approximation $\bar{\mathbf{y}}_{n+1}$, obtained using Richardson extrapolation or otherwise, we wish to determine a stepsize for which the error remains within a user-specified tolerance. Specifically we want to check that

$$\|\bar{\mathbf{y}}_{n+1} - \mathbf{y}_{n+1}\| \leq tol,$$

and if this is not so, we reject the result, decrease the stepsize and try again. Defining

$$err = \frac{\|\bar{\mathbf{y}}_{n+1} - \mathbf{y}_{n+1}\|}{tol}, \tag{A.8}$$

the criterion becomes $err \leq 1$. Since we expect $err \approx C\tau^{p+1}$, the optimal stepsize stepsize is given by $C\tau_{\text{opt}}^{p+1} \approx 1$, we obtain an improved stepsize (by eliminating $C$)

$$\tau_{\text{opt}} = \tau / err^{1/(p+1)}.$$

Since rejected steps are expensive, we multiply the last expression by a safety factor *fac*, having a value of 0.8 or 0.9 or so. Furthermore it is recommended to place a limit on how fast the stepsize can grow and decay, giving

$$\tau_{\text{new}} = \tau \cdot \min(facmax, \max(facmin, fac \cdot (1/err)^{1/(p+1)})) \tag{A.9}$$

where $facmax \approx 2$ and $facmin \approx 0.5$.

To apply the above formula in a code, after computing a solution at time $t_n$, the error is estimated using (A.8) and this is compared against 1. If the $err \leq 1$, the step is accepted and (A.9) gives an estimate for the stepsize $\tau_{n+1}$ of the next step. If $err > 1$, the step is rejected, and (A.9) gives an estimate of $\tau_n$ for a new attempt.

**Example.**
We integrate the Van der Pol equation with parameter $\mu = 5$ this time. Figure A.1 shows the global error as a function of the number of steps. The reduction in the number of steps is approximately a factor of 3. This does not mean that the computation is a factor of 3 faster however. In the first place, the Richardson extrapolation involves a second evaluation of the function $\mathbf{f}$. For large, complex systems, this means the expense of a single step approximately doubles compared to the method without stepsize control. Furthermore, any discarded steps are wasted effort, so a real measure of the benefits of error control must take those into account as well.

---

**Code** `vareuler.m` (`http://www.cwi.nl/~jason/numwisk/vareuler.m`)

```
function [Y,T] = vareuler(y0,tau,Tend,fun,param,tol);
%
%  [Y,T] = vareuler(y0,tau,Tend,fun,param,tol);
```

```
%
%  Integrates the function 'yprime = fun (y,t,param)'
%  using Forward Euler.
%
%  y0 is the initial condition
%  tau is an initial stepsize guess
%  Tend is the length of the integration interval
%  tol is an absolute tolerance
%
%  The output is stored columnwise in the array Y
%  The array T contains the temporal grid
%

Y = y0;  %% output array
y = y0;  %% solution at current time level
T = 0;   %% output array
t = 0;   %% current time level

facmin = 0.5;
facmax = 2;
fac = 0.8;


while t<Tend,  %% Note:  we do not know in advance how many steps are needed
   remesh = 1; %% Assume remeshing is necessary (1==true, 0==false)

   while remesh

      %% Euler step
      F = fun(y,param);
      y1 = y + tau * F;

      %% Richardson extrapolation
      ya = y + tau/2 * F;
      yb = ya + tau/2 * fun(ya,param);
      y2 = 2*yb-y1;

      %% Error estimate:
      err = max(abs(y2-y1))/tol;

      if err <= 1
         remesh = 0;  %% remeshing unnecessary, drop out of the loop
         t = t + tau; %% increment time
```

```
        end

        %% Estimate next timestep
        tau = min(Tend-t,tau * min(facmax,max(facmin,fac/sqrt(err))));
    end

    y = y1;

    Y(:,end+1) = y;
    T(end+1) = t;

end
```

---

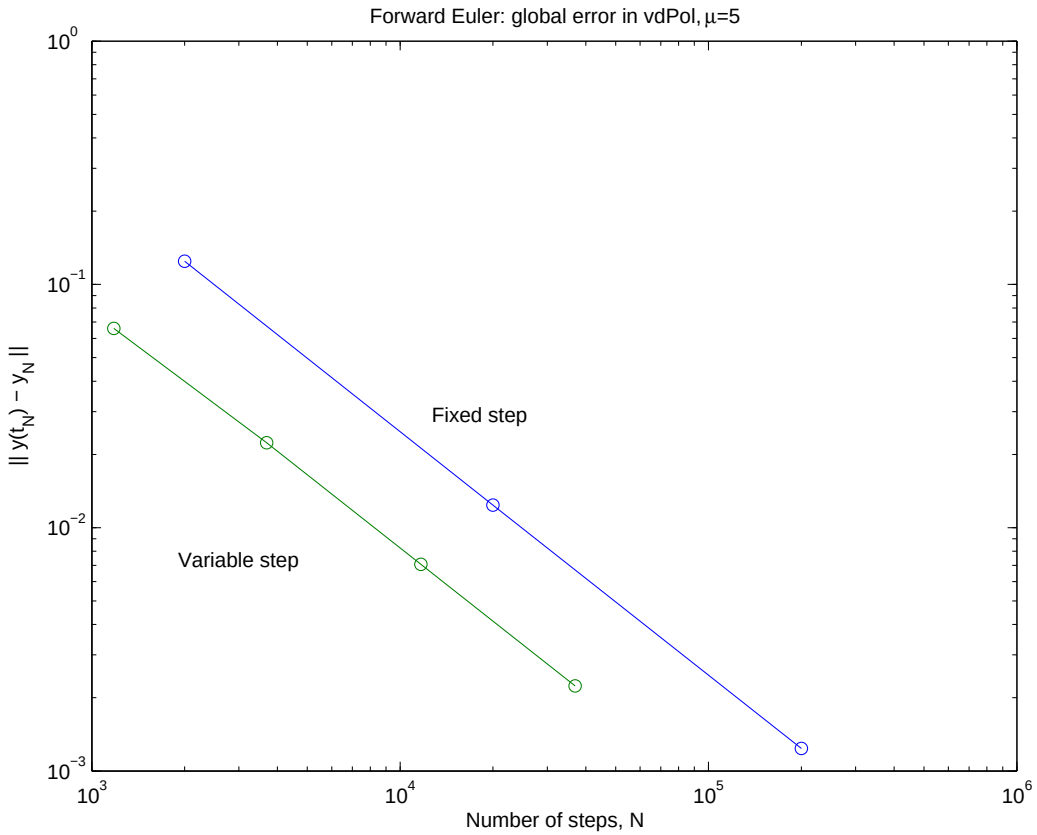



Figure A.1: Solution to Example 1, with $\mu = 5$, using Euler's method with fixed and variable stepsize.