Math 228A: Lecture notes

Numerical solution of (*extra*)*ordinary* differential equations

Fall 2022

M. Lindsey

Latest update: November 2

Contents

Newton has shown that a law is only a necessary relation between the present state of the world and its immediately subsequent state. All the laws discovered since are nothing else; they are in sum, differential equations.

H. Poincaré

Part I
Introduction

1 Systems of ODEs

A general system of ordinary differential equations (ODE) can be written

$$\begin{cases} x'(t) = f(x(t), t), \\ x(0) = x_0, \end{cases} \tag{1.1}$$

whose solution is a continuous function $x : [0, T] \to \mathbb{R}^d$ that satisfies the first equation for all times t in the interval $(0, T)$, together with $x(0) = x_0$, where x_0 is a given *initial condition*. The *state* $x(t)$ at time t is a vector that lives in $\mathbb{R}^d$. (More generally we can consider a nonzero initial time.)

We can alternatively use the Newton dot notation $\frac{dx}{dt}(t) = \dot{x}(t)$ for the time derivative. For the p-th order derivative we can write $\frac{d^p x}{dt^p}(t) = x^{(p)}(t)$.

1.1 Reduction to autonomous case

In (1.1), $f : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ is a function of state and time. If in fact $f(x, t) = g(x)$ for some g, i.e., f does not depend directly on time, then we can write (1.1) as

$$x'(t) = g(x(t)).$$

In this case, the ODE is ***autonomous***, otherwise ***non-autonomous***.

In fact we can always reduce to the autonomous case by adding one dimension to the state variable! To see this, let $y \in \mathbb{R}^{d+1}$ be an augmented state variable. $y_{1:d}$ will correspond to the original state x and y_{d+1} will be an extra variable which simply tracks the progress of time. Then define a function $g : \mathbb{R}^{d+1} \to \mathbb{R}^d$ by

$$g(y) = f(y_{1:d}, y_{d+1})$$

and consider the system

$$\begin{cases} y'(t) = \begin{pmatrix} g(y) \\ 1 \end{pmatrix}, \\ y(0) = \begin{pmatrix} x_0 \\ 0 \end{pmatrix}. \end{cases}$$

Since the last variable solves $y_{d+1}(0) = 0$, $y'_{d+1}(t) = 1$ for all t, we must have $y_{d+1}(t) = t$, and we can recover a solution $x(t)$ of the original system via $x(t) = y_{1:d}(t)$.

1.2 Reduction to first-order case

What about higher-order time derivatives? More generally we could consider a system of the form

$$x^{(p)}(t) = F(x(t), x'(t), x''(t), \ldots, x^{(p-1)}(t), t).$$

For such a system we need initial conditions for each derivative of order less than equation:

$$x(0) = x_0,\ x'(0) = x_0^{(1)},\ \ldots\ ,\ x^{(p-1)}(0) = x_0^{(p-1)}.$$

Actually this can be reduced to the form (1.1).

We will consider the augmented variable $y \in \mathbb{R}^{pd}$ corresponding to the stack of variables

$$y(t) = \begin{pmatrix} x(t) \\ x'(t) \\ \vdots \\ x^{(p-1)}(t) \end{pmatrix}.$$

Correspondingly consider the stacked initial condition

$$y_0 = \begin{pmatrix} x_0 \\ x_0^{(1)} \\ \vdots \\ x_0^{(p-1)} \end{pmatrix},$$

and define

$$f(y,t) = \begin{pmatrix} y_{d+1:2d} \\ y_{2d+1:3d} \\ \vdots \\ y_{(p-1)d+1:pd} \\ F(y_{1:d}, y_{d+1:2d}, \ldots, y_{(p-1)d+1:pd}, t) \end{pmatrix}.$$

Then our solution can be recovered by solving

$$\begin{cases} y'(t) = f(y(t), t), \\ y(0) = y_0. \end{cases}$$

2 Existence and uniqueness: Picard iteration

When does there exist a unique solution to (1.1)? There is a standard sufficient condition. The proof by ***Picard iteration*** is standard issue mathematics that you must know, and it conveys something important about life.

Suppose for a moment that we wanted to solve instead

$$\begin{cases} x'(t) = h(t), \\ x(0) = x_0, \end{cases} \tag{2.1}$$


Figure 2.1: "Someone once told me that time was a predator that stalked us all our lives. But I rather believe that time is a companion who goes with us on the journey and reminds us to cherish every moment...because they'll never come again." Oops, wrong Picard!

where h is some known function. This is easy! The fundamental theorem of calculus guarantees (under mild integrability conditions on h) the existence of a unique solution as

$$x(t) = x_0 + \int_0^t h(s)\,dx. \tag{2.2}$$

Now suppose we already knew the solution $x(t)$. Then we could define

$$h(t) := f(x(t), t) \tag{2.3}$$

and recover $x(t)$ again as the solution of (2.1). Of course this is not yet a very practical insight because it requires us to already know the solution $x(t)$ in order to furnish the solution once again as a solution of (2.1). But it suggests the important insight that the difficulty of solving a general ODE is *self-consistency*, i.e., finding a (or *the*?) function $x : [0, T] \to \mathbb{R}^d$, which maps to a function $h : [0, T] \to \mathbb{R}^d$ via (2.3), which in turn maps back to the same function x via (2.2).

Define Φ to be the composition of the maps (2.3) and (2.2), so Φ is a map $x \mapsto \Phi[x]$ from *functions to functions* defined by

$$\Phi[x](t) = x_0 + \int_0^t f(x(s), s)\,dx. \tag{2.4}$$

The self-consistency discussion is precisely saying that we are looking for *fixed points* of this map, i.e., x such that $\Phi[x] = x$.

In order to proceed we need to be a bit more rigorous and define the domain of Φ, which is going to be a *function space*.

Definition 1. Let $C([0, T]; \mathbb{R}^d)$ denote the space of continuous functions $[0, T] \to \mathbb{R}^d$. Let $\|\cdot\|_\infty$ denote the *uniform* (or *sup*) *norm* on $C([0, T]; \mathbb{R}^d)$, defined by

$$\|g\|_\infty = \sup\{|g(t)| \,:\, t \in [0, T]\}.$$

Here $|\cdot|$ denotes the Euclidean norm on $\mathbb{R}^d$, which will be our convention throughout.

Remark 2. Recall that $C([0, T]; \mathbb{R}^d)$ is a *Banach space* with respect to the uniform norm, i.e., it is a complete metric space with respect to the metric $d(g, h) = \|g - h\|_\infty$.

2.1 Banach fixed point theorem

The key theorem for proving the existence of fixed points is the Banach fixed point theorem. Note that the proof is constructive via an iterative scheme, which even comes with a rate of convergence.

Theorem 3 (Banach fixed point)**.** *Let X denote a complete metric space with metric d. Suppose that $\Phi : X \to X$ is a contraction map, i.e., there exists $\alpha \in [0, 1)$ such that*

$$d(\Phi(x), \Phi(y)) \leq \alpha\, d(x, y)$$

for all $x, y \in X$. Then there exists a unique fixed point for the map Φ, i.e., a unique $x^\star$ such that $\Phi(x^\star) = x^\star$. Moreover, for any $x \in X$, if we define the sequence $\{x_k\}_{k=0}^\infty$ via $x_0 = x$ and $x_k = \Phi(x_{k-1})$ for all $k \geq 1$, then $\lim_{k\to\infty} x_k = x^\star$. In other words, the result of repeated application of Φ converges to $x^\star$ for arbitrary initial input.

Remark 4. In fact, the proof recovers a convergence rate for the limit $x_k \to x^\star$. To wit, we have $d(x_k, x^\star) = O(\alpha^k)$.

Proof. First we show existence. Let $x_0 \in X$ be arbitrary. Then define successively $x_k = \Phi(x_{k-1})$ for $k = 1, 2, \ldots$. We claim that the sequence $\{x_k\}$ converges. Since X is complete, it suffices to show that the sequence is Cauchy. Let $\varepsilon > 0$. We want to show that there exists K sufficiently large such that for any $l > k \geq K$ we have $d(x_k, x_l) \leq \varepsilon$. Note that if $l \geq k \geq K$, then

$$d(x_k, x_l) \leq \sum_{i=k}^{l-1} d(x_i, x_{i+1})$$

by the triangle inequality. Moreover, for any i, by the contraction property we have that

$$d(x_i, x_{i+1}) \leq \alpha\, d(x_{i-1}, x_i) \leq \cdots \leq \alpha^i\, d(x_0, x_1).$$

Therefore

$$d(x_k, x_l) \leq \sum_{i=k}^{l-1} \alpha^i d(x_0, x_1) \leq \alpha^k d(x_0, x_1) \sum_{i=0}^{l-k-1} \alpha^i \leq \frac{\alpha^k}{1-\alpha} d(x_0, x_1) \leq \frac{\alpha^K}{1-\alpha} d(x_0, x_1).$$

The Cauchy claim evidently follows by taking K sufficiently large.

It follows direclty from the definition of continuity that a contraction map is continuous. Now let $x^\star$ be the limit of $\{x_k\}$. Then taking the limit of both sides of $x_k = \Phi(x_{k-1})$, we have by continuity that $x^\star = \Phi(x^\star)$, i.e., the limit point is a fixed point as desired. This establishes existence.

Now suppose there are two fixed points x and x'. We want to show that $x = x'$, from which uniqueness follows. Note that

$$d(x, x') = d(\Phi(x), \Phi(x')) \leq \alpha\, d(x, x'),$$

which can only hold if $d(x, x') = 0$, i.e., $x = x'$. This completes the proof of uniqueness.

The final part of the statement of the theorem follows from uniqueness together with the fact that x_0 was arbitrary in the existence proof. To see the convergence rate claimed in the remark, take the limit as $l \to \infty$ of the inequality $d(x_k, x_l) \leq \frac{\alpha^k}{1-\alpha} d(x_0, x_1)$ proved above. □

2.2 Picard-Lindelöf theorem

The key sufficient condition establishing existence and uniqueness of solutions of (1.1) is phrased in terms of Lipschitz functions.

Definition 5. A function $f : \mathbb{R}^n \to \mathbb{R}$ is *Lipschitz* (or *Lipschitz continuous*) if there exists $L \geq 0$ such that $|f(u) - f(v)| \leq L|u - v|$ for all $u, v \in \mathbb{R}^n$. In this case we can say more specifically that f is L-Lipschitz, and L is a *Lipschitz constant*. (Here $|\cdot|$ denotes the Euclidean norm as always.)

Theorem 6 (Picard-Lindelöf)**.** *Suppose that f is Lipschitz. Then the system (1.1), i.e.,*

$$\begin{cases} x'(t) = f(x(t), t), \\ x(0) = x_0, \end{cases}$$

admits a unique solution on $[0, T]$ for any $T > 0$.

Proof. Suppose that f is L-Lipschitz where $L > 0$. We want to claim that Φ as defined in (2.4) is a contraction map with respect to the uniform norm on $C([0, T]; \mathbb{R}^d)$, but it is not quite true. Indeed, we can compute that:

$$\begin{aligned} \|\Phi[x] - \Phi[y]\|_\infty &= \sup_{t\in[0,T]} \left\{ \left| \int_0^t f(x(s), s) - f(y(s), s)\, dx \right| \right\} \\ &\leq \sup_{t\in[0,T]} \left\{ \int_0^t |f(x(s), s) - f(y(s), s)|\, dx \right\} \\ &\leq \sup_{t\in[0,T]} \left\{ L \int_0^t |x(s) - y(s)|\, dx \right\} \\ &\leq \sup_{t\in[0,T]} \left\{ L \int_0^t \|x - y\|_\infty\, dx \right\} \\ &\leq LT\|x - y\|_\infty. \end{aligned}$$

If we had $LT < 1$, then we would have a contraction map. Unfortunately this cannot be guaranteed a priori.

We sidestep the difficulty in the following way. Let $h = T/N$, where N is sufficiently large such that $Lh < 1$, and consider dividing the interval $[0, T]$ into the N subintervals

$$I_0 := [0, h],\ I_1 := [h, 2h],\ \ldots\ ,\ I_{N-1} := [(N - 1)h, T].$$

We are going to construct a solution for (1.1) on each of these intervals individually and then argue that together they constitute a solution on $[0, T]$.

Indeed let us first define $\Phi_0 : C(I_0; \mathbb{R}^d) \to C(I_0; \mathbb{R}^d)$ via

$$\Phi_0[y](t) = x_0 + \int_0^h f(y(s), s)\, ds.$$

Our preceding calculation ensures that Φ_0 is a contraction mapping, hence admits a unique fixed point in $C(I_0; \mathbb{R}^d)$, which is therefore the unique solution of (1.1) on I_0. As this solution is continuous up to the boundary of $[0, h]$, the final state $x_1 := x(h)$ is well-defined.

Then we can consider x_1 as the initial condition of (1.1) *on the interval* $[h, 2h]$. By shifting the time variable appropriately, the same argument suggests that we can extend x

continuously to the interval $[0, 2h]$ such that x solves (1.1) on both $[0, h]$ and $[h, 2h]$, and we define $x_2 := x(2h)$.

More concretely, given the preceding final state x_n, we inductively define $\Phi_n : C(I_n; \mathbb{R}^d) \to C(I_n; \mathbb{R}^d)$ via

$$\Phi_n[y](t) = x_n + \int_{nh}^{t} f(y(s), s)\, ds.$$

This is a contraction mapping, and we can extend x continuously to $[0, (n+1)h]$ by appending its unique fixed point. Then we define $x_{n+1} = x((n + 1)h)$ to complete the inductive procedure.

In summary the construction yields $x \in C([0, T]; \mathbb{R}^d)$ with $x_n = x(nh)$, solving (1.1) on each of the individual subintervals I_n, $n = 0, \dots, N-1$ in the sense that the restriction $x|_{I_n}$ is the unique fixed point of Φ_n for each n. Note that we do not yet know that x actually solves (1.1) on $[0, T]$, since we have not established that $x'(t) = f(x(t), t)$ is satisfied at the subinterval boundaries $h, 2h, \dots, (N-1)h$. We will do so somewhat indirectly.

Let Φ once again denote the original map $C([0, T]; \mathbb{R}^d) \to C([0, T]; \mathbb{R}^d)$ as in (2.4). Then let $t \in [0, T]$, and let m be the value of n such that $t \in I_n$, defaulting to the smaller value in the case of boundary points.

$$\begin{aligned}
\Phi[x](t) &= x_0 + \int_0^t f(x(s), s)\, dx \\
&= x_0 + \int_{mh}^{t} f(x(s), s)\, ds + \sum_{n=0}^{m-1} \int_{nh}^{(n+1)h} f(x(s), s)\, ds \\
&= x_0 + \int_{mh}^{t} f(x(s), s)\, ds + \sum_{n=0}^{m-1} \left(\left[x_n + \int_{nh}^{(n+1)h} f(x(s), s)\, ds \right] - x_n \right) \\
&\overset{(\star)}{=} x_0 + \int_{mh}^{t} f(x(s), s)\, ds + \sum_{n=0}^{m-1} [x_{n+1} - x_n] \\
&= x_m + \int_{mh}^{t} f(x(s), s)\, ds,
\end{aligned}$$

where in $(\star)$ we have used the fact that $x|_{I_n}$ is the fixed point of of Φ_n for $n = 0, \dots, m-1$ and in the last step have simplified the telescoping sum. But note that the final expression is precisely $\Phi_m[x|_{I_m}](t)$, which is equal to $x(t)$ by the fact that $x|_{I_m}$ is the fixed point of Φ_m. Therefore $\Phi[x](t) = x(t)$, and our candidate $x \in C([0, T]; \mathbb{R}^d)$ is in fact a fixed point of Φ, hence a solution of (1.1) on $[0, T]$.

To see uniqueness, suppose that there is another solution y of (1.1) on $[0, T]$. By the uniqueness of the solution on $[0, h]$, we know that y must agree with x on $[0, h]$. Then by uniqueness of the solution on $[h, 2h]$, since x and y have the same intial condition on this interval, we know that x and y in fact agree on $[0, 2h]$, etc. It follows that x and y coincide on all of $[0, T]$. This completes the proof. □

Remark 7. Note that the most famous counterexample to global-in-time existence, the scalar equation $x'(t) = x(t)^2$, does not satisfy the Lipschitz condition. Solutions of this ODE blow up in finite time, as can be checked by direct solution via separation of variables.

3 Discretization

Unfortunately (1.1) can rarely be solved in closed form. (But when closed-form solutions exist, they're as good as gold!) The main point of this class is to obtain approximate

solutions in the generic unforunate case.

An approximate solution is represented via discretization. We will consider a step size $h = T/N$ and discrete times $0, h, 2h, \ldots, Nh$. We will represent the solution $x : [0, T] \to \mathbb{R}^d$ by its values x_n as these discrete times $t_n := nh$ for $n = 0, \ldots, N$. When N is not fixed, we use the notations $x_n^{(N)}$ and $t_n^{(N)}$ to disambiguate if the meaning is not clear from context.

A numerical scheme is a tractable computational recipe furnishing a collection of states $x_{0:N}^{(N)} = (x_0^{(N)}, \ldots, x_N^{(N)})$ that approximate the true solution states $(x(0), \ldots, x(Nh))$ at our discrete times, i.e., achieving $x_n^{(N)} \approx x(nh)$. Ideally we can control the approximation error, and to have much regard for a scheme at all, it must be the case that the error can be made arbitrarily small by advancing to the limit $N \to \infty$, in the sense that

$$\lim_{N\to\infty} \max_{n=0,\ldots,N} |x_n^{(N)} - x(nh)| = 0.$$

In this case we say that the scheme is ***convergent***.

4 Euler's method

Recall the system (1.1), which we reproduce here for convenience

$$\begin{cases} x'(t) = f(x(t), t), \\ x(0) = x_0. \end{cases} \tag{4.1}$$

The simplest approach to solving it is Euler's method. It is not such a bad old method (and it is in particular convergent), but a large part of the course can be viewed as exploring the ways in which it is not really adequate for all purposes.

4.1 Explicit Euler

Consider the difference quotient approximation of the derivative

$$x'(t_n) \approx \frac{x(t_{n+1}) - x(t_n)}{h}.$$

The first line of (4.1), evaluated at the time $t = t_n$, then suggests

$$\frac{x(t_{n+1}) - x(t_n)}{h} \approx f(x(t_n), t_n), \tag{4.2}$$

or

$$x(t_{n+1}) \approx x(t_n) + hf(x(t_n), t_n).$$

This suggests that we adopt the equation for our discrete approximation

$$x_{n+1} = x_n + hf(x_n, t_n), \quad n = 0, \ldots, N-1, \tag{4.3}$$

which defines ***Euler's method***. (We may alternatively call this the ***explicit Euler method***, by contrast with implicit Euler to be defined later.) Note that x_n determines x_{n+1} uniquely, hence there is a unique solution vector $\mathbf{x}^{(N)}$ solving (4.3).

For concision we will denote

$$f_n := f(x_n, t_n),$$

in which case Euler's method reads as

$$x_{n+1} = x_n + hf_n, \quad n = 0, \ldots, N-1. \tag{4.4}$$

Theorem 8. *Let f be Lipschitz continuous. Then Euler's method is convergent, and moreover* $\max_{n=0,\dots,N} |x_n^{(N)} - x(nh)| = O(h) = O(N^{-1})$.

Remark 9. If we assume that a unique solution exists, for convergence we don't really need to additionally assume that f is Lipschitz. In this case it need only be locally Lipschitz (which follows in particular from being C^1). In the proof, wherever we use the Lipschitz constant for f, it is possible to replace with a local Lipschitz constant on a neighborhood of the true solution of the ODE. For simplicity we just adopt the stronger assumption.

Proof. Let f be L-Lipschitz. By Theorem 6, we know that a unique solution exists for (4.1).

We define at each time an error

$$E_n := |x_n - x(t_n)|.$$

We want to bound the propagation of error from one time step to the next.

Note that

$$x(t_{n+1}) = x(t_n) + \int_{t_n}^{t_{n+1}} f(x(t), t)\, dt,$$

and recall (4.3). Subtracting both equations we obtain

$$\begin{aligned} x_{n+1} - x(t_{n+1}) &= [x_n - x(t_n)] + \left[hf(x_n, t_n) - \int_{t_n}^{t_{n+1}} f(x(t), t)\, dt \right] \\ &= [x_n - x(t_n)] + \int_{t_n}^{t_{n+1}} [f(x_n, t_n) - f(x(t), t)]\, dt. \end{aligned}$$

Now taking the norm of both sides and applying the triangle inequality we obtain

$$E_{n+1} \le E_n + \int_{t_n}^{t_{n+1}} |f(x_n, t_n) - f(x(t), t)|\, dt. \tag{4.5}$$

Then

$$\begin{aligned} |f(x_n, t_n) - f(x(t), t)| &= |f(x_n, t_n) - f(x_n, t) + f(x_n, t) - f(x(t), t)| \\ &\le |f(x_n, t_n) - f(x_n, t)| + |f(x_n, t) - f(x(t), t)| \\ &\le L|t - t_n| + L|x_n - x(t)| \\ &\le Lh + L|x_n - x(t_n) + x(t_n) - x(t)| \\ &\le Lh + L|x_n - x(t_n)| + L|x(t_n) - x(t)| \\ &= Lh + LE_n + L|x(t_n) - x(t)|. \end{aligned}$$

Now we want to get a bound on $|x(t_n) - x(t)|$ that is independent of n. Now

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

so after subtracting $x(t_n)$ from both sides and taking norms, we may derive that

$$|x(t_n) - x(t)| \le Bh,$$

where

$$B := \max_{t \in [0,T]} |f(x(t), t))|.$$

Hence we have derived

$$|f(x_n, t_n) - f(x(t), t)| \leq LE_n + \tilde{B}h,$$

where $\tilde{B} := L(B+1)$. Substituting into (4.5) we obtain

$$E_{n+1} \leq (1 + Lh)E_n + \tilde{B}h^2. \tag{4.6}$$

If we recursively define the sequence c_n, $n = 0, \ldots, N$, via

$$c_{n+1} = (1 + Lh)c_n + \tilde{B}h^2,$$

where $c_n = 0$, we can prove inductively using (4.6) that $E_n \leq c_n$ for all $n = 0, \ldots, N$.

Then by computing c_n explicitly we obtain a bound on the error E_n for all n. For simplicity rewrite

$$c_{n+1} = \alpha c_n + \beta,$$

where $\alpha := 1 + Lh$ and $\beta := \tilde{B}h^2$. Note

$$c_0 = 0,\ c_1 = \beta,\ c_2 = \alpha\beta + \beta,\ c_3 = \alpha^2\beta + \alpha\beta + \beta,\ \ldots$$

and it is easy to show by induction that

$$c_n = \left(\sum_{k=0}^{n-1} \alpha^k \right) \beta,$$

or, summing the finite geometric series,

$$\begin{aligned} c_n &= \frac{\alpha^n - 1}{\alpha - 1}\beta \\ &= \frac{(1 + Lh)^n - 1}{Lh} \cdot \tilde{B}h^2. \end{aligned} \tag{4.7}$$

Now $1 + x \leq e^x$ for all $x \in \mathbb{R}$ (by the convexity of $x \mapsto e^x$) , so

$$E_n \leq c_n \leq \frac{\tilde{B}\left(e^{Lhn} - 1\right)}{L} h \leq (B+1)\left(e^{LhN} - 1\right) h.$$

Now recall $h = T/N$, so

$$E_n \leq (B+1)T\left(e^{LT} - 1\right)\frac{1}{N},$$

and moreover this bound holds for all $n = 0, \ldots, N$. Therefore letting

$$C = (B+1)T\left(e^{LT} - 1\right),$$

we have

$$E_n \leq C/N.$$

In particular, the scheme is convergent. □

4.2 Error estimation and Richardson extrapolation

Note that the proof of Theorem 8 furnishes an explicit error bound on the approximate solution $x^{(N)}_{0:N}$. *In practice this error bound can be extremely pessimistic, although though the order of convergence is sharp*! By order of convergence, we mean the largest exponent p such that the error is $O(h^p) = O(1/N^p)$, and we say for p so defined that a scheme is ***p-th order accurate***. In the case of Euler's method, $p = 1$.

Unfortunately, the optimal preconstant C such that the error at some time t is bounded asymptotically by Ch^p may be very hard to estimate *a priori*. In order to save as much computation as possible, we want to take h only as small as necessary to achieve a desired error tolerance, so getting a sharper estimate, even if *a posteriori*, is quite desirable.

In the sequel we will want to compare our discrete solutions $\mathbf{x}^{(N)}$ solutions for different values of N. Note that the N-th discrete solution is only defined at grid points $(0, h, 2h, \ldots, T)$, where $h = T/N$, so in general for $N' \neq N$, $x^{(N)}_{0:N}$ and $x^{(N')}_{0:N'}$ may not be directly comparable. They can be compared after suitable interpolation to the entire interval $[0, T]$. However, such interpolation is rarely performed in practice, and moreover, care must be taken so that the interpolation preserves the order of accuracy of the scheme! In the simple case of Euler's method, linear interpolation is sufficient to preserve first-order accuracy.

For the purposes of this discussion it is more elegant/convenient to assume that such an interpolation exists (though as we shall see, we will not need to construct it in practice), so for a general p-th order accurate scheme, we assume that we have an interpolant $x^{(N)} : [0, T] \to \mathbb{R}^d$ such that

$$\|x^{(N)} - x\|_\infty = O(1/N^p),$$

where $x(\,\cdot\,)$ is here the true solution. (***In this section, the superscript does not indicate repeated differentiation!***) In particular, we have for every time $t \in [0, T]$,

$$x^{(N)}(t) = x(t) + O(1/N^p).$$

We postulate the more detailed error expansion, consistent with p-th order accuracy:

$$x^{(N)}(t) = x(t) + \frac{C(t)}{N^p} + O(1/N^{p+1}).$$

Plugging in $2N$ in the place of N (in practice, solving the scheme on a grid that is twice as fine), we have

$$x^{(2N)}(t) = x(t) + \frac{C(t)}{2^p N^p} + O(1/N^{p+1}).$$

Then observe that be taking the linear combination

$$\tilde{x}^{(N)}(t) := \frac{2^p x^{(2N)}(t) - x^{(N)}(t)}{2^p - 1} = x(t) + O(1/N^{p+1})$$

we cancel the leading-order contribution to the error.

Then not only is $\tilde{x}^{(N)}$ a more accurate solution, but also we can use it to estimate the error of our original solution $x^{(N)}$. Indeed our error $E^{(N)}$ as a function of time satisfies

$$E^{(N)}(t) := |x^{(N)}(t) - x(t)| = |x^{(N)}(t) - \tilde{x}^{(N)}(t)| + O(1/N^{p+1}),$$

and in the last expression the first term is only $O(1/N^p)$, hence dominates the second term. Therefore the N-point scheme error at time t can be estimated as

$$E^{(N)}(t) \approx |x^{(N)}(t) - \tilde{x}^{(N)}(t)|.$$

In practice, let's fix some N and $h = T/N$ and say we are interested in estimating the accuracy of the solution on the fixed grid $t_n := t_n^{(N)}$, $n = 0, \dots, N$, i.e., on $(0, h, 2h, \dots, T)$. We can do this by *also* solving the ODE on the finer grid $t_n^{(2N)}$, $n = 0, \dots, 2N$, which includes the original grid as a subgrid. Then we can simply perform the above procedure at the grid points t_n instead of the entire interval, defining an ***extrapolated solution***

$$\tilde{x}_n^{(N)} := \frac{2^p x_{2n}^{(2N)} - x_n^{(N)}}{2^p - 1},$$

which is $(p+1)$-th order accurate. This procedure for determining the extrapolated solution is called ***Richardson extrapolation***.

4.3 Implicit Euler

In retrospect, the specification of Euler's method makes a seemingly arbitrary choice in the difference quotient approximation (4.3), which we could alternatively approximate as

$$\frac{x(t_{n+1}) - x(t_n)}{h} \approx f(x(t_{n+1}), t_{n+1}).$$

The resulting scheme

$$x_{n+1} = x_n + hf(x_{n+1}, t_{n+1}), \quad n = 0, \dots, N-1 \tag{4.8}$$

is called the ***implicit*** (or ***backward***) ***Euler method***. The reason it is called ***implicit*** is that given $x_0, \dots, x_n$, *in order to determine x_{n+1} we must solve a system of (possibly nonlinear) equations*, since the right-hand side of (4.8) depends on x_{n+1}.

Note that it is not even obvious a priori that a solution exists! However, concern about this obstacle vanishes as h goes to zero. The general intuition for why this concern vanishes comes from the ***implicit function theorem***. Indeed, observe that if $h = 0$, there exists a (unique) trivial solution $x_{n+1} = x_n$. The implicit function theorem guarantees precisely that if we perturb h by a sufficiently small amount, then we can perturb our solution accordingly to maintain satisfaction of (4.8).

More concretely, rearranging (4.8), we must solve

$$F(h, x) = 0 \tag{4.9}$$

for x, where $F : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ is defined by

$$F(u, x) := x - uf(x, t_n + u) - x_n.$$

Here u is a dummy variable for the step size h, whose value has already been fixed in our discussion. Note that $F(0, x_n) = 0$, i.e., $(0, x_n)$ solves the equation $F(u, x) = 0$. We want to say that as we perturb u away from zero, we can deform x to maintain $F(u, x) = 0$.

Indeed, as long as F is continuously differentiable, the implicit function theorem guarantees that under *one additional condition* (to be specified below), there exists an interval $(-\delta, \delta)$ about 0 and a function $g : (-\delta, \delta) \to \mathbb{R}^d$ with $g(0) = x_n$, which you can think of as mapping a step size u to the corresponding solution x. More concretely, for any $u \in (-\delta, \delta)$, we have $F(u, g(u)) = 0$. The condition is that the Jacobian of F with respect to x is invertible at the solution $(x_n, 0)$, i.e., that

$$D_x F(0, x_n) = \begin{pmatrix} \frac{\partial F}{\partial x_1} & \cdots & \frac{\partial F}{\partial x_n} \end{pmatrix}$$

is invertible. (Note that the partial derivatives $\frac{\partial F}{\partial x_k}$ are vector-valued.) But

$$D_x F(u, x) = I - uD_x f(x, t_n + u),$$

so $D_x F(0, x_n) = I$ is the identity matrix, which is of course invertible.

Therefore, for $0 < h < \delta$, there exists a solution x of (4.9). There is a catch, however! We don't know that we can use the same h for every n. Moreover, as we take h smaller, there infinitely many equations to solve as $N \to \infty$. In order to guarantee a uniform choice of h, one needs to sort through the innards of the implicit function theorem to determine what the choice of δ actually depends on. A sufficient (but certainly not necessary condition) that covers most cases of interest is that f is C^2.

If one adopts the simplifying assumption that f is L-Lipschitz, then the idea of Picard iteration quite easily guarantees the existence of a unique solution to the implicit Euler method for $h < 1/L$. To see this, observe that solving (4.9) for $x \in \mathbb{R}^d$ is equivalent to finding a fixed point of $\phi : \mathbb{R}^d \to \mathbb{R}^d$ defined by

$$\phi(x) := x_n + hf(x, t_n + h).$$

But

$$|\phi(x) - \phi(y)| = h|f(x, t_n + h) - f(y, t_n + h)| \leq Lh|x - y|,$$

so ϕ is a contraction map for $h < 1/L$, as claimed, and there exists a unique solution. This argument, together with a simple modification of our proof of Theorem 8, yields the following.

Theorem 10. *If f is Lipschitz, then for h sufficiently small the implicit Euler method admits a unique solution $x_{0:N}^{(N)}$, which is convergent with $\max_{n=0,\ldots,N} |x_n^{(N)} - x(nh)| = O(h) = O(N^{-1})$.*

Why consider an implicit method as opposed to an explicit one, given the hassle? A large part of the course seeks to address this question, and we defer it for now. We also remark that for any implicit method, considerations regarding the existence of solutions are quite similar to those made above.

5 Taylor series methods

This section is conceptually significant but not practical useful, at least not directly. It is a first attempt to systematically derive schemes of arbitrary orders of accuracy.

In order to derive Euler's method we substituted the difference quotient approximation

$$x'(t_n) = \frac{x(t_{n+1}) - x(t_n)}{h} + O(h),$$

neglecting the $O(h)$ remainder term, into the ODE (4.1). Equivalently we may write

$$x(t_{n+1}) = x(t_n) + hx'(t_n) + O(h^2).$$

The error term in this last expression can be expanded more systematically via Taylor series. For instance, increasing the order of approximation by one, we obtain

$$x(t_{n+1}) = x(t_n) + hx'(t_n) + \frac{h^2}{2}x''(t_n) + O(h^3).$$

The remainder term is called the ***local truncation error*** (***LTE***), which term has slightly different meanings dependent on the context. The general gist is that it is the amount by which the equations for x_n defining our scheme fails to hold when the true solution value $x(t_n)$ at time t_n is plugged in for x_n.

The key in our derivation of Euler's method is that $x'(t_n)$ is known in terms of $x(t_n)$ and t_n alone via the ODE (4.1), which we reproduce here as

$$x'(t) = f(x(t), t). \tag{5.1}$$

To get an expression for $x''(t_n)$ in terms of $x(t_n)$ and t_n alone, we must *differentiate the ODE*. Indeed, differentiating both sides of (5.1) we obtain

$$\begin{aligned} x''(t) &= \nabla_x f(x(t), t)) \cdot x'(t) + \partial_t f(x(t), t) \\ &= \nabla_x f(x(t), t)) \cdot f(x(t), t) + \partial_t f(x(t), t) \end{aligned}$$

Given an analytic expression for f, assuming it is differentiable, we may typically derive analytic expressions for $\nabla_x f$ and $\partial_t f$. Then we can write the ***Taylor series method*** of order 2, or **TS(2)** for short, as

$$x_{n+1} = x_n + hf(x_n, t_n) + \frac{h^2}{2} \left[D_x f(x_n, t_n) \cdot f(x_n, t_n) + \partial_t f(x_n, t_n) \right],$$

where D_x denotes the Jacobian matrix (with respect to the x variables) and the dot indicates matrix-vector multiplication. Note that this is an explicit method.

The general Taylor series method of order p, or **TS(p)**, can be derived similarly by repeated differentiation of (5.1). By differentiating p times, we express the p-th derivative $x^{(p)}(t)$ in terms of $x^{(k)}(t)$ for $k = 0, \ldots, p-1$. One can then recursively replace the derivatives up to $(p-1)$-th order with expressions in terms of derivatives up to $(p-2)$-th order, etc., in order to obtain an expression for $x^{(p)}(t)$ in terms of $x(t)$ alone. It is tedious, however, to write a detailed expression for the scheme.

Abstractly we may write

$$x_{n+1} = x_n + hf(x_n, t_n) + \frac{h^2}{2} f^{(2)}(x_n, t_n) + \cdots + \frac{h^p}{p!} f^{(p)}(x_n, t_n),$$

where the $f^{(k)}$, $k = 2, \ldots, p$ are the appropriate functions, e.g.,

$$f^{(2)}(x, t) := D_x f(x, t) \cdot f(x, t) + \partial_t f(x, t),$$

as derived above. More generally each function $f^{(k)}(x, t)$ satisfies the condition that for the true solution $x(t)$,

$$\frac{d^k}{dt^k} \left[f(x(t), t) \right] = f^{(k+1)}(x(t), t). \tag{5.2}$$

For ease of notation we further define $f^{(1)} := f$, so the TS(p) scheme may be written compactly as

$$x_{n+1} = x_n + \sum_{k=1}^{p} \frac{h^k}{k!} f^{(k)}(x_n, t_n). \tag{5.3}$$

Theorem 11. *Suppose that f is C^p and $f^{(k)}$ is Lipschitz for $k = 1, \ldots, p$. Then* TS(p) *is convergent, and moreover $\|\mathbf{x}^{(N)} - \mathbf{x}^{(N)}_{\text{true}}\|_\infty = O(h^p)$.*

Remark 12. We don't really need to additionally assume that the $f^{(k)}$ are (globally) Lipschitz. Assuming that a unique solution exists, it suffices for them all to be merely locally Lipschitz, which is in particular guaranteed by the assumption that f is C^p. In the proof, wherever we use Lipschitz constants, it is possible to replace them with local Lipschitz constants on a neighborhood of the true solution of the ODE. For simplicity we just adopt the stronger assumption.

Proof. Let L be sufficiently large so that $f^{(k)}$ is L-Lipschitz for $k = 1, \dots, p$. We will reason by analogy to the convergence proof for Euler's method (Theorem 8). Recall the key equation

$$x_{n+1} - x(t_{n+1}) = [x_n - x(t_n)] + \left[hf(x_n, t_n) - \int_{t_n}^{t_{n+1}} f(x(t), t)\, dt \right].$$

in the proof of Euler's method, where the key idea was that the first bracketed term is bounded in terms of E_n and the second is bounded by $(1 + B_1 h)E_n + B_2 h^2$ for some constants B_1, B_2 independent of n.

This time, referring to (5.3), we have instead the analogous equation

$$x_{n+1} - x(t_{n+1}) = [x_n - x(t_n)] + \left[\sum_{k=1}^{p} \frac{h^k}{k!} f^{(k)}(x_n, t_n) - \int_{t_n}^{t_{n+1}} f(x(t), t)\, dt \right], \tag{5.4}$$

and we want to bound the second term by $(1+B_1 h)E_n + B_2 h^{p+1}$ for some constants B_1, B_2. Inside the integral within the second bracketed term, we expand $f(x(t), t)$ by Taylor series, i.e., as

$$f(x(t), t) = \sum_{k=0}^{p-1} \frac{(t - t_n)^k}{k!} \left. \frac{d^k}{dt^k} \left[f(x(t), t) \right] \right|_{t=t_n} + O(h^p),$$

where the $O(h^p)$ term is bounded independently of n and N.

But by the defining property (5.2) of the $f^{(k)}$, we can equivalently write

$$f(x(t), t) = \sum_{k=0}^{p-1} \frac{(t - t_n)^k}{k!} f^{(k+1)}(x(t_n), t_n) + O(h^p),$$

where we additionally define $f^{(1)} := f$. After performing the exact integration

$$\int_{t_n}^{t_{n+1}} (t - t_n)^k\, dt = \frac{(t_{n+1} - t_n)^{k+1}}{(k+1)} = \frac{h^{k+1}}{k+1}$$

and shifting the summation index we have

$$\int_{t_n}^{t_{n+1}} f(x(t), t)\, dt = \sum_{k=1}^{p} \frac{h^k}{k!} f^{(k)}(x(t_n), t_n) + O(h^{p+1}).$$

Then (5.4) can be rewritten as

$$x_{n+1} - x(t_{n+1}) = [x_n - x(t_n)] + \sum_{k=1}^{p} \frac{h^k}{k!} \left[f^{(k)}(x_n, t_n) - f^{(k)}(x(t_n), t_n) \right] + O(h^{p+1}).$$

Taking norms of both sides, using the triangle inequality and the Lipschitz property for each of the $f^{(k)}$, we obtain

$$\begin{aligned} E_{n+1} &\leq E_n + \sum_{k=1}^{p} \frac{h^k}{k!} L E_n + O(h^{p+1}) \\ &= \left(1 + L \sum_{k=1}^{p} \frac{h^k}{k!}\right) E_n + O(h^{p+1}). \end{aligned}$$

Note that for small h (say concretely for $h \leq 1$), there exists some universal $B_1 \geq 0$ such that

$$L \sum_{k=1}^{p} \frac{h^k}{k!} \leq B_1 h,$$

and there exists also some $B_2 \geq 0$ such that then

$$E_{n+1} \leq (1 + B_1 h)\, E_n + B_2 h^{p+1}$$

for all h sufficiently small.

By analogy to (4.7) and the reasoning that follows (with B_1 in the place of L and B_2 in the place of $\tilde{B}$), we nowobtain

$$E_n \leq \frac{(1 + B_1 h)^{n+1} - 1}{B_1 h} \cdot B_2 h^{p+1} \leq C h^p$$

for an appropriate constant C independent of n and N. This completes the proof. □

Taylor series methods have a drawback that is often fatal: they require us to compute all the p-th order partial derivatives of f, which, depending on the practical context may not be readily available or may be very expensive to evaluate. Still they serve as a useful starting point and inspiration for thinking about higher-order schemes.

As the sun eclipses the stars by its brilliancy, so people of knowledge will eclipse the fame of others in assemblies of the public if they propose algebraic problems, and still more if they solve them.

Brahmagupta

Part II
Linear multistep methods

Given the solution $x(t)$ at time t of (1.1), we already encountered two different ways of approximately advancing the solution by one increment h of time. The first was

$$x(t+h) \approx x(t) + hf(x(t), t),$$

which led to the explicit Euler method, and the second was

$$x(t+h) \approx x(t) + hf(x(t+h), t+h),$$

which led to the implicit Euler method.

One can readily imagine a mixture of the two, approaches:

$$x(t+h) \approx x(t) + \frac{1}{2}\left[f(x(t), t) + f(x(t+h), t+h)\right],$$

which in fact has one higher order of accuracy, as we shall see.

The resulting numerical schemes were written compactly as

$$\begin{aligned} x_{n+1} - x_n &= hf_n && \text{(Explicit Euler)} \\ x_{n+1} - x_n &= hf_{n+1} && \text{(Implicit Euler)} \\ x_{n+1} - x_n &= h(f_n + f_{n+1}) && \text{(Trapezoidal rule)} \end{aligned} \tag{5.5}$$

where as always $f_n = f(x_n, t_n)$ implicitly depends on x_n.

Linear multistep methods (***LMMs***) generalize these schemes considerably. The general form of an r***-step*** LMM is

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = h \sum_{j=0}^{r} \beta_j f_{n+j}. \tag{5.6}$$

Assuming that x_m is known for $m = 0, \ldots, n+r-1$, this equation can in principle be solved for the next value x_{n+r}. Note that the schemes in (5.5) are one-step LMMs. We will discuss the necessary construction of ***starting values*** later on.

We will always have $\alpha_r \neq 0$, so by rescaling both sides of (5.6) we can assume without loss of generality that $\alpha_r = 1$ unless otherwise noted.

6 Local truncation error and consistency

In order to check how accurate we expect a method to be, it makes sense to plug the *true solution* $x(t)$ (evaluated at the discrete times t_n) into both sides of (5.6) and measure the size of their discrepancy, which we call the ***local truncation error*** (***LTE***), similarly as in the setting of Taylor series methods. Here as well, an LTE of size $O(h^{p+1})$ will induce a scheme of order p.

To wit, take $x_n := x(t_n)$, and since $x(t)$ solves the ODE (1.1), we have $x'(t_n) = f(x(t_n), t_n) = f_n$. Then the difference of the LHS and RHS of (5.6), namely the LTE, is given by

$$\tau_n := \sum_{j=0}^{r} \alpha_j x(t_{n+j}) - h \sum_{j=0}^{r} \beta_j x'(t_{n+j}).$$

We can define the LTE more generally as a function of continuous time via

$$\tau(t) := \sum_{j=0}^{r} \alpha_j x(t + jh) - h \sum_{j=0}^{r} \beta_j x'(t + jh).$$

More generally we can think of the function $\tau(\,\cdot\,)$ as the result of the application of a linear operator $\mathcal{L}_h$ (fixed for a given scheme) to the solution $x(\,\cdot\,)$, where for a differentiable function z, $\mathscr{L}_h z$ is defined by

$$\mathscr{L}_h z(t) := \sum_{j=0}^{r} \alpha_j z(t + jh) - h \sum_{j=0}^{r} \beta_j z'(t + jh). \tag{6.1}$$

6.1 One-step methods

Let us now consider the simple cases of the one-step LMMs of (5.5).

6.1.1 Explicit Euler

We begin with explicit Euler. In this case

$$\mathscr{L}_h z(t) = z(t + h) - z(t) - hz'(t).$$

The trick for analyzing LTE is to plug in Taylor series about t everywhere we can. Here there is only one opportunity

$$\mathscr{L}_h z(t) = z(t) + hz'(t) + O(h^2) - z(t). - hz'(t) = O(h^2)$$

Note that by the Lagrange error bound, the preconstant in the $O(h^2)$ term is bounded by

$$\sup_{u \in [t, t+h]} \frac{|z''(t)|}{2} h^2 \leq \frac{1}{2} \|z''\|_\infty h^2$$

In general, we will need *one more derivative than our order of accuracy* in order to get a quantitative bound on the LTE. If the order of the scheme is p, a quantative bound can typically be phrased in terms of $\|z^{(p)}\|_\infty$.

By Taylor-expanding to one higher order (assuming an extra derivative), we obtain a more detailed estimate

$$\mathscr{L}_h z(t) = \frac{h^2}{2} z''(t) + O(h^3).$$

From this estimate we can see that we *do not* (in general) have $O(h^3)$ LTE, i.e., $O(h^2)$ is the sharpest possible bound.

We will just assume henceforth that our test function z is smooth and not comment further on how many derivatives are needed.

6.1.2 Implicit Euler

In the case of implicit Euler, we have

$$\mathscr{L}_h z(t) = [z(t+h) - z(t)] - hz'(t+h).$$

Here we must also expand the $z'(t+h)$ term, yielding

$$\begin{aligned}
\mathscr{L}_h z(t) &= \left[hz'(t) + \frac{h^2}{2}z''(t) + O(h^3)\right] - hz'(t+h) \\
&= \left[hz'(t) + \frac{h^2}{2}z''(t) + O(h^3)\right] - h\left[z'(t) + hz''(t) + O(h^2)\right] \\
&= -\frac{h^2}{2}z''(t) + O(h^3).
\end{aligned}$$

From this estimate we can see that the LTE is $O(h^2)$ but not $O(h^3)$ in general.

6.1.3 Trapezoidal rule

For the trapezoidal rule we have

$$\begin{aligned}
\mathscr{L}_h z(t) &= [z(t+h) - z(t)] - \frac{h}{2}z'(t+h) - \frac{h}{2}z'(t) \\
&= \left[hz'(t) + \frac{h^2}{2}z''(t) + O(h^3)\right] - \frac{h}{2}\left[z'(t) + hz''(t) + O(h^2)\right] - \frac{h}{2}z'(t) \\
&= O(h^3).
\end{aligned}$$

More detailed calculation shows that the terms of order h^3 do not cancel.

Therefore in fact the trapezoidal rule is second-order accurate, though it is only a one-step method like explicit/implicit Euler.

6.2 Consistency

We say that an LMM is ***consistent of order*** p or ***order-p consistent*** if $\mathscr{L}_h z(t) = O(h^{p+1})$ for every smooth z. In particular, we say that it is ***consistent*** if it is order-p consistent for some $p \geq 1$.

In general, there exists for every p some formal expansion

$$\mathscr{L}_h z(t) = C_0 z(t) + C_1 hz'(t) + C_2 h^2 z^{(2)}(t) + \cdots,$$

where the constants C_k are independent of z. If an LMM is order-p consistent, then we know that $C_0 = \cdots = C_p = 0$, and

$$\mathscr{L}_h z(t) = C_{p+1} h^{p+1} z^{(p+1)}(t) + O(h^{p+2}),$$

where $C_{p+1} \neq 0$. This nonzero coefficient C_{p+1} is called the ***error constant*** of the LMM. Note for example that the error constants of explicit and implicit Euler are $1/2$ and $-1/2$,

respectively. In general, the error constant can in principle be computed analytically, and it may be useful to do so! Indeed, ***Milne's device*** (to be discussed later) uses error constants to estimate the error of an LMM and possibly improve it.

It is not hard to derive necessary and sufficient conditions for the consistency of (5.6). Recall (6.1) and expand up to $O(h^2)$ error as

$$\begin{aligned}\mathscr{L}_h z(t) &= \sum_{j=0}^{r} \alpha_j z(t+jh) - h\sum_{j=0}^{r} \beta_j z'(t+jh) \\ &= \sum_{j=0}^{r} \alpha_j \left[z(t) + jhz'(t)\right] - h\sum_{j=0}^{r} \beta_j \left[z'(t)\right] + O(h^2) \\ &= \left(\sum_{j=0}^{r} \alpha_j\right) z(t) + h\left(\sum_{j=0}^{r} j\alpha_j - \beta_j\right) z'(t) + O(h^2),\end{aligned}$$

so

$$C_0 = \sum_{j=0}^{r} \alpha_j, \quad C_1 = \sum_{j=0}^{r} \left(j\alpha_j - \beta_j\right), \tag{6.2}$$

and we have consistency if and only if both quantities are zero.

Theorem 13. *An LMM (5.6) is consistent if and only if*

$$\sum_{j=0}^{r} \alpha_j = \sum_{j=0}^{r} \left(j\alpha_j - \beta_j\right) = 0.$$

It is useful for later purposes to define the ***first*** and ***second characteristic polynomials*** of the LMM (5.6) as, respectively

$$\rho(w) = \sum_{j=0}^{r} \alpha_j w^j, \quad \sigma(w) = \sum_{j=0}^{r} \beta_j w^j.$$

We can then see that consistency is equivalent to the conditions that

$$\rho(1) = 0, \quad \rho'(1) = \sigma(1).$$

As we shall later see, ***consistency does not imply convergence!*** There is an additional ***stability*** requirement that we shall treat later.

6.3 Starting values

Note that to solve an r-step LMM, we need r ***starting values*** $x_0, \ldots, x_{r-1}$ in order to initialize the scheme. This is annoying! We are only given $x_0 = x(0)$ from the initial condition of (1.1).

The most naive option is to simply take $x_j = x_0$ for $j = 1, \ldots, r-1$. However, this might sacrifice the order of accuracy of our scheme. Let us try to get some heuristic understanding of what we require.

In general we have $x(t_j) = x_0 + \Theta(h)$ for $j = 0, \ldots, r-1$, since $x(t)$ is differentiable at $t = 0$. Therefore under the naive approach, our error $e_j := x_j - x(t_j)$ is already $\Theta(h)$ by time step $j = r-1$. If our scheme is order-p consistent in the sense that the LTE is $O(h^{p+1})$, we

will accumulate an additional error that is $O(h^{p+1})$ at each of the remaining $O(h^{-1})$ time steps (cf. the proof of Theorem 11, conceptually), i.e., we will accumulate additional error of $O(h^p)$. However, this additional error accumulation is dominated by our $\Theta(h)$ initialization error, and the entire method is really only first-order consistent. If $p = 1$, this is no worse than expected, and we can adopt the naive approach without too much regret.

More generally, the preceding argument suggests that we need to initialize $x_j = x(t_j) + O(h^p)$ in order to fulfill the dream of order-p accuracy for an LMM with $O(h^{p+1})$ LTE. We could determine these intitial values, for example, by running $r-1$ steps of the order-$(p-1)$-accurate Taylor series method TS$(p-1)$. Since the number of steps needed for initialization is independent of the step size h, despite the disadvantages of Taylor series methods this may be a reasonable practical approach. In order to solve the ODE up to time T, we will need to run $\sim h^{-1}$ steps of the LMM, so as $h \to 0$, the initialization cost should be negligible.

Still, it may be more practically efficient to use ***Runge-Kutta methods***, which do not require starting values, for initialization. (Runge-Kutta methods will be the subject of the next unit of the course.)

7 Families of LMMs

At this point we introduce several important families of LMMs that go beyond $r = 1$ to achieve higher orders of accuracy. Later on we will discuss the stability and convergence of these methods, but for now we focus only on the local truncation error.

7.1 Integral-based methods

Recall we are essentially trying to predict $x(t_{n+r})$ based on the values $x(t_{n+j})$ for $j = 0, \ldots, r-1$. Also recall that the solution of (1.1) satisfies

$$x(t_{n+r}) = x(t_{n+r-1}) + \int_{t_{n+r-1}}^{t_{n+r}} f(x(t), t)\, dt. \tag{7.1}$$

Can we use the preceding values $x(t_{n+j})$ to approximate the integral term more accurately? Note that explicit Euler, implicit Euler, and the trapezoidal rule follow, respectively, from applying the left endpoint, right endpoint, and trapezoidal rules for approximating the integral.

In particular, the trapezoidal rule can be interpreted as replacing $f(x(t), t))$ with a linear interpolation over the interval of integration. But to begin with, we will focus on developing a higher-order *explicit* approachg. As such we will first generalize the left endpoint rule. This can be seen as replacing $f(x(t), t)$ with the 'trivial polynomial interpolation' which matches the value at the left endpoint with a constant polynomial.

7.1.1 Adams-Bashforth methods

More generally, we can use all r of the values $f_{n+j} = f\left(x(t_{n+j}), t_{n+j}\right)$, $j = 0, \ldots, r-1$, to construct a more accurate polynomial interpolation over the interval $[t_{n+r-1}, t_{n+r}]$. We expect our interpolation to then achieve $O(h^r)$ accuracy pointwise, so our LTE will be $O(h^{p+1})$, and we will have a method that is order-p consistent. In fact as we shall see the resulting r-step ***Adams-Bashforth method*** will be a LMM, since the interpolation depends linearly on the f_{n+j}.

Note that, provided we can construct the interpolation with such accuracy, we have verified our LTE of $O(h^{r+1})$ by a rather different approach compared to the 'plug in Taylor series and cancel terms' approach suggested above! Indeed, the order of accuracy is ensured by Corollary 61 of Appendix A.

The formula (7.1) suggests that the α_j as in (5.6) for this method are

$$\alpha_r = 1,\ \alpha_{r-1} = -1,\ \alpha_{r-2} = \cdots = \alpha_0 = 0.$$

Then in order to determine the β_j, we can ease the notation by setting $n = 0$. Let

$$\mathbf{t} = (t_0, t_1, \ldots, t_{r-1}) = (0, h, \ldots, (r-1)h),$$

and let $\ell_j(t; \mathbf{t})$ be the Lagrange basis polynomials as in Appendix A:

$$\ell_j(t; \mathbf{t}) = \prod_{i \in \{0,\ldots,r-1\} \setminus \{j\}} \frac{t - ih}{(j-i)h}, \quad j = 0, \ldots, r-1.$$

Note moreover that

$$\prod_{i \in \{0,\ldots,r-1\} \setminus \{j\}} (j - i) = (-1)^{r-1-j} j!(r-1-j)!,$$

so

$$\ell_j(ht; \mathbf{t}) = \frac{(-1)^{r-1-j}}{j!(r-1-j)!} \prod_{i \in \{0,\ldots,r-1\} \setminus \{j\}} (t - i).$$

Then we plug the approximation

$$f(x(t), t) \approx \sum_{j=0}^{r-1} \ell_j(t; \mathbf{t}) f_j$$

into (7.1) to obtain

$$\begin{aligned}
\beta_j &= \frac{1}{h} \int_{(r-1)h}^{rh} \ell_j(t; \mathbf{t})\, dt \\
&= \int_{r-1}^{r} \ell_j(ht; \mathbf{t})\, dt \\
&= \frac{(-1)^{r-1-j}}{j!(r-1-j)!} \int_{r-1}^{r} \prod_{i \in \{0,\ldots,r-1\} \setminus \{j\}} (t - i)\, dt \\
&= \frac{(-1)^{r-1-j}}{j!(r-1-j)!} \int_{r-1}^{r} \prod_{i \in \{0,\ldots,r-1\} \setminus \{r-1-j\}} (t - (r-1-i))\, dt \\
&= \frac{(-1)^{r-1-j}}{j!(r-1-j)!} \int_{0}^{1} \prod_{i \in \{0,\ldots,r-1\} \setminus \{r-1-j\}} (t + i)\, dt,
\end{aligned}$$

or equivalently:

$$\beta_{r-1-j} = \frac{(-1)^j}{j!(r-1-j)!} \int_0^1 \prod_{i \in \{0,\ldots,r-1\} \setminus \{j\}} (t + i)\, dt, \quad j = 0, \ldots, r-1. \tag{7.2}$$

(Note that $\beta_r = 0$, which ensures that the method is explicit).

The integrals in (7.2) can be evaluated analytically, but there does not seem to be a neat formula. Of course, once they are worked out a single time, they never need be computed again!

7.1.2 Adams-Moulton methods

If we allow ourselves to additionally use the right endpoint of our interval of integration (as in the trapezoidal rule), then we can get a polynomial interpolation with one additional order of accuracy. Specifically, the LTE is $O(h^{r+2})$, and the method is now of order $r+1$. However, beware that this so-called ***Adams-Moulton method*** is now implicit!

Here once again

$$\alpha_r = 1,\ \alpha_{r-1} = -1,\ \alpha_{r-2} = \cdots = \alpha_0 = 0,$$

but we take

$$\mathbf{t} = (t_0, t_1, \ldots, t_r) = (0, h, \ldots, rh),$$

and now we have

$$\ell_j(ht; \mathbf{t}) = \frac{(-1)^{r-j}}{j!(r-j)!} \prod_{i \in \{0,\ldots,r\} \setminus \{j\}} (t - i),$$

which ultimately implies

$$\beta_{r-j} = \frac{(-1)^j}{j!(r-j)!} \int_0^1 \prod_{i \in \{0,\ldots,r\} \setminus j\}} (t + i - 1)\, dt, \quad j = 0, \ldots, r.$$

7.1.3 Nyström methods

The ***(explicit) Nyström methods*** are based on the identity

$$x(t_{n+r}) = x(t_{n+r-2}) + \int_{t_{n+r-2}}^{t_{n+r}} f(x(t), t)\, dt. \tag{7.3}$$

The data $(t_n, f_n), \ldots, (t_{n+r-1}, f_{n+r-1})$ can be used to construct a Lagrange interpolating polynomial which may then be substituted for $f(x(t), t)$ in the preceding expression to derive the r-step Nyström method. The order of accuracy is r, same as that of the r-step Adams-Bashforth method. For such methods we always have

$$\alpha_r = 1,\ \alpha_{r-1} = 0,\ \alpha_{r-2} = -1,\ \alpha_{r-3} = \cdots = \alpha_0 = 0.$$

Further general details are left as an exercise.

Let us illustrate the special case of the 2-step explicit Nyström method. In this case we insert into (7.3) the linear interpolation

$$f(x(t), t) \approx f_n \left(1 - \frac{t - t_n}{h}\right) + f_{n+1} \frac{t - t_n}{h},$$

yielding

$$\beta_0 = \int_0^2 (1 - t)\ dt = 0$$

and

$$\beta_1 = \int_0^2 t\, dt = 2.$$

Hence the 2-step explicit Nyström method is given by

$$x_{n+2} - x_n = 2h f_{n+1},$$

or equivalently

$$\frac{x_{n+2} - x_n}{2h} = f_{n+1}.$$

This is called the ***midpoint method*** or ***leapfrog method***. It can of course be derived more simply by substituting the second-order accurate difference quotient approximation

$$x'(t) = \frac{x(t+h) - x(t-h)}{2h} + O(h^2)$$

into the ODE (1.1). (We leave the verification of $O(h^2)$ error here as an exercise.)

7.1.4 Milne-Simpson methods

The ***Milne-Simpson methods*** are to the Nyström methods as the Adams-Moulton methods are to the Adams-Bashforth methods, using the full data $(t_n, f_n), \ldots, (t_{n+r}, f_{n+r})$ for polynomial interpolation. Sometimes these are called (implicit) Nyström methods, and the order of accuracy is $r + 1$ in general. Again

$$\alpha_r = 1,\ \alpha_{r-1} = 0,\ \alpha_{r-2} = -1,\ \alpha_{r-3} = \cdots = \alpha_0 = 0,$$

and further details are left as an exercise.

We illustrate the special case of the 2-step Milne-Simpson method. Here the substitution of the interpolating polynomiali is equivalent to approxmating the integral in (7.3) by *Simpson's rule*,

$$\int_a^b g(t)\,dt \approx \frac{g(a) + 4g\left(\frac{a+b}{2}\right) + g(b)}{6}(b-a),$$

which yields the method that we also call ***Simpson's rule***:

$$x_{n+2} - x_n = \frac{2h}{6}\left(f_n + 4f_{n+1} + f_{n+2}\right).$$

In fact Simpson's rule (due to it's high symmetry about the $(n+1)$-th time step) enjoys one higher order of accuracy than is guaranteed *a priori* by its status as the 2-step Milne-Simpson method, i.e., it is order-4 consistent. The verification of this is left as an exercise.

7.2 Backward differentiation formulas

The LMMs of Adams and Nyström type considered above adopt the approach of taking only a few of the α_j nonzero and then guaranteeing higher orders of accuracy by integral approximation within an integral formulation of the ODE (1.1), as in (7.1).

An alternative approach is to take the differential formulation

$$x'(t) = f(x(t), t) \tag{7.4}$$

and simply plug higher-order accurate finite difference approximations into the left-hand side. In the resulting scheme, only one of the β_j will be nonzero, but perhaps many of the α_j will be nonzero, depending on the order of the method.

We already have some examples that can be interpreted this way: *explicit Euler*, *implicit/backward Euler*, and the *midpoint method* are based on substitution of the backward, forward, and central finite difference formulas

$$\frac{x(t) - x(t-h)}{h}, \quad \frac{x(t+h) - x(t)}{h}, \quad \frac{x(t+h) - x(t-h)}{2h}$$

into (7.4) at time t.

At the same time, we can also think of backward Euler as the result of substituting a backward finite difference formula into (7.4) at time $t+h$. This perspective is generalized by the ***backward differentiation formulas*** (***BDFs***), which are of interest for their *stability* properties to be considered later.

Concretely, we seek coefficients c_j such that

$$z'(t) \approx \frac{1}{h} \sum_{j=0}^{r} c_j z(t - jh) \tag{7.5}$$

for smooth z. In fact it will be possible for the approximation in (7.5) to hold with $O(h^r)$ error.

Then the resulting BDF is derived by substituting the approximation (7.5) into (7.4) at time $t + rh$, resulting in a LMM with coefficients $\alpha_j = c_{r-j}$ for $j = 0, \ldots, r$ and $\beta_r = 1,\ \beta_0 = \cdots = \beta_{r-1} = 0$.

Note that by considering $\tilde{z}(t) = z(-t)$, it is equivalent to find a_j such that

$$z'(t) = \frac{1}{h} \sum_{j=0}^{r} v_j z(t + jh) + O(h^r) \tag{7.6}$$

for all smooth z, i.e., to determine a higher-order *forward* differentiation formula, and then set $c_j = -v_j$ for $j = 0, \ldots, r$.

Simply expand

$$\begin{aligned} \sum_{j=0}^{r} v_j z(t + jh) &= \sum_{j=0}^{r} v_j \left(\sum_{k=0}^{r} z^{(k)}(t) \frac{j^k h^k}{k!} + O(h^{r+1}) \right) \\ &= \sum_{k=0}^{r} \left(\sum_{j=0}^{r} j^k v_j \right) z^{(k)}(t) \frac{h^k}{k!} + O(h^{r+1}), \end{aligned}$$

where we interpret $0^0 = 1$ for notational compactness. Note that to guarantee (7.6) we need

$$\sum_{j=0}^{r} j^k v_j = \begin{cases} 0, & k = 0, \\ 1, & k = 1, \\ 0, & k = 2, \ldots, r. \end{cases}$$

for all $k = 1, \ldots, r$, and we need $\sum_{j=0}^{v_j} j v_j$

We can view this is a linear system of equations for the v_j. Define the $(r+1) \times (r+1)$ matrix $A = (A_{ij}) = (j^i)$, where we have adopted a zero-indexing convention for the indices $i, j = 0, \ldots, r$. We can also write out

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 2 & 3 & \cdots & r \\ 0^2 & 1^2 & 2^2 & 3^2 & \cdots & r^2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0^r & 1^r & 2^r & 3^r & \cdots & r^r \end{pmatrix}$$

If we let $\mathbf{v} = (v_0, \ldots, v_r)^\top \in \mathbb{R}^{r+1}$ be the vector of unknown coefficients, then we are seeking to solve the linear system

$$A\mathbf{v} = e_1, \tag{7.7}$$

where $e_1 = (0, 1, 0, \ldots, 0)^\top$.

In fact $V := A^\top$ is a ***Vandermonde matrix***, i.e., a matrix of the form

$$V = (a_i^j) = \begin{pmatrix} 1 & a_0 & a_0^2 & a_0^3 & \cdots & a_0^r \\ 1 & a_1 & a_1^2 & a_1^3 & \cdots & a_1^r \\ 1 & a_2 & a_2^2 & a_2^3 & \cdots & a_2^r \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_r & a_r^2 & a_r & \cdots & a_r^r \end{pmatrix}.$$

The classical formula

$$\det(V) = \det(A) = \prod_{0 \le i < j \le r} (a_j - a_i)$$

guarantees in particular that A is invertible if and only if the a_i are all distinct. In our case $a_i = i$ for $i = 0, \ldots, r$, so A is invertible and (7.7) has a unique solution $\mathbf{v}$, which uniquely specifies a *unique* BDF scheme of order r via $\alpha_j = -v_{r-j}$ for $j = 0, \ldots, r$.

8 Solving implicit methods

Recall the form (5.6) of the general LMM, which we reproduce here:

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = h \sum_{j=0}^{r} \beta_j f_{n+j}.$$

Without loss of generality, assuming $\alpha_r \neq 0$ (which it always is in methods we consider), we can assume $\alpha_r = 1$ and rearrange as

$$x_{n+r} = \sum_{j=0}^{r-1} (-\alpha_j) x_{n+j} + h\beta_r f(x_{n+r}, t_{n+r}) + \sum_{j=0}^{r-1} h\beta_j f_{n+j}.$$

For fixed $x_n, \ldots, x_{n+r-1}$, define the function $g : \mathbb{R}^d \to \mathbb{R}^d$ by

$$\Phi(u) := \sum_{j=0}^{r-1} (-\alpha_j) x_{n+j} + h\beta_r f(u, t_{n+r}) + \sum_{j=0}^{r-1} h\beta_j f_{n+j}.$$

Then to solve the implicit scheme for x_{n+r}, given all preceding values $x_n, \ldots, x_{n+r-1}$, we must solve the (generally nonlinear) system of equations

$$u = \Phi(u). \tag{8.1}$$

Or, defining $F(u) := u - \Phi(u)$, we may write even more abstractly

$$F(u) = 0. \tag{8.2}$$

General approaches to solving such systems of equations are *iterative*, in the sense that a solution $u^\star$ is furnished as a limit

$$u^\star = \lim_{k \to \infty} u^{(k)},$$

where each successive iterate $u^{(k+1)}$ can be feasibly computed from the last iterate $u^{(k)}$.

We say that an iterative method converges ***linearly*** with rate $\alpha \in (0,1)$ if

$$|u^{(k)} - u^\star| = O(\alpha^k).$$

You might think that we should say that such a method converges 'exponentially,' but in fact it is much less classy to do so. It is the logarithm of the error that converges linearly, and it is important to stay humble, especially since, as we shall see it is always possible in principle to do much better (though perhaps at great cost). Indeed, if a method converges linearly, this means precisely that each successive digit of accuracy is just as costly to produce as the last digit. If a method converges ***superlinearly***, then each successive digit comes more easily, and once we are close to a solution, we tend to get to machine precision in a hurry.

8.1 Fixed-point / Picard iteration

We take the perspective of solving (8.1), which is equivalent to looking for a fixed point of Φ. Hopefully this should recall Picard iteration!

8.1.1 Globally Lipschitz case

Recall our friend the Banach fixed point theorem (Theorem 3), which says that a solution $u^\star$ can be furnished as the limit $\lim_{k\to\infty} u^{(k)}$ of the fixed-point iteration defined by

$$u^{(k+1)} = \Phi(u^{(k)}),$$

for arbitrary initialization $u^{(0)}$, as long as Φ is a contraction mapping.

Note that for our integral-based LMMs, this approach is *a very literal attempt to numerically implement our construction of the solution of (1.1) via Picard iteration on short subintervals of* $[0,T]$, as in the proof of Theorem 6.

Now if f is L-Lipschitz, as in the proof of Theorem 6, then we have

$$\begin{aligned} |\Phi(u) - \Phi(v)| &= h|\beta_r|\,|f(u, t_{n+r}) - f(v, t_{n+r})| \\ &\le L|\beta_r|\,h\,|u - v|, \end{aligned}$$

so if $h < 1/(L|\beta_r|)$, then indeed Φ is a contraction mapping with a unique fixed point.

8.1.2 General case

The global Lipschitz assumption on f is a bit strict, so let us explain why it is not really necessary for obtaining a canonically defined solution (once h becomes sufficiently small).

Unfortunately, there is no way to guarantee in general that Φ is a contraction mapping. However, for small step size h, Φ is *locally* a contraction mapping near $x_n \approx \cdots \approx x_{n+r-1}$, all of which differ only by $O(h)$ provided our solution is so far consistent.

Indeed, by Theorem 13, we know that $\sum_{j=0}^{r-1} \alpha_j = -1$, so

$$\Phi(u) = x_{n+r-1} + O(h),$$

and x_n is 'nearly fixed' by Φ. The same argument via the implicit function theorem as in Section 4.3 implies that for h sufficiently small there exists a (locally unique) fixed point $u^\star = u^\star(h) = x_{n+r-1} + O(h)$.

Moreover,

$$D\Phi(u) = h\beta_r D_x f(u, t_{n+r}),$$

and in particular $\|D\Phi(u^\star)\| = O(h)$, where $\|\cdot\|$ indicates the Euclidean operator norm. Suppose h is sufficiently small such that $\alpha' := \|D\Phi(u^\star)\| < 1$.

We can use $D\Phi$ to control the *local* Lipschitz constant near $u^\star$. Indeed, note that by the fundamental theorem of calculus,

$$\begin{aligned}\Phi(u) - \Phi(v) &= \int_0^1 \frac{d}{dt}\Phi(tu + (1-t)v)\, dt \\ &= \int_0^1 D\Phi(tu + (1-t)v)\,(u - v)\, dt,\end{aligned}$$

so

$$\begin{aligned}|\Phi(u) - \Phi(v)| &\le \int_0^1 |D\Phi(tu + (1-t)v)\,(u - v)|\, dt \\ &\le \int_0^1 \|D\Phi(tu + (1-t)v)\|\, |u - v|\, dt \\ &\le \left(\sup_{w \in B_r(u^\star)} \|D\Phi(w)\|\right) |u - v|,\end{aligned}$$

provided u and v are contained in the ball

$$B_r(u^\star) = \{u \,:\, |u - u^\star| \le r\}$$

of radius r about $u^\star$. Note that if f (hence also Φ) is C^1, then for any $\alpha \in (\alpha', 1)$, there exists $r > 0$ sufficiently small (independent of h) such that $\|D\Phi(u) - D\Phi(u^\star)\| \le \alpha - \alpha'$ on $B_r(u^\star)$. Henceforth assume that r is taken to guarantee this, so

$$|\Phi(u) - \Phi(v)| \le \alpha\, |u - v| \tag{8.3}$$

for all $u, v \in B_r(u^\star)$.

We could apply Banach's fixed point theorem to the space $B_r(u^\star)$ if we knew that it was actually preserved by Φ, i.e., that for $u \in B_r(u^\star)$,the image $\Phi(u) \in B_r(u^\star)$ as well. But actually we *do* know this, because by applying (8.3) with $u^\star$ in place of v, we have

$$|\Phi(u) - u^\star| = |\Phi(u) - \Phi(u^\star)| \le \alpha\, |u - u^\star| \le \alpha r \le r$$

for all $u \in B_r(u^\star)$.

Since $u^\star(h) = x_{n+r-1} + O(h)$, we have for h sufficiently small that $x_{n+r-1} \in B_r(u^\star)$, so in practice we can compute $x_{n+r} := u^\star$ by applying fixed-point iteration intialized at x_{n+r-1}.

As in the discussion of Section (4.3), some extra work (given that the method is convergent) can be used to guarantee that given a fixed choice of $\alpha < 1$, there exists a choice of r that is independent of n and N in the limit $h \to 0$, but we will not get into this.

8.2 Newton's method

In this section we concern ourselves with the solution of a general system of d nonlinear equation sin d unknowns. We have put the problem of solving an implicit method in this form via (8.2), which we reproduce here as

$$F(u) = 0, \tag{8.4}$$

where $F : \mathbb{R}^d \to \mathbb{R}^d$. We have argued above via the implicit function theorem that for the equations of interest to us, we can easily pick an initial guess u_0 which lies close to a true solution $u^\star$ of (8.4) in the sense that $u^\star = u_0 + O(h)$.

The idea of ***Newton's method*** (or the ***Newton-Raphson method***) is to iterate the following steps until convergence is achieved.

1. Linearize the equations (8.4) about our current guess.
2. Solve the linearized equations to get a new guess.

More concretely, given k-th iterate $u_k \in \mathbb{R}^d$ which is our current guess, the $(k+1)$-th iterate is furnished by forming the linearized equations

$$F(u_k) + DF(u_k) \cdot (u - u_k) = 0.$$

Then the unique solution (provided invertibility of the Jacobian at u_k) defines our next iterate as

$$u_{k+1} = u_k - DF(u_k)^{-1} F(u_k). \tag{8.5}$$

The global convergence of Newton's method is very tricky to understand, but *locally* it is not. First of all, if we assume that $DF(u^\star)$ is invertible (as is guaranteed to be the case for h sufficiently small), then $DF(u)$ will remain invertible in some small neighborhood of $u^\star$, and our concerns about invertibility go away.

If we initialize sufficiently close to the true solution $u^\star$ (as can be guaranteed in the $h \to 0$ limit), can we understand the rate of convergence of $u_k \to u^\star$ as $k \to \infty$? Yes:

Theorem 14. *Let $F : \mathbb{R}^d \to \mathbb{R}^d$ be C^2-smooth and $u^\star \in \mathbb{R}^d$ such that $F(u^\star) = 0$ and $DF(u^\star)$ is invertible. There exists $\rho > 0$ such that if $|u_0 - u^\star| \leq \rho$, then Newton's method, as defined by the iteration*

$$u_{k+1} = u_k - DF(u_k)^{-1} F(u_k)$$

for $k \geq 0$, converges to $u^\star$ and moreover there exist constants $\alpha > 0$ and $\beta \in \mathbb{R}$ such that

$$\log |u_k - u^\star| \leq -\alpha 2^k + \beta.$$

Remark 15. We shall see in the proof that there exists $M > 0$ such that

$$|u_{k+1} - u^\star| \leq M |u_k - u^\star|^2$$

for all $k \geq 0$, which implies the stated result. This inequality is the defining condition for ***quadratic convergence***.

Proof. Without loss of generality (just be translating the objective and vertical shift), we can assume that $u^\star = 0$. For any $r > 0$, define $B_r := B_r(0)$ to be the ball of radius r about the solution. Choose $R > 0$ sufficiently small such that DF is invertible on B_R. Now for any $v \in B_R$, we can define

$$\mathcal{I}(v) = v - DF(v)^{-1} F(v)$$

to be the image of v under one iteration of Newton's method.

There are two notions of error relevant to the problem. For a point $v \in \mathbb{R}^d$, we say that its corresponding 'equation error' is $|F(v)|$, which measures the amount by which v fails to solve the target equation $F(u) = 0$. Meanwhile, the 'true error' of v is simply $|v|$, which measures the amount by which v differs from the true solution $u = 0$.

Let $v \in B_R$, and let $w = \mathcal{I}(v)$. We will plug w back into F and determine the equation error $|F(w)|$ for w. We will bound $|F(w)|$ in terms of $|F(v)|$ to show that the equation error is decreasing as we iterate.

Then we will show that the equation error and the true error are essentially equivalent in that they are each bounded by a constant multiple of the other. Therefore the rate of convergence of the equation error implies the same rate of convergence of the true error.

For the first task, it is useful to rewrite $F(u)$ for arbitrary u via Taylor expansion about v as

$$F(u) = F(v) + DF(v) \cdot (u - v) + \psi_v(u), \tag{8.6}$$

where $\psi_v(u)$ is simply defined as the necessary remainder

$$\psi_v(u) := F(u) - \left[F(v) + DF(v) \cdot (u - v)\right],$$

and Taylor's theorem implies that there exists $C > 0$ independent of $v \in B_R$ such that $|\psi_v(u)| \leq C|u - v|^2$ on B_R.

Then observe that if we plug $w = \mathcal{I}(v)$ into (8.6), we obtain precisely

$$F(w) = \psi_v(w),$$

so

$$\begin{aligned} |F(w)| &\leq C|w - v|^2 \\ &= C|DF(v)^{-1}F(v)|^2 \\ &\leq C\|DF(v)^{-1}\|^2|F(v)|^2 \\ &\leq \tilde{C}|F(v)|^2, \end{aligned}$$

where $\tilde{C} > 0$ is a suitably large constant independent of $v \in B_R$, which exists by the continuity of DF^{-1} on B_R.

In summary, we have shown that

$$|F(\mathcal{I}(v))| \leq \tilde{C}|F(v)|^2 \tag{8.7}$$

for all $v \in B_R$.

Now we turn to the second task: showing the equivalence of the equation error and the true error. Specifically, we will show that there exist $a, A, r > 0$ such that

$$a|u| \leq |F(u)| \leq A|u|$$

for all $u \in B_r$.

To see this we Taylor-expand about the origin (true solution) as

$$F(u) = DF(0) \cdot u + \phi(u), \tag{8.8}$$

where $|\phi(u)| \leq C'|u|^2$ for suitable $C' > 0$. Note that then

$$|F(u)| \leq \|DF(0)\|\,|u| + C'|u|^2 = \left(\|DF(0)\| + C'|u|\right)|u|,$$

so as long as $r \leq \frac{\|DF(0)\|}{C'}$, we have

$$|F(u)| \leq A|u|$$

for $u \in B_r$, where we have defined $A := 2\|DF(0)\|$.

Meanwhile, we can rearrange (8.8) to obtain

$$u = DF(0)^{-1}\left[F(u) - \phi(u)\right],$$

so then

$$|u| \leq \frac{1}{2a}\,|F(u)| + C''|u|^2,$$

where $a := \frac{1}{2\|DF(0)^{-1}\|}$ and $C'' := \|DF(0)^{-1}\|C'$. Therefore as long as $r \leq \frac{1}{2C''}$, then we have for $u \in B_r$ that

$$|u| \leq \frac{1}{2a}|F(u)| + \frac{1}{2}|u|,$$

which implies that $|u| \leq (1/a)|F(u)|$.

In summary, we have constructed $a, A, r > 0$ such that

$$a|u| \leq |F(u)| \leq A|u| \tag{8.9}$$

for all $u \in B_r$, as was to be shown. We can always reduce the size of r if necessary to ensure that $r \leq R'$.

Then recall (8.7), which implies that

$$|F(\mathcal{I}(v))| \leq \tilde{C}|F(v)|^2$$

for all $v \in B_r$. But then (8.9) implies that

$$|\mathcal{I}(v)| \leq M|v|^2, \tag{8.10}$$

where $M := \frac{\tilde{C}A^2}{a}$ for all $v \in B_r$.

Then by taking $\rho \leq \min\left(r, \frac{1}{2M}\right)$ we have

$$|\mathcal{I}(v)| \leq \frac{1}{2}|v| \tag{8.11}$$

for all $v \in B_\rho$, so in particular $\mathcal{I}$ maps B_ρ into itself.

Since $\mathcal{I}$ maps B_r into itself, it follows that if our initial guess $u_0 \in B_\rho$, then $u_k \in B_r$ for all $k \geq 0$, and by (8.10), we have

$$|u_{k+1}| = |\mathcal{I}(u_k)| \leq M|u_k|^2, \tag{8.12}$$

i.e., *quadratic convergence*.

We will now see why quadratic convergence in this sense implies the stated rate of convergence of the sequence u_k, i.e., the stated bound on $|u_k - u^\star|$.

By taking logarithms of (8.12) we have

$$\log|u_{k+1}| \leq b + 2\log|u_k|,$$

where $b = \log M$.

Defining a_k by the recursion

$$a_{k+1} := b + 2a_k$$

and $a_0 := \log|u_0|$, we have inductively that $\log|u_k| \leq a_k$, so we need only concern ourselves with the a_k.

In fact we can solve for a_k exactly as

$$a_1 = b + 2a_0, \quad a_2 = b + 2b + 4a_0, \quad a_3 = b + 2b + 4b + 8a_0, \quad \dots$$

or in general,

$$\begin{aligned} a_k &= 2^k a_0 + \left(\sum_{l=0}^{k-1} 2^l \right) b \\ &= 2^k a_0 + (2^k - 1) b \\ &= 2^k (a_0 + b) - b. \end{aligned}$$

Therefore, if $a_0 + b < 0$ we are in good shape.

Indeed, note that since $u_0 \in B_\rho$ and $\rho \leq \frac{1}{2B}$, we have in particular that $|u_0| \leq \frac{1}{2M}$, so $\log |u_0| \leq \log(1/2) - \log M$, which implies

$$a_0 + b \leq \log(1/2),$$

so

$$a_k \leq -\log(2) 2^k - b.$$

Taking $\alpha = \log(2)$ and $\beta = -b$ concludes the proof. □

What are the drawbacks of Newton's method? First observe that computing the Jacobian $DF(u)$, if we retrace our steps back to the ODE $x'(t) = f(x(t), t)$ that we're trying to solve, actually requires us to compute derivatives of f, which is undesirable in general. However, even more fundamentally, the computational scaling of Newton's method with respect to the state dimension d is *in general* $O(d^3)$ due to the cost of the linear solve $DF(u_k)^{-1} F(u_k)$. Depending on the problem structure, it may or may not be possible to do better, with varying amounts of effort. You should roughly think of Newton's method as unbeatable for small d but less and less practical as d becomes large. Even when Newton's method is completely impractical, it is conceptually important as an ideal that one can wishfully (if imperfectly) strive to attain by other, scrappier methods.

8.3 Anderson acceleration / DIIS

Consider the general problem of finding a fixed point of some map $\Phi : \mathbb{R}^d \to \mathbb{R}^d$. Assume even that Φ is a contraction map, so a unique fixed point can be constructed in principle via fixed-point iteration. Since we have been talking a lot about fixed-point iterations, I want to tell you about a more sophisticated approach of great generality that is not widely known in applied math. Indeed I am not confident about the use of it in implicit integrators for ODEs, but this is a good chance to learn it nonetheless!

Define $g : \mathbb{R}^d \to \mathbb{R}^d$ by $g(u) = \Phi(u) - u$, which returns a 'residual' vector, satisfying the condition that $g(u) = 0$ if and only if $\Phi(u) = u$, i.e., if and only if u is a fixed point. For this choice of residual, the following method is known as ***Anderson acceleration***. The ideas have appeared independently in quantum chemistry, where the framework is known as ***direct inversion in the iterative subspace (DIIS)*** or ***Pulay mixing*** and can involve a more general notion of residual.

This also will be an iterative method, furnishing a sequence of iterates $u_k \to u^\star$ satisfying $\Phi(u^\star) = u^\star$. However, unlike ordinary fixed-point iteration, DIIS will use the history of *previous* iterates as well in the computation of the next iterate.

Suppose inductively that $u_1, \ldots, u_k$ have been furnished, and let $m \leq k$ be a 'history parameter' dictating how many previous iterates will be used. Therefore in practice m must be k-dependent, and we should really write $m = m_k := \min(k, m_{\max})$, where $m_{\max}$ is fixed. However, we will omit the k-dependence of m from the notation. It is also useful before proceeding to define residual vectors $v_j := g(u_j)$ for all j.

Consider the $(m-1)$-dimensional *affine* subspace generated by the m vectors

$$\{\Phi(u_{k-m+1}), \ldots, \Phi(u_k)\},$$

i.e., the 'suggestions' made by ordinary fixed-point iteration for the m previous iterates. Concretely this affine subspace, can be defined as

$$\mathcal{S} := \left\{ \sum_{l=1}^{m} c_l \Phi(u_{k-m+l}) \,:\, \sum_{l=1}^{m} c_l = 1 \right\}.$$

We are going to take 'best' composite suggestion, based on the last m iterates, from this affine subspace.

How to define best? Supposing that we are near $u^\star$, we are inspired by the first-order approximation of Φ , which we write as

$$\Phi(u) \approx A(u - u^\star) + b,$$

where $A = D\Phi(u^\star)$ and $b = \Phi(u^\star) = u^\star$. Note that if $\sum_{l=1}^{m} c_l = 1$, then this approximation yields

$$\begin{aligned} \sum_{l=1}^{m} c_l \Phi(u_{k-m+l}) &\approx \sum_{l=1}^{m} c_l A(u_{k-m+l} - u^\star) + b \\ &= A\left(\sum_{l=1}^{m} c_l u_{k-m+l} - u^\star \right) + b \\ &\approx \Phi\left(\sum_{l=1}^{m} c_l u_{k-m+l} \right), \end{aligned}$$

i.e., linear combination according to the c_l can be passed through Φ approximately as

$$\sum_{l=1}^{m} c_l \Phi(u_{k-m+l}) \approx \Phi\left(\sum_{l=1}^{m} c_l u_{k-m+l} \right). \tag{8.13}$$

Consider the problem

$$\underset{c \in \mathbb{R}^m}{\text{minimize}} \left| \sum_{l=1}^{m} c_l v_{k-m+l} \right|^2, \text{ subject to } \sum_{l=1}^{m} c_l = 1, \tag{8.14}$$

which is a constrained least squares problem that can be solved exactly by a suitable $(m+1) \times (m+1)$ matrix inversion.

Now

$$\begin{aligned} \sum_{l=1}^{m} c_l v_{k-m+l} &= \sum_{l=1}^{m} c_l \Phi(u_{k-m+l}) - \sum_{l=1}^{m} c_l u_{k-m+l} \\ &\approx \Phi(\hat{u}) - \hat{u}, \end{aligned}$$

where $\hat{u} := \sum_{l=1}^{m} c_l u_{k-m+l}$, so by solving (8.14) we are approximately solving the problem of finding $\Phi(\hat{u})$ in the (generally nonlinear) submanifold

$$\tilde{\mathcal{S}} := \left\{ \Phi\left(\sum_{l=1}^{m} c_l u_{k-m+l} \right) \,:\, \sum_{l=1}^{m} c_l = 1 \right\},$$

which is very close to its preimage $\hat{u}$. (Locally near $u^\star$, the map Φ is invertible, as is guaranteed provided $D\Phi(u^\star)$ is invertible. If this is not the case, then our intuition is in big trouble.) Near $u^\star$, we have already argued that $\tilde{\mathcal{S}}$ and $\mathcal{S}$ can be identified closely via (8.13). Therefore we propose as our next iterate

$$u_{k+1} := \sum_{l=1}^{m} c_l \Phi(u_{k-m+l}),$$

which completes the specification of the method.

In summary, for each k, given previous iterates $u_1, \ldots, u_k$ and residuals $v_1, \ldots, v_k$, we perform the following:

1. Set $m = m_k = \min(k, m_{\max})$.
2. Determine $c_1, \ldots, c_m \in \mathbb{R}$ with sum equal to 1 by solving (8.14).
3. Define next iterate $u_{k+1} := \sum_{l=1}^{m} c_l \Phi(u_{k-m+l})$ and next residual $v_{k+1} := g(u_{k+1})$.

Note that the evaluations $\Phi(u_j)$ can be stored, so we only need to perform one additional evaluation of Φ per step of this method, no worse than a single step of ordinary fixed-point iteration. Usually computing Φ will be the bottleneck, unless m is allowed to become very large, which often is not a great idea anyway due to stability concerns.

Unfortunately, there aren't really any clean theorems about the convergence of Anderson acceleration in practice, and the results can be somewhat difficult to predict, but the practical speedup can be quite dramatic for many significant problems.

9 Zero-stability

As we warned earlier and shall now see concretely, ***consistency does not imply convergence!*** Let's see what can go wrong.

All of the methods we introduced above had a natural motivation, but we can write down consistent methods that look kind of silly. Consider for example the two-step method

$$x_{n+2} + 4x_{n+1} - 5x_n = h(4f_{n+1} + 2f_n). \tag{9.1}$$

Here we have $r = 2$, $\alpha_0 = -5$, $\alpha_1 = 4$, $\alpha_2 = 1$, $\beta_0 = 2$, $\beta_1 = 4$, and $\beta_2 = 0$, so the conditions of Theorem 13 are satisfied, and the LMM is consistent. We'll see later that this method is not convergent!

9.1 Difference equations

Besides consistency, another necessary condition for convergence can be understood by examining the trivial ODE $x'(t) = 0$ defined by $f \equiv 0$, which admits the constant solution $x(t) = x_0$ for all t. Here it is convenient to consider the equation over the complex numbers, i.e., to think of $x(t), x_n \in \mathbb{C}^d$.

Applying a general LMM (5.6) to this ODE yields the ***difference equation***

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = 0, \quad n = 0, \ldots, N - r. \tag{9.2}$$

Consider the ansatz

$$x_n = w^n e_i \tag{9.3}$$

for a solution, where w is a scalar and $e_i \in \mathbb{R}^d$ is the i-th standard basis vector, $i = 1, \ldots, d$. Plugging into the difference equation (9.2)yields

$$\begin{aligned} 0 &= \sum_{j=0}^{r} \alpha_j w^{n+j} e_i \\ &= \left(\sum_{j=0}^{r} \alpha_j w^j \right) w^n e_i \\ &= \rho(w) w^n e_i, \end{aligned}$$

where we recall that ρ is the first characteristic polynomial of the LMM. Note that if w is a root of ρ, then we have a solution. Note that even in the case where $w = 0$ is a root (i.e., $\alpha_0 = 0$), if we adopt the convention that $0^0 = 1$ so that $x_n = \delta_{n0} e_i$, we still obtain a nontrivial solution of (9.2).

In fact, provided that ρ has N distinct roots, linear combinations of solutions of the form (9.3) define the *general* solution of (9.2), as can be verified by a dimension counting argument. Indeed, note that the set of $\mathbf{x} = (x_0, \ldots, x_N)$ solving (9.2) is a subspace. Moreover, once $x_0, \ldots, x_{r-1}$ are fixed, the rest of the x_j are determined linearly in terms of these. (Each x_j can be determined from the r preceding values just by noting $x_{n+r} = -\sum_{j=0}^{r-1} \alpha_j x_{n+j}$.) However, the values $x_0, \ldots, x_{r-1}$ are themselves unconstrained. It follows that the solution space is $r \times d$ dimensional, and since the vectors $(1, w_k, w_k^2, \ldots, w_k^N)$ are linearly independent (cf. Vandermonde!) for distinct k, the claim follows.

For completeness, what about the case of repeated roots? Suppose that w is a repeated root, so $\rho(w) = \rho'(w) = 0$. For simplicity let $d = 1$, and consider a solution of the form

$$x_n = n w^n.$$

Then

$$\begin{aligned} \sum_{j=0}^{r} \alpha_j x_{n+j} &= \sum_{j=0}^{r} \alpha_j (n + j) w^{n+j} \\ &= n w^n \sum_{j=0}^{r} \alpha_j w^j + w^n \sum_{j=0}^{r} j \alpha_j w^j \\ &= n w^n \rho(w) + w^{n+1} \rho'(w) \\ &= 0. \end{aligned}$$

Similarly, if w is root of order at least 3, then $\rho''(w) = 0$ as well, and

$$x_n = n^2 w^n$$

also defines a solution. Indeed, using the fact already shown that $\sum_{j=0}^{r} \alpha_j w^j = \sum_{j=0}^{r} j\alpha_j w^j = 0$, we have

$$\begin{aligned}
\sum_{j=0}^{r} \alpha_j x_{n+j} &= \sum_{j=0}^{r} \alpha_j (n+j)^2 w^{n+j} \\
&= w^n \sum_{j=0}^{r} j^2 \alpha_j w^j \\
&= w^n \sum_{j=0}^{r} j(j-1) \alpha_j w^j \\
&= w^{n+2} \rho''(w) \\
&= 0.
\end{aligned}$$

More generally, if w is a root of order m, then $x_n = n^k w^n$ defines a solution for each $k = 0, \ldots, m-1$.

Note that in the annoying case where $w = 0$ is a repeated root, our formula $x_n = n^k w^n = \delta_{n0}$ does not furnish any new solutions. Let us inspect this case more closely. Observe that if 0 is a root of order m, then it must be the case that $\alpha_0 = \cdots = \alpha_{m-1} = 0$. In fact, for any $k = 0, \ldots, m-1$, then the formula $x_n = \delta_{nk}$ evidently furnishes a solution of (9.2).

Theorem 16. *A general complex solution of (9.2) is a complex linear combination of the $r \times d$ basis solutions of the form*

$$x_n = \begin{cases} n^k w^n e_i, & w \neq 0 \\ \delta_{nk} e_i, & w = 0 \end{cases}, \quad n = 0, \ldots, N,$$

where $i = 1, \ldots, d$, w is an order-m root of the first characteristic polynomial ρ of the LMM, and $k = 0, \ldots, m-1$.

Remark 17. To complete the proof of this theorem, we would have to establish that the r vectors $(n^k w^n)_{n=0}^{N}$, where w is an order-m root of ρ and $k = 0, \ldots, m-1$, are in fact linearly independent. Note that it suffices to check that they are linearly independent in the case $N = r - 1$. We omit a proof of this fact.

9.2 Root condition

Note that if we perfectly initialize (9.2) with $x_0 = \cdots = x_{r-1} = x(0)$, then by inspection of

$$x_{n+r} = -\sum_{j=0}^{r-1} \alpha_j x_{n+j}$$

together with the fact that $-\sum_{j=0}^{r-1} \alpha_j = 1$, we can inductively deduce the solution

$$x_n = x_0$$

for $n = 0, \ldots, N$, which perfectly agrees with the exact solution of the ODE $x'(t) = 0$.

This corresponds to a linear combination of basis solutions $x_n = n^k w^n e_i$, where $k = 0$ and $w = 1$. (Indeed, note that 1 is *always* a root of ρ for a consistent LMM!)

However, if we make even a vanishly small initialization error $x_n = x_0 + \Theta(h^p)$, we will in general pick up contributions to our solution of size $\Theta(n^k w^n h^p)$, where $k > 1$ or $w \neq 0$,

corresponding to a coefficient of size $\Theta(h^p)$ for some basis solutions $n^k w^n e_i$. Observe that the contribution at time N will amount to $\Theta(N^{k-p}w^N)$. This will not be $o(1)$ unless either $|w| < 1$ or $|w| = 1$, $k < p$.

Now writing $x_n^{(N)}$ to emphasize dependence of the solution on the time step, we normally say that an LMM is ***convergent*** if we have $\|x_{0:N}^{(N)} - x(t_{0:N})\|_\infty \to 0$ provided that the start values are merely consistent in that $|x_n^{(N)} - x(nh)| = o(1)$ for $n = 0, \dots, r-1$ as $h \to 0$. Evidently, given such a loose assumption on the starting value accuracy, we cannot accommodate a repeated root with $|w| = 1$. Even with tighter assumptions on the starting values, we can see that in general a repeated root with $|w| = 1$ would mess up the order of accuracy of the method, which is not cool.

Definition 18. We say that the first characteristic polynomial of an LMM satisfies the ***root condition*** if all roots w have modulus at most 1 and all repeated roots have modulus strictly less than 1. An LMM that satisfies the root condition is called ***zero-stable***.

It can be shown (but we won't bother to do so formally) that zero-stability is necessary for an LMM to converge in general. (Just look at the trivial system considered above.) What we want to prove is that zero-stability, together with consistency, implies convergence. But first we consider a few examples.

9.3 Examples

9.3.1 Silly example

Let's show that the silly example (9.1) fails to be zero-stable. Note that in this case

$$\rho(w) = w^2 + 4w - 5 = (w+5)(w-1),$$

hence $w = -5$ is a root! Evidently $|-5| > 1$, so the method is not zero-stable.

9.3.2 Adams-type methods

Next, observe that the Adams-type methods all yield

$$\rho(w) = w^r - w^{r-1} = w^{r-1}(w-1),$$

with roots $w = 0$ (multiplicity $r-1$) and $w = 1$ (multiplicity 1). Hence the Adams-type methods are all zero-stable.

9.3.3 Nyström-type methods

Meanwhile, the Nyström-type methods all yield

$$\rho(w) = w^r - w^{r-2} = w^{r-2}(w^2-1) = w^{r-2}(w+1)(w-1),$$

with roots $w = 0$ (multiplicity $r-2$), $w = 1$ (multiplicity 1), and $w = -1$ (multiplicity 1). Hence the Nyström-type methods are all zero-stable.

9.3.4 Backward differentiation formulas

The zero-stability of the remaining family of methods introduced above, the BDFs, is more difficult to ascertain. It can be shown that r-th order BDF is zero-stable for $r = 1, \dots, 6$, but not zero-stable for $r > 6$. This is one of those weird accidental facts, and there does not appear to be an elegant way to justify it. The best way is probably just to see for yourself by root-finding.

9.4 Dahlquist's first barrier theorem

We already have some intuition for how increasing the number of steps of a LMM can allow us to improve the order of accuracy. If we additionally insist on zero-stability (which is an essential requirement as we have seen), what is the best of order of accuracy possible for a given number of steps? Dahlquist's first barrier theorem tells us some fairly tight constraints.

Theorem 19 (Dahlquist's first barrier theorem)**.** *The order p of an r-step **zero-stable** LMM is constrained by the following conditions:*

1. *$p \leq r + 2$ if r is even,*
2. *$p \leq r + 1$ if r is odd,*
3. *$p \leq r$ if $\beta_r \leq 0$ (in particular whenever $\beta_r = 0$, i.e., whenever the LMM is explicit).*

Remark 20. Recall that the r-step Adams-Moulton method has order of accuracy $r + 1$, hence saturates the bound for odd r, and the r-step Adams-Bashforth method has order of accuracy r, hence saturates the bound for explicit LMMs. Meanwhile, Simpson's rule is a 2-step method with order-4 accuracy, hence saturates the bound in the even case. I am not sure whether in general there exists an implicit method saturating the bound for arbitrary even r.

10 Convergence theorem

Our goal in this section is to prove the following important theorem (due essentially to Dahlquist).

Theorem 21 (Dahlquist)**.** *Suppose that f C^p-smooth with uniformly bounded first and second derivatives and that the r-step LMM (5.6) is both order-p consistent and zero-stable, furnished with start values $x_j = x(jh) + O(h^p)$ for $j = 0, \ldots, r-1$, where $x = x(t)$ is the exact solution of (1.1). For $h = 1/N$ sufficiently small, there exists a unique solution $(x_0, \ldots, x_N)$ to the LMM (5.6), and*

$$\max_{n=0,\ldots,N} |x_n - x(tn)| = O(h^p).$$

Remark 22. Given smoothness, having uniformly bounded first derivative is equivalent to being globally Lipschitz (cf. the manipulations in Section 8.1.2). By similar considerations as mentioned in earlier theorems, such global boundedness assumptions can be relaxed with extra effort.

Remark 23. Note that the existence and uniqueness of a solution of the (5.6) for h sufficiently large follows from the discussion of Section 8.1.1. Moreover, the C^p-smoothness of f guarantees, via repeated differentiaton of the equation $x'(t) = f(x(t), t)$, that x is C^{p+1}-smooth. Order-p consistency and the Lagrange error bound then together imply that $|\mathscr{L}_h x(t)| \leq C \sup_{\xi \in [0,T]} \left|x^{(p+1)}(\xi)\right|$ for some C.

For notational simplicity we will simply consider the scalar case $d = 1$. There is no essential difference in proving the case of general d.

10.1 The step map

It's useful to consider the linear map $T : \mathbb{R}^r \to \mathbb{R}^r$ defined by

$$T(x_0, \ldots, x_{r-1}) = \left(x_1, \ldots, x_{r-1}, -\sum_{j=0}^{r-1} \alpha_j x_j \right),$$

which sends a solution of (9.2) at times $n, \ldots, n+r-1$ to the solution at times $n+1, \ldots, n+r$. (***We will use a zero-indexing convention for vectors in this section!***) We'll call this the 'step map' (*not a standard name!*) since it steps forward the solution of (9.2) by one increment of time.

In fact the basis solutions introduced in Theorem 16, earlier furnish a generalized eigenbasis for T. Indeed, let w be a nonzero root, and consider $\mathbf{x} = (x_0, \ldots, x_{r-1})$ defined by

$$x_j = w^j, \quad j = 0, \ldots, r-1.$$

Following Theorem 16, $\mathbf{x}$ defines the first r values of a solution $x_j = w^j$ for (9.2). Then $[T(\mathbf{x})]_{r-1}$ simply fills in the next value of this solution, i.e., $[T(\mathbf{x})]_{r-1} = w^r$, and in general we have

$$[T(\mathbf{x})]_j = w^{j+1} = w x_j, \quad j = 0, \ldots, r-1,$$

so

$$T(\mathbf{x}) = w\mathbf{x},$$

i.e., $\mathbf{x}$ is an eigenvector with eigenvalue w.

In the case where $w = 0$ is a root, repeated with order m (so that $\alpha_0 = \cdots = \alpha_{m-1} = 0$), it is clear that for any $k = 0, \ldots, m-1$, the vector $\mathbf{x}$ defined by

$$x_j = \delta_{jk}, \quad j = 0, \ldots, r-1$$

is $k = 0, \ldots, m-1$ is a null vector for T, i.e., $T(\mathbf{x}) = 0$.

Meanwhile, if w is a repeated root with order $m > k$ and

$$x_j = j^k w^j, \quad j = 0, \ldots, r-1,$$

then again following Theorem 16, $[T(\mathbf{x})]_{r-1}$ fills in the next value of this solution for (9.2), i.e., $[T(\mathbf{x})]_{r-1} = r^k w^r$, and

$$[T(\mathbf{x})]_j = (j+1)^k w^{j+1} = w \sum_{l=0}^{k} \binom{k}{l} j^l w^j, \quad j = 0, \ldots, r-1.$$

At this point it is helpful to label our basis vectors. If w is a root of multiplicity m, then for $k = 0, \ldots, m-1$, we let $\mathbf{x}^{(w,k)}$ be defined by

$$x_j^{(w,k)} = \begin{cases} j^k w^j, & w \neq 0 \\ \delta_{jk}, & w = 0 \end{cases}, \quad j = 0, \ldots, r-1. \tag{10.1}$$

Then we have precisely shown that

$$T(\mathbf{x}^{(w,k)}) = w \sum_{l=0}^{k} \binom{k}{l} \mathbf{x}^{(w,l)}.$$

In other words, the basis $\{\mathbf{x}^{(w,k)}\}$ block-diagonalizes the operator T, and the $m \times m$ block of the matrix associated with the root w can be written

$$B^{(w)} = w \begin{pmatrix} 1 & \binom{1}{0} & \binom{2}{0} & \binom{3}{0} & \cdots & \binom{m-1}{0} \\ & 1 & \binom{2}{1} & \binom{3}{1} & \cdots & \binom{m-1}{1} \\ & & 1 & \binom{3}{2} & \cdots & \binom{m-1}{2} \\ & & & 1 & \cdots & \binom{m-1}{3} \\ & & & & \ddots & \vdots \\ & & & & & 1 \end{pmatrix}.$$

Note that $B^{(w)} - wI$ is nilpotent, implying in particular that the $\mathbf{x}^{(w,k)}$ are generalized eigenvectors, and the eigenvalues of T are precisely the roots of ρ.

If ρ satisfies the root condition, all such blocks $B^{(w)}$ are either 1×1 matrices with $|w| \leq 1$ or possibly larger matrices with $|w| < 1$. In the latter case, it can be shown (though we leave this as an exercise) that $[B^{(w)}]^p \to 0$ as $p \to \infty$. In particular, there exists some p sufficiently large such that $\|B^{(w)}\| \leq 1$ for all roots w of ρ. Accordingly, the matrix of T^p with respect to the generalized eigenbasis has operator norm bounded by 1.

It is useful to introduce a new norm on $\mathbb{C}^r$ that is adapted to the generalized eigenbasis. Indeed, define $|||\mathbf{x}|||$ to be the Euclidean norm of the vector coefficients of $\mathbf{x}$ in the generalized eigenbasis. Then let $||| \cdot |||$ denote also by extension the associated operator norm. Then the preceding discussion guarantees that $|||T^p||| \leq 1$ for all p sufficiently large. That's the upshot of this whole subsection! In summary, we have proved:

Theorem 24. *The eigenvalues of the step map T are precisely the roots of the first characteristic polynomial ρ of the LMM (5.6). The vectors $\mathbf{x}^{(w,k)}$, where w is a root of ρ of multiplicity m and $k = 0, \ldots, m-1$, form a generalized eigenbasis for T, each associated to the eigenvalue w. If ρ satisfies the root condition, then there exists a norm $||| \cdot |||$ on $\mathbb{C}^r$ and an associated operator norm (denoted the same) such that for all p sufficiently large, $|||T^p||| \leq 1$.*

10.2 Warm-up: inhomogeneous linear difference equation

Let us now consider the following *inhomogeneous* generalization of the difference equation (9.2):

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = g_n, \quad n = 0, \ldots, N-r, \tag{10.2}$$

where g_n is some fixed right-hand side, independent of the x_n. We can solve (10.2) using the step map. Indeed, note that given all preceding values, we can solve for x_{n+r} as

$$x_{n+r} = g_n - \sum_{j=0}^{r-1} \alpha_j x_{n+j} = g_n - [T(\mathbf{x}^{(n)})]_{r-1},$$

where $\mathbf{x}^{(n)} := (x_n, \ldots, x_{n+r-1})$. More compactly, we have

$$\mathbf{x}^{(n+1)} = T(\mathbf{x}^{(n)}) + \mathbf{g}^{(n+1)},$$

where we define

$$\mathbf{g}^{(n+1)} = (0, \ldots, g_n), \quad n = 0, \ldots, N-r.$$

It is convenient also to define

$$\mathbf{g}^{(0)} := \mathbf{x}^{(0)}.$$

Then given an initialization of starting values $\mathbf{g}^{(0)} = \mathbf{x}^{(0)} = (x_0, \dots, x_{r-1})$, we obtain sequentially:

$$\begin{aligned}
\mathbf{x}^{(1)} &= T(\mathbf{g}^{(0)}) + \mathbf{g}^{(1)}, \\
\mathbf{x}^{(2)} &= T^2(\mathbf{g}^{(0)}) + T(\mathbf{g}^{(1)}) + \mathbf{g}^{(2)}, \\
\mathbf{x}^{(3)} &= T^3(\mathbf{g}^{(0)}) + T^2(\mathbf{g}^{(1)}) + T(\mathbf{g}^{(2)}) + \mathbf{g}^{(3)}, \\
&\dots,
\end{aligned}$$

and in general we have

$$\mathbf{x}^{(n)} = \sum_{k=0}^{n} T^{n-k}(\mathbf{g}^{(k)}), \quad n = 0, \dots, N - r + 1, \tag{10.3}$$

which defines an exact solution for (10.2).

Now is the time to remember the hidden mesh-dependence (i.e., N-dependence or h-dependence) of the scheme. Suppose

$$\max_{n=0,\dots,r-1} |x_n| = O(h^p), \quad \max_{n=0,\dots,N-r} |g_n| = O(h^{p+1})$$

as $h \to 0$. We have perturbed the equation (9.2) by adding an $O(h^{p+1})$-small right-hand side, and we are considering an $O(h^p)$-small initialization. The goal is to show that the solution is also $O(h^p)$-small.

Indeed, note that from (10.3) that

$$|||\mathbf{x}^{(n)}||| \le \sum_{k=0}^{n} |||T^{n-k}||| \cdot |||\mathbf{g}^{(k)}|||.$$

Suppose ρ satisfies the root condition, so by Theorem 24

$$C := \sup_{k=0,1,2,\dots} |||T^k|||$$

is finite. (The fancy norm is not really necessary here, but it will be useful later.)

It follows by triangle inequality that

$$\max_{n=0,\dots,N} |||x^{(n)}||| \le C\,|||\mathbf{x}^{(0)}||| + CN \max_{k=1,\dots,N-r+1} |||\mathbf{g}^{(k)}||| = O(h^p),$$

where we have used the fact that all finite-dimensional norms are equivalent. Indeed, said fact then further implies that

$$\max_{n=0,\dots,N} |x^{(n)}| = O(h^p),$$

as desired.

10.3 Getting warmer: linear case

Now consider the case of a linear ODE defined by $f(x,t) = \lambda x$. We want to show that the LMM (5.6) is convergent in this case, given consistency and zero-stability. More specifically, given order-p consistency we want to show order-p accuracy of the solution.

Recall the defining equation for the LMM (5.6):

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = \lambda h \sum_{j=0}^{r} \beta_j x_{n+j}, \quad n = 0, \ldots, N-r,$$

let $x = x(t)$ be the true solution, and suppose $|x_n - x(nh)| = O(h^p)$ for $n = 0, \ldots, r-1$. (Note that there is a hidden dependence of the solution x_n on the mesh, i.e., on N or equivalently on h.)

Now the whole point of (order-p) consistency is that if we plug in the true solution to the defining equation for the LMM, it holds with vanishingly small error, specifically $O(h^{p+1})$ error. To wit

$$\sum_{j=0}^{r} \alpha_j x(t_{n+j}) = \lambda h \sum_{j=0}^{r} \beta_j x(t_{n+j}) + g_n,$$

where $\max_{n=0,\ldots,N-r} |g_n| = O(h^{p+1})$.

Now define an error displacement $\epsilon_n := x_n - x(t_n)$. (Note that ϵ_n is to be distinguished from the absolute error notation $E_n = |\epsilon_n|$ considered earlier in the course. We'll still call it the 'error.') Observe that the error satisfies a linear difference equation with small right-hand side, namely:

$$\sum_{j=0}^{r} (\alpha_j - \lambda h \beta_j)\, \epsilon_j = g_n, \quad n = 0, \ldots, N-r.$$

This is a small variation on (10.2) in that the shift map itself has been perturbed by $O(h)$. Indeed define a perturbed a shift map T_h associated to the perturbed polynomial $\rho_h(w) = \sum_{j=0}^{r} (\alpha_j - \lambda h \beta_j) w^j$. It is easy to see that $T_h = T + B_h$, where $B_h = O(h)$ is a linear map.

Following (10.3), we have a solution

$$\boldsymbol{\epsilon}^{(n)} = \sum_{k=0}^{n} T_h^{n-k}(\mathbf{g}^{(k)}), \quad n = 0, \ldots, N-r+1, \tag{10.4}$$

where $\boldsymbol{\epsilon}^{(n)} := (\epsilon_n, \ldots, \epsilon_{n+r-1})$, and

$$\mathbf{g}^{(n)} = \begin{cases} (0, \ldots, g_{n-1}), & n = 1, \ldots, N-r+1 \\ (\epsilon_0, \ldots, \epsilon_{r-1}), & n = 0. \end{cases}$$

Now by Theorem 24 there exists some m, which shall now remain *fixed* (even as h is changed), such that $|||T^m||| \leq 1$. Then expand

$$T_h^m = (T + B_h)^m = T^m + \sum_{k=1}^{m} \binom{m}{k} T^{m-k} B_h^k,$$

and note that the remainder term of the sum is $O(h)$, so there exists some $c > 0$ such that

$$|||T_h^m||| \leq 1 + ch$$

for all h sufficiently small.

Then for any nonnegative integer M, we can always write $M = qm + l$, where $q \leq M/m$ and the remainder $l \in \{0, \ldots, m-1\}$. Therefore

$$T_h^M = T_h^l \, (T_h^m)^q \,,$$

so

$$|||T_h^M||| \leq |||T_h|||^l \; |||T_h^m|||^q.$$

Since $l \in \{0, \ldots, m-1\}$ and m is fixed, $|||T_h|||^l$ is bounded by some constant A for all h sufficiently small. Then

$$|||T_h^M||| \leq A(1+ch)^q \leq A(1+ch)^{M/m} \leq A(e^{ch})^{M/m} = Ae^{aMh},$$

where we have defined a new constant $a := c/m$.

This bound is key because as long as $M \leq N$, we still have a constant bound

$$|||T_h^M||| \leq C := Ae^{aT}.$$

Then plugging into (10.4), we obtain

$$\begin{aligned} |||\boldsymbol{\epsilon}^{(n)}||| &\leq |||T^n||| \cdot |||\mathbf{g}^{(0)}||| + \sum_{k=1}^{n} |||T_h^{n-k}||| \cdot |||\mathbf{g}^{(k)}||| \\ &\leq C|||\mathbf{g}^{(0)}||| + CN \max_{k=1,\ldots,N} |||\mathbf{g}^{(k)}|||. \end{aligned}$$

By norm equivalence and our bounds on $\epsilon_0, \ldots, \epsilon_{r-1}$ as well as our uniform bound over $g_0, \ldots, g_n$, we have in turn that

$$|||\mathbf{g}^{(0)}||| = O(h^p), \quad \max_{k=1,\ldots,N} |||\mathbf{g}^{(k)}||| = O(h^{p+1}),$$

and it follows that

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

By norm equivalence we conclude that

$$\max_{n=0,\ldots,N} E_n = O(h^p),$$

which is precisely convergence of order p, as desired.

10.4 Lipschitz case

Once again we want to obtain a difference equation for the error $\epsilon_n = x_n - x(nh)$. Order-p consistency (cf. Remark 23) implies that

$$\sum_{j=0}^{r} \alpha_j x(t_{n+j}) = h \sum_{j=0}^{r} \beta_j f(x(t_{n+j}), t_{n+j}) + \tilde{g}_n, \quad n = 0, \ldots, N-r.$$

where $\max_n |\tilde{g}_n| = O(h^{p+1})$. Subtracting from the LMM (5.6) we have

$$\sum_{j=0}^{r} \alpha_j \epsilon_{n+j} = h \sum_{j=0}^{r} \beta_j d_{n+j} + \tilde{g}_n, , \quad n = 0, \ldots, N-r. \tag{10.5}$$

where

$$d_n := f(x_n, t_{n+j}) - f(x(t_n), t_n).$$

In Section 10.3, we were able to simplify the expression for d_n dramatically using the linearity of f. We will mimic this as best as we can via *linearization*! Indeed, for all $n = 0, \ldots, N$, we can approximate

$$d_n = \lambda_n \epsilon_n + b_n,$$

where

$$\lambda_n := \partial_x f(x(t_n), t_n), \quad b_n := d_n - \lambda_n \epsilon_n,$$

and

$$|b_n| \leq \frac{1}{2} \|\partial_x^2 f\|_{L^\infty} |\epsilon_n|^2 \tag{10.6}$$

by the Lagrange error bound.

Then we can rewrite (10.5) as

$$\sum_{j=0}^{r} (\alpha_j - \lambda_{n+j} \beta_j h) \, \epsilon_{n+j} = g'_n, \quad n = 0, \ldots, N-r, \tag{10.7}$$

where

$$g'_n := h \sum_{j=0}^{r} \beta_j b_{n+j} + \tilde{g}_n, \quad n = 0, \ldots, N-r.$$

Now the inhomogeneous difference equation is 'non-autonomous' in the sense that our perturbation to the step map T depends on n via λ_n. This is not actually a significant issue. The more annoying thing is that the RHS of (10.7) now includes a garbage term b_n that depends on the solution $\{x_n\}$ (or, equivalently, on $\{\epsilon_n\}$), via the dependency $x_n \to \{d_n, \lambda_n\} \to b_n$, and we have no *a priori* control over it. The fact that the that the b_n are quadratically small (cf.(10.6)) in the error will allow us to massage away this difficulty via fixed-point iteration.

We will define a map that takes in a candidate solution $\{y_n\}$ and spits out an updated candidate $\{y'_n\}$ such that the map has fixed point $\{x_n\}$. First the map defines g_n in terms of $\{y_n\}$ as we defined g'_n terms of $\{x_n\}$ above. Then the map solves the appropriate difference equation for an error displacement $\{\delta_n\}$, which in turn determines a new candidate solution $\{y_n\}$. We want to obtain control over both stages of this map. This is accomplished in the following two lemmas.

Lemma 25. *The map $\mathcal{G} = \mathcal{G}^{(N)} : y_{0:N} \mapsto g_{0:N-r}$ defined by*

$$g_n = h \sum_{j=0}^{r} \beta_j \left[f(y_{n+j}, t_{n+j}) - f(x(t_{n+j}), t_{n+j}) - \partial_x f(x(t_{n+j}), t_{n+j}) \cdot (y_{n+j} - x(t_{n+j})) \right] + \tilde{g}_n$$

is Lipschitz with respect to the infinity norm. If we restrict the map $\mathcal{G}$ to the ball

$$B_r := B_r^{(N)} = \{ y_{0:N} \, : \, \|y_{0:N} - x(t_{0:N})\|_\infty \leq r \},$$

the Lipschitz constant, which we call $L_{h,r}$, is $O(rh)$.

Proof (of lemma). First compute

$$[\mathcal{G}(y_{0:N}) - \mathcal{G}(z_{0:N})]_n = h \sum_{j=0}^{r} \beta_j \left[f(y_{n+j}, t_{n+j}) - f(z_{n+j}, t_{n+j}) - \partial_x f(x(t_{n+j}), t_{n+j}) \cdot (y_{n+j} - z_{n+j}) \right].$$

In order to apply Lagrange error bounds we rewrite as

$$\begin{aligned} [\mathcal{G}(y_{0:N}) - \mathcal{G}(z_{0:N})]_n = h \sum_{j=0}^{r} \beta_j \left[f(y_{n+j}, t_{n+j}) - f(z_{n+j}, t_{n+j}) - \partial_x f(z_{n+j}, t_{n+j}) \cdot (y_{n+j} - z_{n+j}) \right] \\ + \; h \sum_{j=0}^{r} \beta_j \left[\partial_x f(x(t_{n+j}), t_{n+j}) - \partial_x f(z_{n+j}, t_{n+j}) \right] \cdot (y_{n+j} - z_{n+j}). \end{aligned}$$

Now

$$f(y_n, t_n) - f(z_n, t_n) - \partial_x f(z_n, t_n) \cdot (y_n - z_n) = O(|y_n - z_n|^2)$$

and

$$\partial_x f(x(t_n), t_n) - \partial_x f(z_n, t_n) = O(|x(t_n) - z_n|)$$

by the assumption of uniform boundedness on $\partial_x^2 f$, so it follows that

$$\|\mathcal{G}(y_{0:N}) - \mathcal{G}(z_{0:N})\|_\infty \leq Ch \left(\|y_{0:N} - z_{0:N}\|_\infty + \|x(t_{0:N}) - z_{0:N}\|_\infty \right) \|y_{0:N} - z_{0:N}\|_\infty$$

for a suitable constant C independent of n, N. Now

$$\|y_{0:N} - z_{0:N}\|_\infty \leq \|y_{0:N} - x(t_{0:N})\|_\infty + \|z_{0:N} - x(t_{0:N})\|_\infty,$$

so if $y_{0:N}, z_{0:N} \in B_r$, then

$$\|\mathcal{G}(y_{0:N}) - \mathcal{G}(z_{0:N})\|_\infty \leq 3Chr \|y_{0:N} - z_{0:N}\|_\infty,$$

and the lemma is proved. □

Lemma 26. *Suppose that $\lambda_n = \lambda_n^{(N)} = O(1)$ uniformly in n and N. (Superscript dependence on N will be omitted from the notation for visual clarity.) Then the linear map $\mathcal{S} = \mathcal{S}^{(N)}$ from the starting values and RHS $(\delta_{0:r-1}, g_{0:N-r})$ to the full solution $\delta_{0:N}$ of*

$$\sum_{j=0}^{r} (\alpha_j - \lambda_{n+j}\beta_j h)\, \delta_{n+j} = g_n, \quad n = 0, \ldots, N - r \tag{10.8}$$

is bounded uniformly in N in the sense that there exists C indpendent of N such that

$$\|\mathcal{S}[\delta_{0:r-1}, g_{0:N-r}]\|_\infty \leq C \left(\|\delta_{0:r-1}\|_\infty + N \|g_{0:N-r}\|_\infty \right).$$

Proof (of lemma). The solution of (10.8) proceeds by analogy to (10.4), except that the perturbed step map is now time-dependent, $T_n := T + B_n$, where the B_n are linear maps (implicitly h-dependent) with $\max_n \|B_n\| = O(h)$. Indeed, with notation as in the preceding (*mutatis mutandi*) we obtain

$$\boldsymbol{\delta}^{(n)} = \sum_{k=0}^{n} T_n \cdots T_{k+2} T_{k+1}(\mathbf{g}^{(k)}), \quad n = 0, \ldots, N - r + 1,$$

and we can straightforwardly generalize our preceding analysis to obtain the bound

$$|||\boldsymbol{\delta}^{(n)}||| \leq \tilde{C} |||\mathbf{g}^{(0)}||| + \tilde{C} N \max_{k=1,\ldots,N-r+1} |||\mathbf{g}^{(k)}|||$$

for some $\tilde{C}$ indpendent of n, N. Recall the notation $\mathbf{g}^{(0)} = (\delta_0, \ldots, \delta_{r-1})$. By norm equivalence the lemma follows. □

Proof of Theorem 21. Define the map

$$\Phi = \Phi^{(N)} : y_{0:N} \mapsto x(t_{0:N}) + \mathcal{S}\left[\epsilon_{0:r-1}, \mathcal{G}(y_{0:N})\right].$$

From the preceding two lemmas it follows that

$$\begin{aligned} \Phi(y_{0:N}) - \Phi(z_{0:N}) &= \mathcal{S}[\epsilon_{0:r-1}, \mathcal{G}(y_{0:N})] - \mathcal{S}[\epsilon_{0:r-1}, \mathcal{G}(z_{0:N})] \\ &= \mathcal{S}[0, \mathcal{G}(y_{0:N}) - \mathcal{G}(z_{0:N})], \end{aligned}$$

so

$$\begin{aligned}\|\Phi(y_{0:N}) - \Phi(z_{0:N})\|_\infty &\leq CN\|\mathcal{G}(y_{0:N}) - \mathcal{G}(z_{0:N})\|_\infty \\ &\leq CNL_{r,h}\|y_{0:N} - z_{0:N}\|_\infty,\end{aligned}$$

as long as $y_{0:N}, z_{0:N} \in B_r$. Now $CNL_{r,h} = O(r)$, so let r be sufficiently small (*independent of* N) such that Φ is Lipschitz on B_r with Lipschitz constant $1/2$.

We claim that then for h sufficiently small, Φ maps B_r into itself. To see this, suppose $y_{0:N} \in B_r$. Then we want to show that $\|\Phi(y_{0:N}) - x(t_{0:N})\|_\infty \leq r$. First we can apply the triangle inequality:

$$\begin{aligned}\|\Phi(y_{0:N}) - x(t_{0:N})\|_\infty &\leq \|\Phi(y_{0:N}) - \Phi(x(t_{0:N}))\|_\infty + \|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty \\ &\leq \frac{1}{2}\|y_{0:N} - x(t_{0:N})\| + \|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty \\ &\leq \frac{r}{2} + \|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty. \end{aligned} \tag{10.9}$$

But

$$\begin{aligned}\|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty &= \|\mathcal{S}[\epsilon_{0:r-1}, \mathcal{G}(x(t_{0:N}))\|_\infty \\ &\leq C\left(\|\epsilon_{0:r-1}\|_\infty + N\|\tilde{g}_n\|_\infty\right),\end{aligned}$$

from which it follows that

$$\|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty = O(h^p). \tag{10.10}$$

Then by plugging into (10.9), we see that $\|\Phi(y_{0:N}) - x(t_{0:N})\|_\infty \leq r$, i.e., $\Phi(y_{0:N}) \in B_r$, provided h is sufficiently small, and the claim that Φ maps B_r into itself is proved.

Then the Banach fixed point theorem (Theorem 3) furnishes the existence of a unique fixed point $x^\star_{0:N}$ of Φ in B_r. But such a fixed point is precisely a solution of (5.6), hence by uniqueness is is our LMM solution $x_{0:N}$.

Moreover, since $x(t_{0:N}) \in B_r$, Banach's fixed point theorem allows us to write

$$x_{0:N} = \lim_{k\to\infty} \Phi^k(x(t_{0:N})),$$

and by applying the triangle inequality to a telescoping sum obtain

$$\begin{aligned}\|x_{0:N} - x(t_{0:N})\|_\infty &\leq \sum_{k=0}^{\infty} \|\Phi^{k+1}(x(t_{0:N})) - \Phi^k(x(t_{0:N}))\|_\infty \\ &\leq \sum_{k=0}^{\infty} \frac{1}{2^k}\|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty \\ &= 2\|\Phi(x(t_{0:N})) - x(t_{0:N})\|_\infty,\end{aligned}$$

which controls the distance between our LMM solution and the true solution in terms of the error incurred by a single application of the map Φ. We have already bounded this error in (10.10), which implies

$$\|x_{0:N} - x(t_{0:N})\|_\infty = O(h^p),$$

completing the proof of the theorem. □

11 Milne's device and predictor-corrector methods

Milne's device refers to a trick that allows us to use two LMMs of the same order of accuracy to obtain an *a posteriori* estimate of the local truncation error, which may be far tighter than any *a priori* bound we could derive on paper. The device also allows us to extrapolate a better solution of one higher order of accuracy. The idea of Milne's device can be applied in the specific context of a predictor-corrector method, which can also be viewed as an approximation to the full solution of an implicit method, equipped via Milne's device with an *a posteriori* estimate on the local error.

11.1 Milne's device

Suppose we have two r-step LMMs defined via coefficients α_j, β_j and $\tilde{\alpha}_j, \tilde{\beta}_j$, respectively, for $j = 0, \ldots, r$. (As we shall see, sometimes it is convenient to view an $(r-1)$-step LMM as an r-step LMM by padding the coefficients with zeros for $j = 0$.) We will denote the solutions of these by $\{x_n\}$ and $\{\tilde{x}_n\}$, and for the purpose of estimating the error that we accrue in a *single step* of the LMM, assume that $x_j = \tilde{x}_j = x(t_j)$ for $j = n, \ldots, n+r-1$.

Then using the fact that

$$\mathscr{L}_h x(t) = ch^{p+1}x^{(p+1)}(t) + O(h^{p+2}),$$

where $c := C_{p+1} \neq 0$ is the error constant of the first LMM, and subtracting equations from the (5.6), we see that

$$x(t_{n+r}) - x_{n+r} = h\left[f(x(t_{n+r}), t_{n+r}) - f(x_{n+r}, t_{n+r})\right] + ch^{p+1}x^{(p+1)}(t) + O(h^{p+2}),$$

but since $x(t_{n+r}) = x_{n+r} + O(h^{p+1})$, the first term is merely $O(h^{p+2})$, and we have

$$x(t_{n+r}) - x_{n+r} = ch^{p+1}x^{(p+1)}(t) + O(h^{p+2}),$$

so

$$\epsilon_{n+r} \approx -ch^{p+1}x^{(p+1)}(t),$$

where from now on, '$\approx$' indicates equality up to $O(h^{p+2})$ error.

Note that even though we can compute the error constant c *a priori*, we cannot compute $x^{(p+1)}(t)$ *a priori* since we don't know the exact solution! So the leading order contribution to the error is unknown to us, practically. However, if we let $\tilde{c}$ denote the error constant of the second LMM, we also have

$$x(t_{n+r}) - \tilde{x}_{n+r} \approx \tilde{c}h^{p+1}x^{(p+1)}(t),$$

and we can subtract equations to obtain

$$\tilde{x}_{n+r} - x_{n+r} \approx (c - \tilde{c})h^{p+1}x^{(p+1)}(t) \approx (c - \tilde{c})\left(\frac{-\epsilon_{n+r}}{c}\right).$$

Assuming that $c \neq \tilde{c}$, we can then solve for

$$\epsilon_{n+r} \approx \frac{c}{c - \tilde{c}}\left(x_{n+r} - \tilde{x}_{n+r}\right),$$

and likewise

$$\tilde{\epsilon}_{n+r} \approx \frac{\tilde{c}}{\tilde{c} - c}(\tilde{x}_{n+r} - x_{n+r}).$$

We can use these formulas to estimate the error accumulated in one time step of either LMM. A natural criterion to meet is

$$\epsilon_{n+r} \leq h\delta,$$

where δ is some target tolerance for the error accumulation per unit time. If we do not meet the tolerance, we decrease h until we do. Adjusting the step size on the fly is annoying for LMMs but can be achieved via polynomial interpolation of previously computed values.

Note that since $x(t_{n+r}) = x_{n+r} - \epsilon_{n+r}$ we can extrapolate an improved solution which now enjoys one order higher of LTE via

$$\hat{x}_{n+r} := x_{n+r} + \frac{c}{c - \tilde{c}} \left(\tilde{x}_{n+r} - x_{n+r} \right).$$

This procedure, known as ***local extrapolation***, is harder to analyze and may be a bit risky from the point of view of stability.

11.2 Predictor-corrector methods

Now for something completely different (but not really). Suppose we want to implement an implicit LMM such as the r-step Adams-Moulton method but we are lazy and don't want to solve a bunch of nonlinear equations. Suppose this method is p-th order accurate, and recall we need to solve

$$x_{n+r} = -\sum_{j=0}^{r-1} \alpha_j x_{n+j} + h \sum_{j=0}^{r-1} \beta_j f_{n+j} + h\beta_j f(x_{n+r}, t_{n+r})$$

for x_{n+r}. The difficulty is the x_{n+r} appearing on the right-hand side. If we just swapped it out for a *different* p-th order accurate guess $\tilde{x}_{n+r}$ for $x(t_{n+r})$, obtained by alternative means, then the implicit method would become explicit. If our original LMM is the $(r-1)$-step Adams-Moulton method, we could use, for example, the r-step Adams-Bashforth method to obtain the guess $\tilde{x}_{n+r}$. In this case, the Adams-Bashforth method would be playing the role of ***predictor***, and the update

$$x_{n+r} = -\sum_{j=0}^{r-1} \alpha_j x_{n+j} + h \sum_{j=0}^{r-1} \beta_j f_{n+j} + h\beta_j f(\tilde{x}_{n+r}, t_{n+r})$$

defines a so-called ***predictor-corrector method***.

Although the resulting method is not an LMM, we can still define the LTE as the amount by which the true solution $\{x(t_n)\}$ fails to the satisfy the scheme. By exchanging $f(x_{n+r}, t_{n+r}) \approx f(\tilde{x}_{n+r}, t_{n+r})$, since the LTE for $\tilde{x}_{n+r}$ is already only $O(h^{p+1})$, we see that the predictor-corrector method has an LTE that differs from that of the original implicit LMM by $O(h^{p+2})$. *Therefore, both the order and the error constant of the original method and the predictor-corrector method are the same!*

It follows that Milne's device (applied to the pair of predictor and predictor-corrector methods) can be used just as in the preceding subsection, furnishing a local error estimate for the predictor-corrector method.

It's nice to have an error estimate, but why bother with an implicit method at all? So far we don't really have any theoretical tools to distinguish between methods with the same order of accuracy. The notion of absolute stability, which addresses this gap, will be the topic of the next section. (Implicit methods can offer better stability properties than explicit methods, and predictor-corrector methods can split the difference.)

12 Stiff systems and absolute stability

Consider the scalar ODE

$$x'(t) = a(\cos(t) - x(t)), \tag{12.1}$$

where $a > 0$. Intuitively we can think of the equation as driving $x(t)$ toward the fixed pattern $\cos(t)$, and the time it takes for the solution to relax to this pattern become smaller as $a \to \infty$.

In fact, we can solve the ODE exactly as

$$x(t) = \frac{a\sin(t)}{a^2+1} + \frac{a^2\cos(t)}{a^2+1} + ce^{-at},$$

where c is a constant determined by the initial condition. If the initial condition is $x_0 = 0$, then we can solve for $c = -\frac{a^2}{a^2+1}$, hence

$$x(t) = \frac{a}{a^2+1}\left(\sin(t) + a\cos(t) - ae^{-at}\right).$$

Evidently as $t \to \infty$, the solution relaxes with exponential rate a to a sinusoidal pattern with fixed frequency. Moreover, in the limit $a \to \infty$, the time scales of oscillation and relaxation separate, and we see that $x(t) \to \cos(t)$.

As we approach the limit $a \to \infty$, this becomes a prototypical example of a ***stiff*** system. Colloquially such systems are systems with several internal time scales of different orders of magnitude. Many schemes (and all explicit schemes) collapse completely when presented with a stiff system, unless the time step is taken to be as small as the fastest time scale inherent to the system. But if we really don't care about resolving the fastest time scales, we may be able to get away with a much longer time step. The notion of absolute stability addresses this capacity.

I lied....The real prototypical example of a stiff system is a poorly conditioned linear system, let's see how it relates to (12.1). Consider the $y(t) = x(t) - \cos(t)$, which satisfies the ODE

$$y'(t) = -ay(t) + \sin(t),$$

with an inhomogeneous contribution $\sin(t)$ that is now independent of a. As we have seen conceptually in the proof of the Dahlquist theorem, to understand the propagation of error for a linear ODE with an inhomogeneous contribution, it really suffices to consider the homogeneous ODE

$$y'(t) = -ay(t).$$

The general study of linear systems of ODEs can essentially be reduced to this scalar case via diagonalization, as we now describe.

12.1 Linear systems of ODEs

Consider a general linear system of ODEs

$$x'(t) = Ax(t).$$

In fact an explicit solution is available as

$$x(t) = \exp(At)x_0,$$

where $\exp(\,\cdot\,)$ denotes the matrix exponential. This solution can be verified by term-by-term differentiation of the power series expansion for the matrix exponential.

A more concrete representation of the solution can be obtained by considering a Jordan decomposition for A. For simplicity, let's just assume the generic condition that A is diagonalizable (though possibly only over the complex numbers), so $A = P\Lambda P^{-1}$, where P is invertible and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ is diagonal, though both are possibly complex. Then by considering the transformation $y(t) = P^{-1}x(t)$ (which operation also commutes with any reasonable scheme), we obtain

$$y'(t) = P^{-1}x'(t) = P^{-1}Ax(t) = \Lambda P^{-1}x(t) = \Lambda y(t),$$

i.e.,

$$y'(t) = \Lambda y(t).$$

Note that the dynamics for $y_1, \ldots, y_d$ decouple entirely after this transformation as

$$y_i'(t) = \lambda_i y_i(t),$$

which can be solved exactly as

$$y_i(t) = e^{\lambda_i t} y_i(0).$$

It follows that $\lim_{t\to\infty} x(t) = 0$ if and only if $\mathrm{Re}(\lambda_i) < 0$ for all $i = 1, \ldots, d$. In general, even if A is not diagonalizable, it can be verified that $\lim_{t\to\infty} x(t) = 0$ if and only if $\mathrm{Re}(\lambda) < 0$ for all eigenvalues λ of A.

12.2 Absolute stability

Ideally we want numerical schemes that conserve this long-time behavior. We can always imagine the solution of a LMM as being defined for arbitrarily many time steps with no final time. As such we can view any r-step LMM as furnishing a sequence $\{x_n\}_{n=0}^{\infty}$, given only the start values $x_0, \ldots, x_{r-1}$.

Definition 27. We say that a LMM is ***absolutely stable*** (for given h, λ) if the solution $\{x_n\}_{n=0}^{\infty}$ satisfies $\lim_{n\to\infty} x_n = 0$ whenever the LMM is applied with time step h to the *scalar* ODE $x'(t) = \lambda x(t)$ with arbitrary starting values. We will say that an LMM is ***weakly absolutely stable*** if the sequence $\{x_n\}_{n=0}^{\infty}$ remains bounded.

We can unpack this a bit. The LMM in this case reads as

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = h\lambda \sum_{j=0}^{r} \beta_j x_{n+j},$$

or equivalently

$$\sum_{j=0}^{r} \left(\alpha_j - \hat{h}\beta_j\right) x_{n+j} = 0,$$

where we have defined $\hat{h} = h\lambda$.

Remark 28. In our discussion of absolute stability h and λ will always appear together jointly within the expression $h\lambda$, so it is useful to think of $\hat{h}$ as the essentially important quantity, rather than h and λ individually. Hence we will say that an LMM is *absolutely stable for given* $\hat{h}$ in the future.

Our previous analysis of this kind of difference equation yields the result that $\lim_{n\to\infty} x_n = 0$ for all choices of starting values if and only if the 'step map'

$$T_{h\lambda}(x_0, \ldots, x_{r-1}) = \left(x_1, \ldots, x_{r-1}, -\sum_{j=0}^{r-1} \frac{\alpha_j - \hat{h}\beta_j}{1 - \hat{h}\beta_r} x_j \right)$$

has eigenvalues *strictly less than one in magnitude*.

Note that this step map is the same as the one considered before, with coefficients

$$\tilde{\alpha}_j := \frac{\alpha_j - \hat{h}\beta_j}{1 - \hat{h}\beta_r}$$

in the place of the α_j defining T.

Hence the eigenvalues are precisely the roots of the polynomial $\sum_{j=0}^{r} \tilde{\alpha}_j w^j$, which are the same (multiplying through by $1 - \hat{h}\beta_r$) as the roots of the ***stability polynomial*** $p = p_{\hat{h}}$ defined by

$$p(w) := \sum_{j=0}^{r} (\alpha_j - \hat{h}\beta_j) w^j.$$

Hence we have shown:

Theorem 29. *An LMM is absolutely stable for given $\hat{h}$ if and only if all the roots of the corresponding stability polynomial are strictly less than one in magnitude. Meanwhile it is weakly absolutely stable if the stability polynomial satisfies the root condition.*

Definition 30. The ***region of*** (***weak***) ***absolute stability*** for an LMM is the set of values $\hat{h} \in \mathbb{C}$ for it is (weakly) absolutely stable. The ***interval of absolute stability*** for an LMM is the largest interval of the form $(\hat{h}_0, 0)$ where $\hat{h}_0 < 0$ such that the LMM is absolutely stable on the entire interval.

Remark 31. Note that an LMM is zero-stable if and only if the region of weak absolute stability contains the origin.

Exercise: compute the absolute stability regions for explicit and implicit Euler, the trapezoidal rule, and the midpoint/leapfrog method.

12.3 Computing arbitrary absolute stability regions

Although absolute stability regions can be computed analytically for a few methods (cf. the preceding exercise), in general it is not possible to do so. There is nonetheless a foolproof computational approach for determining absolute stability regions called the ***boundary locus method***. This approach, as the name suggests, directly seeks the boundary of the absolute stability region, which is defined by the condition that at least one of the roots w of the stability polynomial $p_{\hat{h}}$ has modulus $|w| = 1$. Any such root can be written $w = e^{i\theta}$ for $\theta \in [0, 2\pi)$.

Therefore, sweeping over $\theta \in [0, 2\pi)$, we plug $w = e^{i\theta}$ into $p_{\hat{h}}(w) = 0$ and then solve for $\hat{h} = \hat{h}(\theta)$, which we can view as a function of θ. The map $\theta \mapsto \hat{h}(\theta)$ defines a curve in $\mathbb{C}$, called the boundary locus. The curve may self-intersect, dividing the complex plane into possibly several connected components. For $\hat{h}$ in the interior of each connected component, no roots of $p_{\hat{h}}$ have modulus $|w| = 1$, and we can check whether that component belongs to the absolute stability region just by testing a single point.

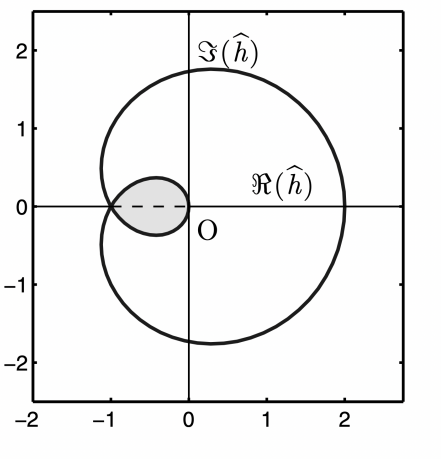

Figure 12.1: The solid curve is the boundary locus for (12.2), i.e., the set of points $\hat{h}$ such that the stability polynomial $p_{\hat{h}}$ has a root of modulus 1. The shaded region is the absolute stability region.

For example, consider the LMM

$$x_{n+2} - x_{n+1} = hf_n. \tag{12.2}$$

The stability polynomial is

$$p_{\hat{h}}(w) = w^2 - w - \hat{h}.$$

Plugging $w = e^{i\theta}$ into the equation $p_{\hat{h}}(w) = 0$ and solving for $\hat{h}$:

$$\hat{h}(\theta) = e^{2i\theta} - e^{i\theta}.$$

This is already something we can plot (see Figure 12.3), though it is of course possible to write out the real and imaginary parts of $\hat{h}(\theta)$ separately.

Note: *In general, since $p_{\hat{h}}(w)$ depends linearly on $\hat{h}$, it is always possible to solve uniquely for $\hat{h} = \hat{h}(\theta)$.*

Evidently the curve $\theta \mapsto \hat{h}(\theta)$ depicted in Figure 12.3 divides the complex plane into three connected components, so to determine the absolute stability region is suffices to check one point in the interior of each for absolute stability and then shade accordingly.

Remark 32. Note that determining weak absolute stability regions is technically a bit more subtle, since each point on the boundary locus must be checked for the presence of repeated roots of modulus 1.

12.4 A-stability

Definition 33. An LMM is ***A-stable*** if its region of absolute stability includes the entire (strict) left half plane $\{\hat{h} \in \mathbb{C} : \mathrm{Re}(\hat{h}) < 0\}$.

A-stability is desirable because it implies (for a linear system of ODEs) that if the true solution satisfies $x(t) \to 0$ as $t \to \infty$, then the scheme solution satisifies the analogous condition that $x_n \to 0$ as $n \to \infty$.

We have seen (through exercises above) that implict Euler and the trapezoidal rule are both A-stable. (Their absolute stability regions are $\{\hat{h} : |\hat{h} - 1| > 1\}$ and $\{\hat{h} : \mathrm{Re}(\hat{h}) < 1\}$, respectively.)

A-stability is actually an extremely restrictive condition! To wit:

Theorem 34 (Dahlquist's second barrier theorem)**.**

1. *Any explicit LMM is not A-stable.*
2. *The order of accuracy of an A-stable LMM is at most* 2*.*

Remark 35. As verified in our above exercise, the absolute stability region of the *trapezoidal rule* is precisely $\{\hat{h} : \mathrm{Re}(\hat{h}) < 0\}$, hence the trapezoidal rule is in particular A-stable. Also recall that the trapezoidal rule is order-2 accurate. It is not the unique order-2 accurate A-stable LMM, but it is the best one in a sense that is also clarified by Dahlquist's second barrier theorem, though omitted here. In summary, if you demand A-stability and second-order accuracy, use the trapezoidal rule. In the setting of PDEs, the trapezoidal rule defines the *Crank-Nicolson scheme*, which is essentially the default approach for parabolic PDE such as the heat equation. A-stability is an important requirement in this case because the discretization of the heat equation becomes infinitely stiff as it is refined.

The proof is outside the scope of the course.

12.5 A_0-stability

Definition 36. An LMM is A_0***-stable*** if its region of absolute stability includes the entire (strictly) negative real axis $\{\hat{h} \in \mathbb{C} : \mathrm{Re}(\hat{h}) < 0,\ \mathrm{Im}(\hat{h}) = 0\}$.

The condition of A_0-stability is looser than A-stability and Dahlquist's second barrier theorem does not preclude higher-order A_0-stable schemes. Indeed, as you can verify using the technique introduced below, the zero-stable BDF methods (i.e., the BDF methods of order up to 6) are all A_0-stable, though they are only A-stable up to order 2.

12.6 Absolute stability and linear systems of ODEs

The following theorem guarantees that the notion of absolute stability for the scalar test problem can be lifted to linear systems of ODEs.

Theorem 37. *Assume that A is diagonalizable, and let $\{x_n\}_{n=0}^{\infty}$ be furnished by a LMM with step size h such that $h\lambda$ lies in the region of absolute stability for all eigenvalues λ of A. Then $\lim_{n\to\infty} x_n = 0$.*

Proof. Write our LMM as

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = h \sum_{j=0}^{r} \beta_j f_{n+j}.$$

In our case we have

$$\sum_{j=0}^{r} \alpha_j x_{n+j} = h \sum_{j=0}^{r} \beta_j A x_{n+j}. \tag{12.3}$$

Diagonalize $A = P\Lambda P^{-1}$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$. By assumption $h\lambda_i$ lies in the region of absolute stability.

Then left-multiplying (12.3) by P and writing $I = P^{-1}P$ yields

$$\sum_{j=0}^{r} \alpha_j P x_{n+j} = h \sum_{j=0}^{r} \beta_j P A P^{-1} P x_{n+j},$$

i.e.,

$$\sum_{j=0}^{r} \alpha_j y_{n+j} = h \sum_{j=0}^{r} \beta_j \Lambda y_{n+j}$$

where $y_n := Px_n$. It is useful further to define the entries via $y_n = (y_{n,1}, \ldots, y_{n,d})$.

We see that $\{y_n\}_{n=0}^{\infty}$ solves the same LMM applied to the system $y'(t) = \Lambda y$. It follows that $\{y_{n,i}\}_{n=0}^{\infty}$ solves the same LMM applied to the *scalar* ODE $y_i'(t) = \lambda_i y_i$ for each i. But since $h\lambda_i$ lies in the region of absolute stability, it follows that $\lim_{n\to\infty} y_{n,i} = 0$ as $n \to \infty$ for each i, hence $\lim_{n\to\infty} y_n = 0$, hence $\lim_{n\to\infty} x_n = 0$. □

12.7 Stiff systems

The theorem suggests that in order to guarantee absolute stability ***for a given linear system of ODEs*** (defined by the matrix A), we must choose h sufficiently small such that $h\lambda$ lies in the region of absolute stability region *for all eigenvalues λ of A*.

Consider the case, for example, where all the eigenvalues λ of A satisfy $\mathrm{Re}(\lambda) < 0$. In this case we hope for our scheme to be absolutely stable for the corresponding linear system of ODEs. If we order the eigenvalues increasingly (with multiplicity) by real part, we see that the fastest timescale present in the problem is $1/\mathrm{Re}(-\lambda_1)$, while the slowest timescale is $1/\mathrm{Re}(-\lambda_d)$. In the case where the ***stiffness ratio***

$$S := \frac{\mathrm{Re}(\lambda_1)}{\mathrm{Re}(\lambda_d)} \gg 1,$$

then we say that our system is ***stiff*** (though for general nonlinear systems, such a notion is tricky to define precisely), due to the simultaneous presence of timescales that differ by many orders of magnitude.

If our scheme is not A-stable, then in general we will have to scale $h \sim \frac{1}{\mathrm{Re}(-\lambda_1)}$, i.e., on the order of the smallest timescale of the problem, in order to guarantee stability, while we might for example only care about features of the solution on the longest timescale $T \sim \frac{1}{\mathrm{Re}(-\lambda_d)}$. In this case, we will need to run our scheme for $N \sim S$ time steps in order to resolve the features of interest without sacrificing stability.

Differential equations won't help you much in the design of aeroplanes—not yet, anyhow.

N. Shute

Part III
Runge-Kutta methods

Runge-Kutta (RK) methods are the major alternative to LMMs. They are all one-step methods in the sense that the next value x_{n+1} is produced using only x_n and none of the preceding values $x_{n-1}, x_{n-2}, \ldots$.

However, RK methods are ***multi-stage*** methods in that several 'intermediate values' for $f(x(t))$, $t \in [t_n, t_{n+1}]$ are produced before the ultimate value x_{n+1} is computed using a weighted average of them.

13 The general RK method

In general, a RK method can be expressed as

$$x_{n+1} = x_n + h \sum_{i=1}^{s} b_i k_i, \tag{13.1}$$

where s is the number of ***stages***, and $k_1, \ldots, k_s \in \mathbb{R}^d$ are defined via the equations

$$k_i = f\left(x_n + h \sum_{j=1}^{s} a_{ij} k_j,\ t_n + c_i h \right),\ i = 1, \ldots, s. \tag{13.2}$$

Thus an RK method is completely specified by the number s of stages, the $s \times s$ ***RK matrix*** $A = (a_{ij})$, the s-dimensional vector of ***weights*** $b = (b_i)$, and the s-dimensional vector of ***nodes*** $c = (c_i)$. The data specifying an RK method can be collected in a ***Butcher tableau*** as:

$$\begin{array}{c|c} c & A \\ \hline & b^\top \end{array}$$

Intuitively the values $x_{n,i} := x_n + h \sum_{j=1}^{s} a_{ij} k_j$ can often be viewed as intermediate guesses for the value of $x(t)$ at intermediate times $t_{n,i} := t_n + c_i h$. Under this view the values k_i are guesses for the slopes $f(x(t))$ at intermediate time $t_{n,i}$.

Assuming the existence of a unique or canonically defined solution of (13.2) (a system of $s \times d$ nonlinear equations in $s \times d$ unknowns), the RK method (13.1)-(13.2) can be thought of as specifying a map $\mathcal{R} : \mathbb{R}^d \to \mathbb{R}^d$ (*not a standard notation!*) which sends $x_n \mapsto x_{n+1}$.

13.1 Sum rule

It is natural (but *not necessary*) to impose the condition

$$c_i = \sum_{j=1}^{s} a_{ij}, \text{ or } c = A\mathbf{1} \tag{13.3}$$

because this guarantees that the answer yielded by the RK method for a non-autonomous system is independent of whether we choose to

1. solve directly, or
2. first reduce to the autonomous case via the reduction described in Section 1.1 (**exercise**), then solve the resulting autonomous system, and finally recover the solution to the original system from this solution.

Indeed, imagine for the sake of argument that $k_j = k$ for some k and all $j = 1, \ldots, s$. Then under the condition 13.3, $x_{n,i} = x_n + (c_i h)k$, which makes intuitive sense if we view $x_{n,i}$ as the solution at time $t_{n,i} = t_n + c_i h$.

13.2 Explicit RK methods

In general 13.2 specifies a system of $s \times d$ nonlinear equations in $s \times d$ unknowns. Note that when h is small, the equations are a small perturbation of the trivial system of equations $k_i = f(x_n)$, $i = 1, \ldots, s$, and implicit function theorem arguments should reassure us.

However, it is worthwhile to consider the special case of ***explicit RK methods***, where it is not necessary to solve equations at all because each k_i can be computed directly in terms of only the k_j for $j < i$. Evidently, to ensure this, we must have $a_{ij} = 0$ for $j \geq i$, i.e., the RK matrix must be ***strictly lower triangular***.[1] (Technically the RK matrix only needs to be strictly lower triangular after some permutation of the indices, but we can assume this WLOG.)

In this case, if we insist on the sum rule 13.3, we must have $c_1 = 0$. This is not a necessary requirement for explicit methods, but we will always assume it.

13.3 Local truncation error, consistency, and convergence

The ***local truncation error*** (***LTE***) for a RK method can be defined like before as the discrepancy in the defining equations that occurs when we plug in the true solution $x(t)$. The precise definition here is more reminiscent of the case of Taylor series methods since those are also one-step methods.

Indeed, consider taking $x_n := x(t_n)$, and then determine $x_{n+1} = \mathcal{R}(x_n)$ from x_n according to the RK method (13.1)-(13.2). The LTE τ_n at the n-th time step is then defined as the discrepancy between the true solution $x(t_{n+1})$ and the computed solution x_{n+1} at time t_{n+1}, i.e.,

$$\tau_n = x(t_{n+1}) - \mathcal{R}(x(t_n)). \tag{13.4}$$

We say that a RK method is consistent if the LTE satisfies

$$\max_{n=0,\ldots,N-1} |\tau_n| = O(h^2)$$

[1] The case where the RK matrix is merely lower triangular corresponds to the category of ***diagonally implicit RK***, or ***DIRK***, methods, which are implicit but only require the solution of a sequence of s systems of d equations, rather than a fully general system $s \times d$ equations.

as $h \to 0$ and more generally order-p consistent if

$$\max_{n=0,\ldots,N-1} |\tau_n| = O(h^{p+1})$$

as $h \to 0$.

In fact, since RK methods are one-step methods, consistency implies convergence, and moreover order-p consistency implies order-p accuracy. The proof is analogous to the proof for Taylor series methods and is left as an **exercise**. In other words, there is ***no concern about zero-stability*** analogous to the concern for LMMs. On the other hand, ***absolute stability*** remains an important concept and can be defined similarly. We shall return to this later.

It is in fact not too hard to see (**exercise)** that consistency is equivalent to the condition that $\sum_{i=1}^{s} b_i = 1$, which shall henceforth be assumed.

13.4 Simple example: modified Euler

One-stage RK methods are not very exciting, so our introductory example is 2-stage RK method, namely the ***modified Euler method***. As an explicit RK method satisfying the sum rule (13.3), it must be specified by a Butcher tableau of the form:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ a & a & 0 \\ \hline & 1-b & b \end{array}$$

Specifically it corresponds to the choice $a = 1/2$, $b = 1$:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1/2 & 1/2 & 0 \\ \hline & 0 & 1 \end{array}$$

Concretely, this yields the update equations

$$\begin{aligned} k_1 &= f(x_n, t_n) \\ k_2 &= f(x_n + hk_1/2, t_n + h/2) \\ x_{n+1} &= x_n + hk_2, \end{aligned}$$

which allow us to determine x_{n+1} directly, given x_n.

You can think of modified Euler as attempting to get an estimate of $f(x(t))$ at $t = t_n + h/2$ and then using this estimate at the midpoint of the interval $[t_n, t_{n+1}]$ to advance the state.

We will see shortly that modified Euler is second-order accurate, and moreover that given the choice $a = 1/2$, the further choice $b = 1$ is the unique choice guaranteeing second-order accuracy.

14 Designing higher-order explicit schemes

We are going to get a taste of how to derive explicit RK schemes of arbitrary orders of accuracy. Since we will assume the sum rule (13.3), by the preceding remarks it will suffice to consider the case where f does not depend on t, i.e., where (abusing notation slightly) $f(x, t) = f(x)$.

14.1 One-stage methods

Any explicit one-stage method is simply of the form

$$x_{n+1} = x_n + bhf_n.$$

However, consistency demands that $b = 1$, hence we have simply recovered Euler's method, which we know to be first-order accurate.

14.2 Two-stage methods

Let us leave a and b undetermined in our preceding discussion of the modified Euler method in Section 13.4.

In order to estimate the LTE (13.4), it is useful to write $x(t_{n+1})$ via Taylor series expansion of $x(t)$ about $t = t_n$, truncating at order high enough to detect the order of consistency.

$$x(t_{n+1}) = x(t_n) + hx'(t_n) + \frac{h^2}{2}x''(t_n) + O(h^3).$$

Now recall

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

so differentiating through the ODE we obtain

$$x''(t) = Df(x(t)) \cdot x'(t) = Df(x(t))f(x(t)),$$

and we can substitute to obtain

$$x(t_{n+1}) = x_n + hf_n + \frac{h^2}{2}f_n^{(1)}f_n + O(h^3)$$

where $f_n^{(1)} := Df(x_n)$ and we have defined $x_n = x(t_n)$.

Meanwhile our general consistent explicit RK method reads as

$$\begin{aligned} k_1 &= f(x_n) \\ k_2 &= f(x_n + ahk_1) \\ x_{n+1} &= x_n + h\left[(1-b)k_1 + bk_2\right], \end{aligned}$$

or, all in one line:

$$\begin{aligned} x_{n+1} &= x_n + h\left[(1-b)f_n + bf(x_n + ahf_n)\right] \\ &= x_n + h\left[(1-b)f_n + b\left(f_n + ahf_n^{(1)}f_n + O(h^2)\right)\right] \\ &= x_n + h\left[f_n + abhf_n^{(1)}f_n\right] + O(h^3) \\ &= x_n + hf_n + abh^2 f_n^{(1)}f_n + O(h^3) \end{aligned}$$

Subtracting equations yields

$$\begin{aligned} \tau_n &= x(t_{n+1}) - x_{n+1} \\ &= h^2\left(\frac{1}{2} - ab\right)f_n^{(1)}f_n + O(h^3). \end{aligned}$$

Therefore $\tau_n = O(h^3)$ in general (i.e., we have second-order accuracy) if and only if $ab = 1/2$. Otherwise we have only first-order accuracy.

Thus the most general possible second-order accurate 2-stage explicit RK method (satisfying the sum rule) is specified, for $b \neq 0$, by the Butcher tableau:

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

The case $b = 1$ corresponds to the modified Euler method, but we can see there is an entire continuous family of choices.

14.3 Three-stage methods

Performing a Taylor series expansion to one higher order we find

$$x(t_{n+1}) = x_n + hf_n^{(0)} + \frac{h^2}{2} f_n^{(1)} f_n^{(0)} + \frac{h^3}{3!}\left[f_n^{(2)} f_n^{(0)} f_n^{(0)} + f_n^{(1)} f_n^{(1)} f_n^{(0)}\right] + O(h^4),$$

where we have now defined $f_n^{(k)} := D^k f(x_n)$, so in particular $f_n^{(0)} = f_n$. Note that in general $f_n^{(k)}$ is a $(k+1)$-index tensor, and when $d > 1$, we need to disambiguate how to 'multiply' or contract tensors in the above expression.

Concretely we have, for example,

$$[f_n^{(2)}]_{i,j_1 j_2} = \partial_{j_1}\partial_{j_2} f_i(x_n),$$

and in general we write

$$[f_n^{(k)}]_{i,j_1\cdots j_k} = \partial_{j_1}\cdots\partial_{j_k} f_i(x_n).$$

Note that (by commutation of partial derivatives) $f_n^{(k)}$ is symmetric with respect to permutation of the last k indices.

We define the contraction $f_n^{(2)} f_n^{(0)} f_n^{(0)} \in \mathbb{R}^d$ in particular via

$$[f_n^{(2)} f_n^{(0)} f_n^{(0)}]_i = \sum_{j_1,j_2} [f_n^{(2)}]_{i,j_1 j_2} [f_n^{(0)}]_{j_1} [f_n^{(0)}]_{j_2},$$

and $f_n^{(1)} f_n^{(1)} f_n^{(0)} \in \mathbb{R}^d$ via

$$\left[f_n^{(1)} f_n^{(1)} f_n^{(0)}\right]_i = \sum_{j_1,j_2} [f_n^{(1)}]_{i,j_1} [f_n^{(1)}]_{j_1,j_2} [f_n^{(0)}]_{j_2}.$$

The most general possible Butcher tableau for an explicit 3-stage RK method satisfying the sum rule is the following:

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ c & c & 0 & 0 \\ d & d-a & a & 0 \\ \hline & b_1 & b_2 & b_3 \end{array}$$

One can check (**exercise**) that order-3 consistency is equivalent to the following nonlinear equations being satisfied:

$$\begin{aligned} b_1 + b_2 + b_3 &= ? \quad \text{(order 1 condition)} \\ b_2 c + b_3 d &= ? \quad \text{(order 2 condition)} \\ b_2 c^2 + b_3 d^2 &= ? \quad \text{(order 3 condition)} \\ b_3 ac &= ? \quad \text{(order 3 condition)} \end{aligned}$$

The values of the ?'s are constants left to be determined by the reader.

Why do we get four equations? Let us consider order-by-order.

When we compute the LTE, we will get an expansion of the following form

$$\tau_n = C_1 f_n^{(0)} h + C_2 f_n^{(1)} h^2 + \left(C_3 f_n^{(2)} f_n^{(0)} f_n^{(0)} + C_4 f_n^{(1)} f_n^{(1)} f_n^{(0)} \right) h^3 + O(h^4),$$

where the constants C_k are defined in terms of our RK data. We need

In order to guarantee the cancellation of first-order terms, we need $C_1 = 0$. For second-order terms, we need $C_2 = 0$. But for third-order terms we need both $C_3 = 0$ and $C_4 = 0$. This yields four nonlinear equations. Notice that we have 6 free parameters, hence we expect that there are many solutions to these equations.

14.4 Beyond third order

It seems that with three stages, we have a bit of extra wiggle room and perhaps can ask for the cancellation of one additional order. Let us evaluate this possibility.

The fourth-order contribution will involve the following four terms:

$$f^{(3)} f^{(0)} f^{(0)} f^{(0)}, \; f^{(2)} f^{(1)} f^{(0)} f^{(0)}, \; f^{(1)} f^{(2)} f^{(0)} f^{(0)}, \; f^{(1)} f^{(1)} f^{(1)} f^{(0)},$$

where we now omit the n from the subscript for notational clarity. The convention for evaluating these tensor contractions generalizes our earlier convention, as we shall illustrate in the case of $f^{(1)} f^{(2)} f^{(0)} f^{(0)}$:

- Write out all the indexed expressions in order with no repeated indices:

$$\left[f^{(1)} \right]_{i_1, j_1} \left[f^{(2)} \right]_{i_2, j_2 j_3} \left[f^{(0)} \right]_{i_3} \left[f^{(0)} \right]_{i_4}$$

- Reading from left to right after the first tensor, sequentially replace each newly appearing index with the first index that remains unpaired among the preceding factors, unless no unrepeated indices remain:

$$i_2 \leftarrow j_1, \quad i_3 \leftarrow j_2, \quad i_4 \leftarrow j_3$$

  to obtain

$$\left[f^{(1)} \right]_{i_1, j_1} \left[f^{(2)} \right]_{j_1, j_2 j_3} \left[f_n^{(0)} \right]_{j_2} \left[f_n^{(0)} \right]_{j_3}$$

- The first index of the first tensor will be the index for the contracted tensor. Sum over the rest of the indices:

$$\left[f^{(1)} f^{(2)} f^{(0)} f^{(0)} \right]_i = \sum_{j_1, j_2, j_3} \left[f^{(1)} \right]_{i, j_1} \left[f^{(2)} \right]_{j_1, j_2 j_3} \left[f^{(0)} \right]_{j_2} \left[f^{(0)} \right]_{j_3}.$$

Notice importantly that

$$f^{(2)} f^{(1)} f^{(0)} f^{(0)} \neq f^{(1)} f^{(2)} f^{(0)} f^{(0)},$$

in general for $d > 1$,so the order conditions for general systems are more restrictive than they would be for scalar equations!

In general such expressions are in one-to-one correspondence with rooted trees of order 4, cf. Figure (14.4). Each node in the tree corresponds to one of the factors of $f^{(k)}$. The

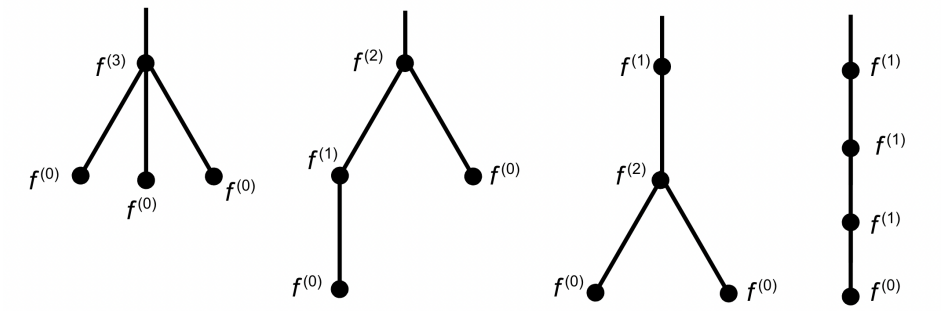

Figure 14.1: Rooted trees of order 4, corresponding, left to right, to the expressions $f^{(3)}f^{(0)}f^{(0)}f^{(0)}$, $f^{(2)}f^{(1)}f^{(0)}f^{(0)}$, $f^{(1)}f^{(2)}f^{(0)}f^{(0)}$, and $f^{(1)}f^{(1)}f^{(1)}f^{(0)}$.

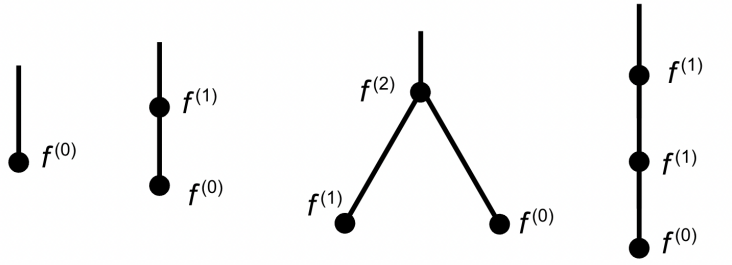

Figure 14.2: Rooted trees of order at most 3, corresponding, left to right, to the expressions $f^{(0)}$, $f^{(1)}f^{(0)}$, $f^{(2)}f^{(0)}f^{(0)}$, and $f^{(1)}f^{(1)}f^{(0)}$.

edge connecting each node to its parent corresponds to the first index of the tensor, while the edges connecting it to its children correspond to the remaining k indices. The root of the tree corresponds to the first factor, whose first index remains uncontracted in the final expression.

The contraction convention outlined above corresponds to a depth-first search through the tree. It can be seen (by symmetry of the tensors with respect to permutation of the child indices) that any depth-first search yields an equivalent expression, e.g., for the second tree in Figure (14.4), we have

$$f^{(2)}f^{(1)}f^{(0)}f^{(0)} = f^{(2)}f^{(0)}f^{(1)}f^{(0)},$$

corresponding to two different possible depth-first searches through the tree.

For completeness, we illustrate in Figure Figure (14.4) all rooted trees of order at most 3, which correspond to the lower-order terms considered earlier.

In summary, to get to fourth-order accuracy, we need to solve *four additional equations*, but we only have 6 degrees of freedom among 3-stage RK explicit methods satisfying the sum rule. Therefore we do not expect this to be possible, and indeed it is not.

14.5 Attainable order?

What order of accuracy can we expect from an s-stage explicit RK method satisfying the sum rule. In general, such a method is specified by a Butcher array of the following form,

k	1	2	3	4	5	6	7	8	9	10	11	12
T_k	1	1	2	4	9	20	48	115	286	719	1842	4766

Table 1: Number of rooted trees of order k as a function of k.

p	1	2	3	4	5	6	7	8	9	10	11	12
$\sum_{k=1}^{p} T_k$	1	2	4	8	17	37	85	200	486	1205	3047	7813
$s(p)$	1	2	3	4	6	9	13	20	31	49	78	125

Table 2: Number of rooted trees of order at most p as a function of p, i.e., the number of equations to solve to guarantee p-th order accuracy, as well as $s(p)$, the 'naive' number of stages one needs to guarantee at least as many unknowns as equations.

where b is unconstrained and A is unconstrained besides the requirement that it is *strictly lower triangular*:

$$\begin{array}{c|c} A\mathbf{1} & A \\ \hline & b^\top \end{array}$$

Note that A yields $s(s-1)/2$ degrees of freedom, and b yields another s degrees of freedom, so we have in total

$$\frac{s(s+1)}{2} \text{ degrees of freedom.}$$

How many equations do we need to solve to guarantee p-th order accuracy? Our preceding discussion suggests that we need to solve as many equations as the number of rooted trees of order at most p. The number of rooted trees T_k of order k grows quite quickly, cf. Table 1. In fact it is known that

$$\lim_{k\to\infty} \frac{T_{k+1}}{T_k} \approx 3,$$

and in particular the limit exists. Therefore, one expects asymptotic growth that looks roughly like $T_k \propto 3^k$.

In Table 2 we report $\sum_{k=1}^{p} T_k$ as a function of p, i.e., the number of equations that must be satisfied to guarantee p-th order accuracy. We also report $s = s(p)$, the 'naive' number of stages one needs to guarantee at least as many unknowns as equations. We say 'naive' to emphasize that some equations may be redundant and moreover that existence of a scheme is not guaranteed by having 'enough equations.' The precise number of stages needed is only known rigorously up to order 8.

In practice the most widely-used RK method, commonly called RK4, is a four-stage method with fourth-order accuracy, specified by the Butcher array

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

Historically, many popular RK methods were devised before the era of computers, and sparse Butcher arrays with nice-looking numbers were therefore favored. However, there is not any obvious *a priori* reason to favor the RK4 method among the many possible explicit fourth-order accurate four-stage RK methods. That said, it seems to have withstood the test of time rather well!

15 Absolute stability

The study of absolute stability for RK methods is similar to that of LMMs. As in the case of LMMs, we can reduce from the general linear setting to the scalar setting

$$x'(t) = \lambda x(t)$$

by diagonalization. We will now study the result of applying an *explicit* RK method to this scalar ODE. Along the way we will define some general notions pertaining to the absolute stability of RK methods that are relevant in the general (possibly implicit) case.

15.1 The stability function

Consider what happens when we apply an s-stage explicit RK method to the case $f(x,t) = \lambda x$. Equation (13.2) reduces simply to

$$k_i = \lambda x_n + \hat{h} \sum_{j=1}^{i-1} a_{ij} k_j, \ i = 1, \ldots, s,$$

where $\hat{h} := h\lambda$ as before.

To simplify expressions, we can define $\hat{k}_i = k_i/\lambda$ and moreover fix $x = x_n$, since n will remain fixed throughout the discussion. (Note that the case $\lambda = 0$ is trivial.) This yields the recursion

$$\hat{k}_i = x + \hat{h} \sum_{j=1}^{i-1} a_{ij} \hat{k}_j, \ i = 1, \ldots, s. \tag{15.1}$$

If we compute sequentially, we have

$$\begin{aligned}
\hat{k}_1 &= x \\
&= (1)x \\
\hat{k}_2 &= x + \hat{h} a_{21} \hat{k}_1 \\
&= \left(1 + a_{21}\hat{h}\right) x \\
\hat{k}_3 &= x + \hat{h}\left(a_{31}\hat{k}_1 + a_{32}\hat{k}_2\right) \\
&= \left(1 + a_{31}\hat{h} + a_{32}\hat{h}(1 + a_{21}\hat{h})\right) x \\
&= \left(1 + (a_{31} + a_{32})\hat{h} + a_{32}a_{21}\hat{h}^2\right) x \\
&\ \ \vdots
\end{aligned}$$

It is not hard then to see that in general we can write

$$\hat{k}_i = P_i(\hat{h})x, \tag{15.2}$$

where P_i is a polynomial of order $i - 1$.

Now (15.2), together with (13.1), implies that the image of x under the RK iteration

map is simply

$$\begin{aligned}\mathcal{R}(x) &= x + \hat{h}\sum_{i=1}^{s} b_i \hat{k}_i \\ &= \left(1 + \hat{h}\sum_{i=1}^{s} b_i P_i(\hat{h})\right) x.\end{aligned}$$

In summary,

$$\mathcal{R}(x) = R(\hat{h})x, \tag{15.3}$$

where

$$R(z) = 1 + z\sum_{i=1}^{s} b_i P_i(z). \tag{15.4}$$

Note that we have removed λ, h from the discussion as separate entities, appearing now only jointly via $\hat{h} = h\lambda$.

The function $R(z)$ is called the ***stability function*** of the RK method. *In general, the stability function is defined as the function that makes (15.3) hold in general for the scalar equation $x'(t) = \lambda x(t)$.* We will see later that for general (possibly implicit) RK methods, the stability function $R(z)$ is always a *rational function*. However, in the explicit case, we have just established that it is a polynomial and, moreover, that it is of *order s.* In the explicit case, we may then also refer to it as the ***stability polynomial*** of the RK method, for emphasis.

15.2 Deriving the stability polynomial

In fact we can compute the stability function in the explicit case fairly explicitly. Plugging (15.2) into (15.1), the polynomials satisfy the recursion

$$P_i(\hat{h})x = \hat{k}_i = \left[1 + \hat{h}\sum_{j=1}^{i-1} a_{ij} P_j(\hat{h})\right] x,$$

from which we deduce that

$$P_i(z) = 1 + z\sum_{j=1}^{i-1} a_{ij} P_j(z). \tag{15.5}$$

We have already computed

$$\begin{aligned}P_1(z) &= 1 \\ P_2(z) &= 1 + a_{21}z \\ P_3(z) &= 1 + (a_{31} + a_{32})z + a_{32}a_{21}z^2,\end{aligned}$$

from which we deduce next that

$$P_4(z) = 1 + (a_{41} + a_{42} + a_{43})z + (a_{42}a_{21} + a_{43}a_{31} + a_{43}a_{32})\, z^2 + (a_{43}a_{32}a_{21})z^3.$$

In general we can define a matrix c_{ij} of coefficients via

$$P_i(z) = \sum_{j=1}^{i-1} c_{ij} z^j,$$

where $c_{i0} = 1$, and (speculatively) c_{ij} consists for $j \geq 1$ of a sum of terms, one for each decreasing sequence of *positive* integers of length $j+1$, starting at i.

For example, when $i = 4$ we have 3 such sequences for $j = 1$:

$$(4,1),\ (4,2),\ (4,3);$$

3 such sequences for $j = 2$:

$$(4,2,1),\ (4,3,1),\ (4,3,2);$$

and 1 such sequence for $j = 3$:

$$(4,3,2,1).$$

These correspond to the terms

$$a_{41},\ a_{42},\ a_{43};$$

$$a_{42}a_{21},\ a_{43}a_{31},\ a_{43}a_{32};$$

$$a_{43}a_{32}a_{21};$$

respectively.

The sums of each row of terms can be written, respectively, as

$$[A\mathbf{1}]_4;$$

$$[A^2\mathbf{1}]_4;$$

$$[A^3\mathbf{1}]_4;$$

where $\mathbf{1} \in \mathbb{R}^s$ is the vector of all ones.

Indeed, observe that

$$[A^j\mathbf{1}]_i = \sum_{k_1,\ldots,k_j} a_{ik_1}a_{k_1k_2}\cdots a_{k_{j-1}k_j},$$

but since A is strictly lower triangular, the only remaining terms in the sum correspond to decreasing sequences $(i, k_1, \ldots, k_j)$. Therefore we reckon that $c_{ij} = [A^j\mathbf{1}]_i$, which in fact also recovers $c_{i0} = 1$ conveniently.

In summary, we conjecture

$$P_i(z) = \sum_{j=0}^{i-1} [A^j\mathbf{1}]_i z^j, \tag{15.6}$$

and we can check the claim by induction.

It is helpful first to note that in general, $[A^j\mathbf{1}]_i = 0$ if $j \geq i$. To see this, observe that $[A^j\mathbf{1}]_i$ consists of sum of terms in correspondence with decreasing sequences $(i, k_1, \ldots, k_j)$ of positive integers. There is no such sequence if $j \geq i$, so the claim follows.

Now to prove (15.6), suppose inductively that it hols for all $j < i$. We want to show that it holds for i. From (15.5), we have

$$\begin{aligned} P_i(z) &= 1 + z\sum_{j=1}^{i-1} a_{ij}P_j(z) \\ &= 1 + z\sum_{j=1}^{i-1} a_{ij}\sum_{k=0}^{j-1}[A^k\mathbf{1}]_j z^k \\ &= 1 + \sum_{j=1}^{i-1}\sum_{k=0}^{j-1} a_{ij}[A^k\mathbf{1}]_j z^{k+1}. \end{aligned}$$

Now $[A^k\mathbf{1}]_j = 0$ whenever $k \geq j$. Therefore we can extend the summation over k up to the upper limit $i-2$ without changing the expression, obtaining

$$P_i(z) = 1 + \sum_{j=1}^{i-1} \sum_{k=0}^{i-2} a_{ij} [A^k\mathbf{1}]_j z^{k+1}.$$

This allows us to exchange the order of summation and compute

$$\begin{aligned} P_i(z) &= 1 + \sum_{k=0}^{i-2} \left(\sum_{j=1}^{i-1} a_{ij} [A^k\mathbf{1}]_j \right) z^{k+1} \\ &= 1 + \sum_{k=0}^{i-2} \left[A(A^k\mathbf{1}) \right]_i z^{k+1} \\ &= 1 + \sum_{k=0}^{i-2} \left[A^{k+1}\mathbf{1} \right]_i z^{k+1} \\ &= 1 + \sum_{k=1}^{i-2} \left[A^k\mathbf{1} \right]_i z^k \\ &= \sum_{k=0}^{i-1} [A^k\mathbf{1}]_i z^k, \end{aligned}$$

as was to be shown.

In summary, we have shown.

Lemma 38. *With notation as in the preceding,*

$$P_i(z) = \sum_{j=0}^{i-1} [A^j\mathbf{1}]_i z^j.$$

Note in particular that $[A^j\mathbf{1}]_i = 0$ *whenever* $i \leq j$.

Now let's get back to the discussion of the stability polynomial. Plugging the result of the lemma into (15.4), we derive:

$$R(z) = 1 + z \sum_{i=1}^{s} b_i \sum_{j=0}^{i-1} [A^j\mathbf{1}]_i z^j.$$

Once again, the sum over j can be extended to the upper limit $s-1$ since $[A^j\mathbf{1}]_i = 0$ for $j \geq i$, yielding

$$\begin{aligned} R(z) &= 1 + z \sum_{i=1}^{s} b_i \sum_{j=0}^{s-1} [A^j\mathbf{1}]_i z^j \\ &= 1 + \sum_{j=0}^{s-1} \left(\sum_{i=1}^{s} b_i [A^j\mathbf{1}]_i \right) z^{j+1} \\ &= 1 + \sum_{j=0}^{s-1} \left(b^\top A^j\mathbf{1} \right) z^{j+1}, \end{aligned}$$

or equivalently,

$$R(z) = 1 + \sum_{j=1}^{s} (b^\top A^{j-1}\mathbf{1})z^j.$$

Evidently the linear term in $R(z)$, corresponding to the index $j = 1$ in the summation, is $(b \cdot \mathbf{1})z$. Hence the linear term is z if and only if the RK method is consistent.

Theorem 39. *For an explicit s-stage RK method with RK matrix A and weights b, the stability polynomial R is a polynomial of order s, given by the formula*

$$R(z) = 1 + \sum_{j=1}^{s} (b^\top A^{j-1}\mathbf{1})z^j.$$

The linear term in the polynomial is z if and only if $b^\top \mathbf{1} = 1$, i.e., if and only if the method is consistent.

15.3 Digression on the order of accuracy

Order-p consistency more generally places constraints on the stability function, but these constraints are only *necessary*, not *sufficient*, for order-p accuracy! Indeed, notice that order-p consistency requires that

$$\mathcal{R}(x) - e^{\lambda h}x = O(h^{p+1})$$

for arbitrary x, i.e., that

$$R(z) - e^z = O(z^{p+1})$$

as $z \to 0$. In the explicit case, we can match terms order-by-order against the Taylor series of e^z about the origin to obtain:

Corollary 40. *Consider an explicit s-stage RK method with RK matrix A and weights b. If the method is order-p accurate, then*

$$b^\top A^{j-1}\mathbf{1} = \frac{1}{j!}$$

for $j = 1, \ldots, p$.

Remark 41. Recall that for second-order accuracy of an explicit RK method satisfying the sum rule, there was *only one equation* to be satisfied, besides the order 1 condition that $b^\top \mathbf{1} = 1$. The preceding corollary furnishes one necessary equation for second-order accuracy, so in fact this equation is also sufficient. However, each successive higher order of accuracy demands more than one extra equation, while the corollary only furnishes one new equation per order.

In other words, an s-stage explicit RK method satisfying the sum rule $c = A\mathbf{1}$ is second-order accurate if and only

$$b^\top \mathbf{1} = 1, \quad b^\top A\mathbf{1} = \frac{1}{2}.$$

This isn't really a proof, but the claim can be verified directly (**exercise**). Alas we cannot make such a simple statement for higher orders.

For any s-stage, s-order accurate explicit RK method, the stability polynomial $R(z)$ is of order s and must match the Taylor series of e^z exactly up to order s, hence is completely determined as

$$R(z) = \sum_{j=1}^{s} \frac{1}{j!} z^j.$$

Recall that there are only four cases $s = 1, 2, 3, 4$ where the construction of such a method is possible, and though there may be many s-stage, s-order accurate explicit RK methods, they all have the same stability polynomial.

15.4 Absolute stability regions

We make a definition of absolute stability analogous to the definition in the setting of LMMs.

Definition 42. An RK method is ***absolutely stable*** (for given $\hat{h}$) if the solution $\{x_n\}_{n=0}^{\infty}$ produced by applying the method with step size h to the scalar ODE $x'(t) = f(x, t)$ satisfies $\lim_{n\to\infty} x_n = 0$ for all initializations. The ***region of absolute stability*** is the set of all $\hat{h}$ for which the method is absolutely stable. Other definitions are assumed to be similar without comment.

Since $x_{n+1} = \mathcal{R}(x_n) = R(\hat{h})x_n$, it is easy to see that the RK method furnishes the solution

$$x_n = (R(\hat{h}))^n x_0,$$

and therefore:

Theorem 43. *The region of absolute stability for an RK method is* $\{z \in \mathbb{C} : |R(z)| < 1\}$.

Remark 44. The weak region of absolute stability is simply the set $\{z \in \mathbb{C} : |R(z)| \leq 1\}$, and there are no complications due to repeated roots as in the case of multistep methods.

Observe that consistency implies that

$$R(z) = 1 + z + O(z^2).$$

Therefore, in a neighborhood of the origin, the absolute stability region should resemble the absolute stability region induced by $R(z) = 1 + z$, which is the stability function for the explicit Euler method. With some menial work, it is possible to conclude:

Theorem 45. *The region of absolute stability of any consistent RK method contains some interval* $(-h_0, 0)$*, where* $h_0 > 0$*, and in fact contains some open set in* $\mathbb{C}$ *containing this interval. Moreover, the region of absolute stability does not contain any interval of the form* $(0, h_1)$ *for* $h_1 > 0$.

Note that this conclusion is not true of LMMs, cf. the midpoint/leapfrog method.

For the s-stage, s-order accurate explicit RK methods, the absolute stability regions in all four cases are depicted in Figure (15.4).

Note that none of the methods are A-stable, and indeed since $R(z) \to z$ as $z \to -\infty$:

Theorem 46. *No explicit RK method is A-stable or even* A_0*-stable.*

How can we compute arbitrary absolute stability regions? We can again adopt the boundary locus perspective. Sweeping across $\theta \in [0, 2\pi)$, solve

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

for z. Since $R(z)$ is an order-s polynomial, this is now a root-finding problem with s solutions, yielding s values of z on the boundary locus associated to the each value of θ.

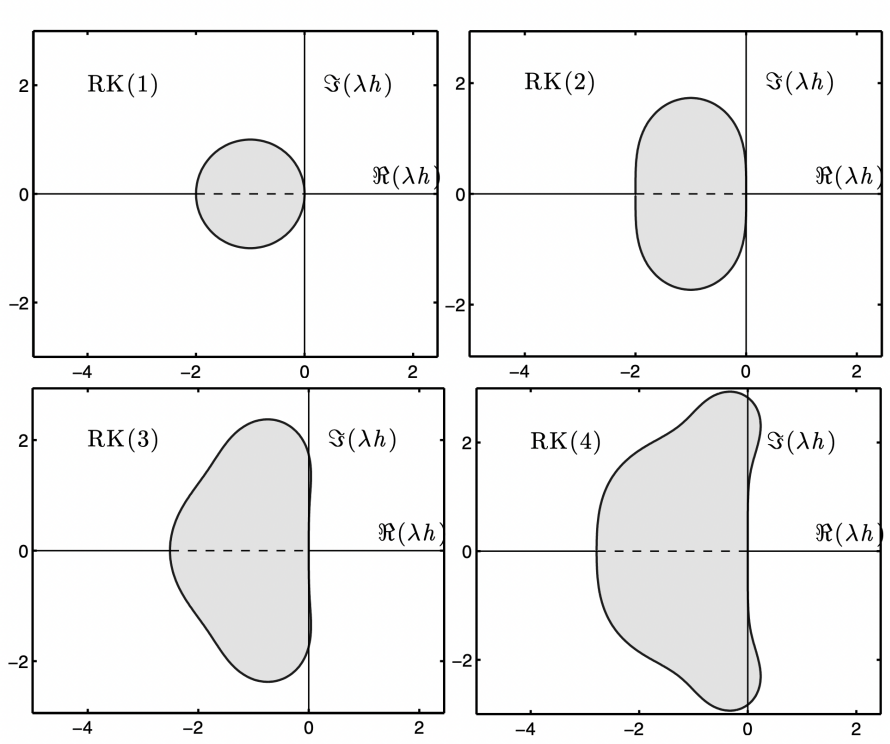

Figure 15.1: Absolute stability regions for the s-stage, s-order accurate explicit RK methods, $s = 1, 2, 3, 4$.

15.5 Implicit stability functions

In the general (not necessarily explicit) case, we still have

$$\hat{k}_i = x + \hat{h} \sum_{j=1}^{s} a_{ij} \hat{k}_j, \quad i = 1, \dots, s,$$

and letting $\hat{\mathbf{k}} = (\hat{k}_1, \dots, \hat{k}_s)^\top$, we can repackage these equations in vectorized form as

$$\hat{\mathbf{k}} = x\mathbf{1} + \hat{h} A \hat{\mathbf{k}},$$

which we can solve as

$$\hat{\mathbf{k}} = x(1 - \hat{h}A)^{-1}\mathbf{1}.$$

Then

$$\begin{aligned} \mathcal{R}(x) &= x + \hat{h} \sum_{i=1}^{s} b_i \hat{k}_i \\ &= x + \hat{h} b^\top \hat{\mathbf{k}} \\ &= \left(1 + \hat{h} b^\top (1 - \hat{h}A)^{-1}\mathbf{1}\right) x, \end{aligned}$$

from which it follows that the stability function can be written

$$R(z) = 1 + z b^\top (1 - zA)^{-1}\mathbf{1},$$

which is evidently a rational function.

A-stability can be assessed in terms of the stability function $R(z)$. Since it is a rational function, unlike the explicit case, we cannot rule out the possibility of A-stability. It is hard to predict in advance whether an RK method will be A-stable, but we will see examples that are (even to arbitrary order of accuracy). *A posteriori*, A-stability can be checked by locating the poles of $R(z)$ and bounding $R(z)$ on the imaginary axis alone.

Theorem 47. *An RK method is A-stable if and only if all of the poles of the stability function lie in the right half-plane and $|R(z)| \leq 1$ for all z on the imaginary axis.*

Proof. Use the maximum modulus principle for holomorphic functions. (If you don't know this, don't worry about it.) □

16 Runge-Kutta-Chebyshev methods

Our goal in this section is do design an s-stage explicit RK method whose stability polynomial yields an interval of (weak) absolute stability that is as long as possible. Our target is the polynomial

$$R(z) = T_s\left(1 + \frac{z}{s^2}\right),$$

where T_s is the s-th Chebyshev polynomial, defined by

$$T_s(t) = \cos(s \cos^{-1}(t)), \quad t \in [-1, 1].$$

The choice of the s-th polynomial is required since the order of the stability polynomial must be s, the number of stages. Refer to Appendix B for background on Chebyshev polynomials. Furthermore, note that by construction

$$T_s(1) = 1, \quad T_s'(1) = s^2,$$

where the second equation can be confirmed via elementary calculus and application of L'Hospital's rule.

The shifting and scaling in our definition of the target R are designed to ensure that

$$R(0) = 1, \ R'(0) = 1,$$

which is equivalent to consistency of the corresponding RK method, and moreover that

$$|R(t)| \leq 1, \quad t \in [-2s^2, 0],$$

so that the interval of (weak) absolute stability is $[-2s^2, 0]$. The remarkable property of ***Runge-Kutta-Chebyshev methods*** is that this interval grows *quadratically* (as opposed to linearly) with the number s of stages.

How can we define an RK method that yields this stability polynomial? We will do so somewhat obliquely. As with any computation involving orthogonal polynomials, the key is to exploit their three-term recurrence relation, which for Chebyshev polynomials is (B.1), reproduced here as

$$T_{j+1}(t) = 2tT_j(t) - T_{j-1}(t) \tag{16.1}$$

for $j = 1, 2, \ldots$. Recall moreover that $T_0 \equiv 1$ and $T_1(t) = t$.

Suppose that we are given a value $x = x_0$ and want to compute $\mathcal{R}(x)$, the image under the Runge-Kutta map for a time step h. We will consider only the autonomous case $f(x,t) = f(x)$, which we can pass to from the non-autonomous case as usual. (For simplicity we will avoid the time step index n in the presentation. Accordingly, the subscripts in the following ***do not*** indicate the time step index.) We will produce $\mathcal{R}(x)$ over the course of several stages.

First define

$$x_1 = x_0 + \frac{h}{s^2} f(x_0)$$

and then recursively define

$$x_{j+1} = 2x_j - x_{j-1} + \frac{2h}{s^2} f(x_j)$$

for $j = 1, \ldots, s-1$. Then return

$$\mathcal{R}(x) = x_s.$$

The reader should verify (**exercise**) that this algorithm in fact defines an explicit, consistent RK method. (How should the recursive construction be defined directly in the non-autonomous case so that, when the algorithm is interpreted as an RK method, the sum rule (13.3) is satisfied?)

To confirm that $R(z) = T_s\left(1 + \frac{z}{s^2}\right)$, we need only consider the scalar case $f(x) = \lambda x$ and verify that

$$\mathcal{R}(x) = T_s\left(1 + \frac{\hat{h}}{s^2}\right) x. \tag{16.2}$$

Evidently in this case we have

$$x_1 = \left(1 + \frac{\hat{h}}{s^2}\right) x = T_1\left(1 + \frac{\hat{h}}{s^2}\right) x.$$

We claim that in general

$$x_j = T_j\left(1 + \frac{\hat{h}}{s^2}\right) x$$

To see this, inductively assume that it holds for all $j < i$ and then compute

$$\begin{aligned} x_{i+1} &= 2x_i - x_{i-1} + \frac{2h}{s^2} f(x_i) \\ &= 2\left(1 + \frac{\hat{h}}{s^2}\right) x_i - x_{i-1} \\ &= 2\left(1 + \frac{\hat{h}}{s^2}\right) T_i\left(1 + \frac{\hat{h}}{s^2}\right) - T_{i-1}\left(1 + \frac{\hat{h}}{s^2}\right) \\ &= T_{i+1}\left(1 + \frac{\hat{h}}{s^2}\right), \end{aligned}$$

which follows by applying the 3-term recurrence (16.1) at $t = 1 + \frac{\hat{h}}{s^2}$. By induction, the claim is proved, from which (16.2) immediately follows.

17 Collocation methods

The idea of collocation methods is based once again on the integral form of our ODE:

$$x(t+h) = x(t) + \int_t^{t+h} f(x(\tau), \tau)\, d\tau. \tag{17.1}$$

We want to replace the integral in the right-hand side with a quadrature formula, possibly of high order in h.

17.1 Basic motivation

By shifting and scaling, we can reduce the problem of designing quadrature formulas to the interval $[0, 1]$. If we choose nodes $c_1, \ldots, c_s$ and weights $b_1, \ldots, b_s$ for this interval such that

$$\int_0^1 \xi(u)\, du \approx \sum_{i=1}^{s} b_i \xi(c_i), \tag{17.2}$$

then we have the approximation of the integral form (17.1):

$$x(t+h) \approx x(t) + h \sum_{i=1}^{s} b_i f\left(x(t + c_i h), t + c_i h\right).$$

Consider the nodes and weights as being now fixed in the discussion that immediately follows.

Consider the time $t = t_n$, define $x_n \approx x(t_n)$ as usual as our discrete solution, define intermediate times $t_{n,i} := t_n + c_i h$ following our earlier convention, loosely define

$$k_i \approx f(x(t_{n,i}), t_{n,i}),$$

and define a time-stepping scheme

$$x_{n+1} = x_n + h \sum_{i=1}^{s} b_i k_i, \tag{17.3}$$

inspired by (17.1), of the form of an RK method as in (13.1).

17.2 Defining equations for the slopes

How can the k_i be determined? The idea is to replace $x(t)$ by a polynomial $p(t)$ of order s which satisfies

$$p(t_n) = x_n, \quad p'(t_{n,i}) = f(p(t_{n,i}), t_{n,i}), \ i = 1, \ldots, s, \tag{17.4}$$

matching the value of $x(t)$ at time $t = t_n$ and ***satisfying the differential equation exactly at the intermediate times*** $t_{n,i}$.

Then we will more concretely define

$$k_i = f(p(t_{n,i}), t_{n,i}), \tag{17.5}$$

which in turn specifies the numerical scheme via (17.3).

Note that (17.4) demands a self-consistency condition on p, which will translate to a system of nonlinear equations defining the k_i, of the form (13.2).

To see this, note that instead of looking for p satisfying (17.4), we can look for $q(t) := p'(t)$ of order $s - 1$ satisfying

$$q(t_{n,j}) = k_j, \quad j = 1, \ldots, s$$

and then recover

$$p(t) = x_n + \int_{t_n}^{t} q(\tau)\, d\tau. \tag{17.6}$$

But q can be recovered as a linear combination of Lagrange basis polynomials subordinate to the interpolation points

$$\mathbf{t}_h = (t_{n,1}, \ldots, t_{n,s}) = t_n \mathbf{1} + hc,$$

where c is the (fixed) vector of nodes. Concretely,

$$q(t) = \sum_{j=1}^{s} \ell_j(t; \mathbf{t}_h) k_j.$$

(**Note:** *Here we follow a one-indexing rather than zero-indexing convention for the interpolation points, by contrast with the convention of Appendix A used in the discussion of interpolation in the context of LMMs.*)

Plugging into (17.6) and evaluting at $t = t_{n,i}$, we obtain

$$\begin{aligned} p(t_{n,i}) &= x_n + \sum_{j=1}^s k_j \int_{t_n}^{t_{n,i}} \ell_j(\tau; \mathbf{t}_h)\, d\tau. \\ &= x_n + h \sum_{j=1}^s k_j \int_0^{c_i} \ell_j(\tau; c)\, d\tau \\ &= x_n + h \sum_{j=1}^s a_{ij} k_j, \end{aligned}$$

where

$$a_{ij} = \int_0^{c_i} \ell_j(\tau; c)\, d\tau.$$

Then to satisfy (17.5), we must have

$$k_i = f\left(x_n + h \sum_{j=1}^s a_{ij} k_j,\, t_{n,i} \right),$$

which, together with (17.3), precisely defines an RK method of the general form (13.1)-(13.2).

17.3 Defining the weights, given the nodes

We will imagine for now that the nodes $0 \leq c_1 < \ldots < c_s \leq 1$ are fixed and return later to the question of defining the 'optimal' nodes. For now we ask ourselves: given the nodes, how to define appropriate quadrature weights $b_1, \ldots, b_s$, guaranteeing (17.2), i.e.,

$$\int_0^1 \xi(u)\, du \approx \sum_{i=1}^s b_i \xi(c_i)$$

in some appropriate sense?

One way to think about quadrature rules is to ask for them to be exact for polynomials up to some order p. This motivates replacing ξ with its interpolating polynomial for the data

$$(c_1, \xi(c_1)), \ldots, (c_s, \xi(c_s)),$$

which exactly matches ξ as long as ξ is a polynomial of order at most $s-1$, by the uniqueness property of the Lagrange interpolating polynomial. Concretley, we replace

$$\int_0^1 \xi(u)\, du \approx \int_0^1 \sum_{i=1}^s \ell_i(u; c) \xi(c_i)\, du = \sum_{i=1}^s b_i \xi(c_i),$$

where

$$b_i := \int_0^1 \ell_i(u; c)\, du.$$

We know that the corresponding quadrature rule is exact for polynomials of order up to $s-1$, and moreover by Corollary 61, the error more generally satisfies

$$\left| \int_0^1 \xi(u)\, du - \sum_{i=1}^s b_i \xi(c_i) \right| \leq C \sup_{u \in [0,1]} |\xi^{(p)}(u)|, \tag{17.7}$$

where $p = s$, and the constant $C > 0$ depends on the nodes c but is otherwise universal and independent of ξ.

In general, if a quadrature rule is exact for polynomials of order up to $p-1$, even if $p \neq s$, then (17.7) holds by replacing ξ in the integral $\int_0^1 \xi(u)\,du$ with a $(p-1)$-th order polynomial interpolation through p points, e.g., the Chebyshev nodes.

Such a bound in terms of the p-th derivative ensures that if we plug our quadrature rule into (17.1), appropriately shifting and scaling the integration interval from $[t_n, t_n + h]$ to $[0, 1]$, we introduce the discrepancy

$$\begin{aligned}\left|\int_t^{t+h} g(\tau)\,d\tau - h\sum_{i=1}^s b_i g(t+c_i h)\right| &= h\left|\int_0^1 g(t+uh)\,du - \sum_{i=1}^s b_i g(t+c_i h)\right| \\ &\leq Ch^{p+1}\sup_{\tau\in[0,T]}|g^{(p)}(\tau)|\end{aligned}$$

into the integral form of the ODE. This is strongly suggestive of an LTE of size $O(h^{p+1})$, hence an order of accuracy of $O(h^p)$. This is in fact the case, as we shall discuss below.

Thus far our discussion has held for an *arbitrary* choice of nodes $c_1, \ldots, c_s$, which we have found induce an order-s accurate implicit RK method. Somewhat remarkably, we shall see that for a *special choice of nodes* c, we can define a quadrature rule that is exact for polynomials up to order $2s-1$ and more generally satisfies (17.7) with $p = 2s$, yielding an order-$(2s)$ accurate implicit RK method!

17.4 Digression on solving implicit methods

Now that we are encountering our first implicit RK methods, it is worthwhile to think about when these schemes admit (unique) solutions. Moreover, given an RK method specified by matrix A, weights $b = (b_i)$, and nodes $c = (c_i)$, we may ask in practice how to solve for the slopes k_i defined by (13.2).

Let $x = x_n$ and $t = t_n$ be fixed. We can repackage the equations (13.2) in vectorized form as

$$K = F(X + hAK, \mathbf{t}_h), \tag{17.8}$$

where X and K are $s \times d$ matrices and $\mathbf{t}_h$ is an $s \times 1$ defined by

$$X = \begin{pmatrix} x^\top \\ \vdots \\ x^\top \end{pmatrix}, \quad K = \begin{pmatrix} k_1^\top \\ \vdots \\ k_s^\top \end{pmatrix}, \quad \mathbf{t}_h = \begin{pmatrix} t + c_1 h \\ \vdots \\ t + c_s h \end{pmatrix},$$

and $F : \mathbb{R}^{s\times d} \times \mathbb{R}^{s\times d} \to \mathbb{R}^{s\times d}$ is a vectorized extension of f. Concretely, for

$$Y = \begin{pmatrix} y_1^\top \\ \vdots \\ y_s^\top \end{pmatrix} \in \mathbb{R}^{s\times d}, \quad \boldsymbol{\tau} = \begin{pmatrix} \tau_1 \\ \vdots \\ \tau_s \end{pmatrix} \in \mathbb{R}^s,$$

we can define

$$F(Y, \boldsymbol{\tau}) = \begin{pmatrix} f(y_1, \tau_1) \\ \vdots \\ f(y_d, \tau_d) \end{pmatrix} \in \mathbb{R}^{s\times d}.$$

If we endow $\mathbb{R}^{s\times d}$ with the Frobenius norm, it is worthwhile to see that L-Lipschitzness of f (in the first slot, which is all the really matters) extends to F as follows:

$$\|F(Y,\boldsymbol{\tau}) - F(Z,\boldsymbol{\tau})\|_{\mathrm{F}} \leq L\|Y - Z\|_{\mathrm{F}}.$$

We can view the problem of solving (17.8) as a fixed-point problem for the map

$$\Phi : \mathbb{R}^{s\times d} \to \mathbb{R}^{s\times d},\ \Phi(K) = F(X + hAK, \mathbf{t}_h).$$

Evidently, if f is L-Lipschitz, then

$$\|\Phi(K) - \Phi(K')\|_{\mathrm{F}} \leq Lh\|AK - AK'\|_{\mathrm{F}} \leq (L\|A\|h)\,\|K - K'\|_{\mathrm{F}},$$

where $\|\cdot\|$ indicates the ordinary operator norm as usual. Therefore Φ is $(L\|A\|h)$- Lipschitz, hence the entire scheme admits a unique solution for all $h < (L\|A\|)^{-1}$, which can be found by simple fixed-point iteration and accelerated once again by Anderson acceleration. Alternatively, we can view $K = \Phi(K)$ as a generic system of nonlinear equations to be solved by Newton's method. Considerations such as those discussed in Section 8.1.2 are enough to pass more broadly to the locally Lipschitz case, provided that the underlying system admits a solution.

17.5 Summary up to determining the nodes

We summarize our discussion so far in the following theorem, which also formalizes the order of accuracy and the situation with regard to the sum rule (13.3).

Theorem 48. *Fix arbitrary nodes $0 \leq c_1 < \ldots < c_s \leq 1$, and define*

$$a_{ij} = \int_0^{c_i} \ell_j(\tau; c)\, d\tau, \quad b_i := \int_0^1 \ell_i(u; c)\, du.$$

Then the s-stage RK method defined by matrix $A = (a_{ij})$, weights $b = (b_i)$, and nodes $c = (c_i)$ satisfies the sum rule $c = A\mathbf{1}$ and is order-p accurate, where p is the order of accuracy of the quadrature rule induced by b, c, in the sense that the quadrature rule is exact for polynomials up to order $p-1$ (or equivalently that (17.7) holds). In particular, $p \geq s$ regardless of the choice of nodes.

Remark 49. **(Subtle!)** We will only sketch the proof of the order of accuracy, which is fairly technical, but the work that we leave out will be similar to work we've done in the proof of Dahlquist's theorem, Theorem 21. (If you only want order-s accuracy—which is not sharp in the most important case of Gauss-Legendre quadrature where $p > s$, to be discussed below—then the proof is significantly easier.) The key ingredient for the more powerful proof is that the discrete solution of the RK method can be viewed as being produced from a polynomial solution p on $[t_n, t_{n+1}]$ that satisfies the differential equation ***exactly*** at the nodes, cf. (17.4). The reason why it is easier to get order-s accuracy is that this polynomial is actually ***not*** generally $O(h^{p+1})$-accurate pointwise! It is only accurate up to $O(h^{s+1})$. But magically its value at t_{n+1} is accurate up to $O(h^{p+1})$, even when $p > s$. This motivates the following lemma, which identifies the RK method's discrete solution $x_{n+1} = \mathcal{R}(x_n)$ with the value $p(t_{n+1})$ of this polynomial.

To get merely order-s accuracy, it is sufficient to first show that the RK slopes k_i satisfy $k_i = f(x(t_{n,i}), t_{n,i}) + O(h^s)$, which can be verified by plugging in the true slopes $k_i^{(0)} = f(x(t_{n,i}), t_{n,i})$ as the initial guess for the RK slopes in a fixed-point iteration, then verifying that the entire fixed-point iteration does not move them by more than $O(h^s)$, since it

only moves them by $O(h^s)$ in the first iteration, since they nearly already satisfy the RK equations. Then they can be swapped out for the true slopes in the quadrature formula defining x_{n+1} (17.3) at the cost of only $O(h^{s+1})$. But according to the preceding claims, this approach throws away a lot of important information.

Lemma 50. *With notation as in the preceding, the value $p(t_{n+1})$ of the polynomial p defined via (17.4) coincides with the solution $x_{n+1} = x_n + h\sum_{i=1}^s b_i k_i$ of the RK method at time step $n+1$, defined as in (17.3).*

Proof. Recall $p(t_n) = x_n$ and $p'(t_{n,i}) = k_i$ by construction. Therefore

$$x_{n+1} = p(t_n) + h\sum_{i=1}^s b_i p'(t_{n,i}).$$

Now p' is a polynomial of order $s-1$, so the quadrature formula defined by b, c is exact for it, and

$$h\sum_{i=1}^s b_i p'(t_{n,i}) = \int_{t_n}^{t_{n+1}} p'(\tau)\,d\tau = p(t_{n+1}) - p(t_n),$$

from which it follows that $x_{n+1} = p(t_{n+1})$, as was to be shown. □

Proof of Theorem 48. First we see to the sum rule. Compute

$$\sum_{j=1}^s a_{ij} = \int_0^{c_i} \sum_{j=1}^s \ell_j(\tau; c)\,d\tau.$$

Now $\sum_{j=1}^s \ell_j(\tau; c)$ is literally the Lagrange interpolating polynomial for the data $(c_1, 1), \ldots, (c_s, 1)$. Since the constant function 1 is a zero-th order polynomial, the Lagrange interpolating polynomial recovers it exactly by uniqueness, i.e., $\sum_{j=1}^s \ell_j(\,\cdot\,; c) \equiv 1$. This implies

$$\sum_{j=1}^s a_{ij} = \int_0^{c_i} 1\,d\tau = c_i,$$

as desired. □

17.5.1 Proof of order of accuracy: autonomous linear case

For the order of accuracy, consider first the linear case

$$f(x) = Ax.$$

Let $x(t)$ denote the true solution, and consider for simplicity the time step $n = 0$. The LTE is the size of the difference between $x(h)$ and $\mathcal{R}(x_0)$, where $\mathcal{R}$ is the RK iteration map for step size h. Let $y(t)$ denote the polynomial solution furnished by the collocation method on $[0, h]$, so by construction $y'(t) = f(y(t))$ at intermediate times $t = t_{0,1}, \ldots, t_{0,s}$. It is convenient to define the 'defect'

$$\delta(t) := y'(t) - f(y(t)),$$

so by construction $\delta(t_{0,i}) = 0$ for $i = 1, \ldots, s$, and

$$y'(t) = f(y(t)) + \delta(t)$$

for all $t \in [0, h]$. Moreover, define an error displacement

$$\epsilon(t) := y(t) - x(t)$$

as a function of continuous time. We want to show that $\epsilon(h) = O(h^{p+1})$.

Then by subtracting ODEs we obtain an ODE

$$\epsilon'(t) = A\epsilon(t) + \delta(t)$$

This ODE can be solved *exactly* by ***Duhamel's principle***:

$$\epsilon(t) = e^{At}\epsilon(0) + \int_0^t e^{A(t-\tau)}\delta(\tau)\, d\tau,$$

as can be verified directly by differentiation, and remembering that the initial condition for the displacement is $\epsilon(0) = 0$, we have in fact that

$$\epsilon(h) = \int_0^h \underbrace{\Phi(h, \tau)\, \delta(\tau)}_{=:\zeta(\tau)}\, d\tau, \tag{17.9}$$

where $\Phi(t, \tau) := e^{A(t-\tau)}$, and $\zeta(\tau)$ is defined as indicated

We can replace this integral with the quadrature formula at the cost of only $O(h^{p+1})$ error, i.e.,

$$\epsilon(h) = h \sum_{i=1}^{s} b_i \zeta(c_i h) + O(h^{p+1}).$$

But by construction

$$\zeta(c_i h) = \Phi(h, c_i h)\delta(t_{0,i}) = 0,$$

so in fact

$$\epsilon(h) = O(h^{p+1}),$$

as was to be shown.

17.5.2 Proof of order of accuracy: non-autonomous linear case

Now consider more generally the case

$$f(x, t) = A(t)x(t) + b(t).$$

Here we obtain, *mutatis mutandi*, the same equation 17.9, where now $\Phi(\tau_1, \tau_0)$ denotes the matrix that sends an initial condition $u(\tau_0)$ at time τ_0 to the solution of $u'(t) = A(t)u(t)$ at time $t = \tau_1$.

In fact $\Phi(t, \tau)$ can be defined as the solution of a matrix-valued differential equation

$$\begin{cases} \frac{d}{dt}\Phi(t, \tau) = A(t) \\ \Phi(\tau, \tau) = I, \end{cases}$$

in which we view τ as fixed, defining the initial time. One can then check directly that indeed $u(t) := \Phi(t, \tau)u(\tau)$ satisfies $u'(t) = A(t)u(t)$.

Moreover, for a general inhomogeneous linear system

$$v'(t) = A(t)v(t) + g(t),$$

there is still a ***general Duhamel principle***:

$$v(t) = \Phi(t,0)v(0) + \int_0^t \Phi(t,\tau)g(\tau)\,d\tau,$$

which can likewise be verified directly by differentiation.

17.5.3 Sketch in general case

In the case of general f, first let us reduce for simplicity to the autonomous case

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

which we can do guilt-free via the sum rule. We again have

$$y'(t) = f(y(t)) + \delta(t)$$

As we did for the proof of Dahlquist's theorem (Theorem 21), we will try to reduce to the **non-autonomous** linear case (even when the underlying equation is autonomous).

Indeed, subtracting equations we obtain

$$\epsilon'(t) = A(t)\epsilon(t) + g(t) + \delta(t),$$

where $A(t) := Df(x(t))$, $\delta(t_{0,i}) = 0$, and $\|g\| = O(\|\epsilon\|^2)$, with $\|\cdot\|$ here denoting the L^∞ norm on the interval $[0,h]$.

The lesson of the proof of Theorem 21 was that, even though g is not known a priori, we can essentially wave it away for present purposes with fixed-point iteration. In other words, we solve a sequence of equations

$$\epsilon_k'(t) = A(t)\epsilon_k(t) + g_k(t) + \delta(t),$$

where we initialize $g_0 = g$ and define g_{k+1} appropriately in terms of the previous solution ϵ_k. Throughout this enterprise, we maintain $\max_k \|g_k\| = O(h^{2(p+1)})$, which never bothers us for the purpose of proving $\max_k \|\epsilon_k\| = O(h^{p+1})$.

17.6 Gauss-Legendre methods

The orthogonal polynomials (see Appendix C) with respect to the weight function $w \equiv 1$ on the interval $[-1,1]$ are called the ***Legendre polynomials***. Let $-1 < t_1 < \cdots < t_s < 1$ be the zeros of the s-th Legendre polynomial, which are called the ***Gauss-Legendre quadrature*** nodes. Gauss-Legendre methods are derived by choosing nodes $c_i = (t_i+1)/2$, i.e., shifting and scaling the Gauss-Legendre nodes to the interval $[0,1]$.

From Theorem 66, Gauss-Legendre quadrature is exact for polynomials of order up to $2s-1$. Likewise the quadrature rule on $[0,1]$ induced by the nodes $c_1,\ldots,c_s$ is exact for polynomials up to order $2s-1$, and we have:

Theorem 51. *Let $c_i = (t_i+1)/2$ for $i = 1,\ldots,s$, where $t_i \in (-1,1)$ is the i-th zero of the Legendre polynomial of order s. Define*

$$a_{ij} = \int_0^{c_i} \ell_j(\tau; c)\,d\tau, \quad b_i := \int_0^1 \ell_i(u; c)\,du.$$

Then the s-stage RK method defined by matrix $A = (a_{ij})$, weights $b = (b_i)$, and nodes $c = (c_i)$ satisfies the sum rule $c = A\mathbf{1}$ and is order-$(2s)$ accurate.

Remark 52. These methods are called the ***Gauss-Legendre methods***. Remarkably, the Gauss-Legendre methods are A-stable for all $s = 1, 2, \ldots$. The stability function is in fact the (s, s)-Padè approximant of e^z. The Wanner-Hairer Norsett theorem in turn guarantees that this Padè approximant satisfies the condition for A-stability. However, we will prove the A-stability of the Gauss-Legendre methods by different means in the next part, where we shall see that they satisfy a stronger condition called ***algebraic stability***.

My methods are really methods of working and thinking; this is why they have crept in everywhere anonymously.

E. Noether

Part IV
Geometric numerical integration

Besides some order-p accuracy and absolute stability, what more could we possibly ask of a numerical method for ODEs? A lot more, it turns out. Many differential equations of interest exhibit certain monotonicity or conservation properties, and the study of numerical integrators that preserve such structure exactly is called ***geometric numerical integration***. We will consider case studies for certain properties of major interest.

18 Monotone systems

We have seen that linear systems of the form

$$x'(t) = -Ax(t),$$

where A is symmetric (or more generally Hermitian) positive definite are ***dissipative*** in the loose sense that small perturbations are exponentially damped over time.

We can think of such systems as falling into a more general category of ***gradient flows***

$$x'(t) = -\nabla U(x(t)),$$

where $U : \mathbb{R}^d \to \mathbb{R}$ is a convex function. Indeed, note that the linear case is recovered by taking

$$U(x) = \frac{1}{2} x^\top A x,$$

which is a convex function with gradient $\nabla U(x) = Ax$. Such a system is also dissipative, and the globally attractive fixed point is the minimizer of U, which is unique as long as U is strictly convex.

The choice $f(x,t) = \nabla U(x)$ can be generalized considerably to a category of functions of possibly non-gradient type called ***monotone mappings***. We motivate their definition in the following way.

Suppose that U is convex, and let $x, y \in \mathbb{R}^d$. Then by the definition of convexity

$$\begin{aligned} U(y) &\geq U(x) + \langle \nabla U(x), y - x \rangle, \\ U(x) &\geq U(y) + \langle \nabla U(y), x - y \rangle. \end{aligned}$$

Here and in the sequel $\langle \cdot, \cdot \rangle$ denotes an inner product on $\mathbb{R}^d$, though the space itself can of course be generalized. We will also use $\| \cdot \|$ to denote the norm induced by this inner product via $\|x\|^2 = \langle x, x \rangle$.

By adding both inequalities we can cancel all the non-gradient terms, yielding

$$\langle \nabla U(x) - \nabla U(y), x - y \rangle \geq 0.$$

Accordingly, we make the following definition (which accommodates dynamics in both real and complex[2] inner product spaces):

Definition 53. If V is an inner product space, we say that $T : V \to V$ is ***monotone*** if

$$\operatorname{Re}\langle T(x) - T(y), x - y\rangle \geq 0$$

for all $x, y \in V$. We say that the system

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

is ***monotone*** if for every fixed t, the map $-f(\,\cdot\,, t)$ is monotone. Concretely, this means that

$$\operatorname{Re}\langle f(x,t) - f(y,t), x - y\rangle \leq 0$$

for all $x, y \in \mathbb{F}^d$ and $t \in [0, T]$, where either $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$.

The big deal about monotone equations is that their flow is contractive, in the following sense:

Theorem 54. *Suppose that $x'(t) = f(x(t), t)$ is a monotone system, and consider two solutions $u(t)$ and $v(t)$ induced by initial conditions u_0 and v_0. The function*

$$t \mapsto \|u(t) - v(t)\|^2$$

is monotone non-increasing.

Proof. Simply compute

$$\begin{aligned}
\frac{d}{dt}\|u(t) - v(t)\|^2 &= \frac{d}{dt}\langle u(t) - v(t), u(t) - v(t)\rangle \\
&= \langle u'(t) - v'(t), u(t) - v(t)\rangle + \langle u(t) - v(t), u'(t) - v'(t)\rangle \\
&= 2\operatorname{Re}\langle u(t) - v(t), u'(t) - v'(t)\rangle \\
&= 2\operatorname{Re}\langle u(t) - v(t), f(u(t), t) - f(v(t), t)\rangle \\
&\leq 0,
\end{aligned}$$

which implies the desired result. □

What are some interesting monotone systems, besides gradient flows? Consider, for example,

$$T(x) = \nabla U(x) + Bx, \tag{18.1}$$

where $U : \mathbb{R}^d \to \mathbb{R}$ is strictly convex and B is real ***skew***-symmetric (**read:** normal with imaginary eigenvalues) and the inner product is the ordinary dot product. Then

$$\langle Bx - By, x - y\rangle = (x - y)^\top B(x - y) = 0,$$

which in turn implies that T is monotone. Note however that such T is not of gradient type!

Here the gradient term $\nabla U(x)$ is strictly dissipative, while the Bx term alone induces length-preserving (***isometric***) dynamics. Indeed, note that the solution of $x'(t) = Bx$, is $x(t) = e^{Bt}x_0$, and e^{Bt} is an orthogonal matrix as long as B is skew-symmetric.

For the rest of this section, we concern ourselves with the design of numerical integrators that, for monotone systems, preserve the structure of Theorem 54.

[2]For complex inner product spaces, we adopt the convention that the inner product is linear in the second slot and conjugate-linear in the first slot.

18.1 Algebraic stability

Motivated by Theorem 54, we make the following definition:

Definition 55. We say that an RK method is ***algebraically stable*** if

$$\|\mathcal{R}(u) - \mathcal{R}(v)\| \leq \|u - v\|$$

holds for any $u, v \in \mathbb{F}^d$, where $\mathcal{R}$ is the RK iteration map for a *monotone system*.

In fact algebraic stability is stronger than A-stability:

Theorem 56. *An algebraically stable RK method is A-stable.*

Proof. Suppose $\lambda \in \mathbb{C}$ with $\mathrm{Re}(\lambda) \leq 0$. Then $x'(t) = \lambda x(t)$ is a monotone system (*check this!*), following the same reasoning as in our justification that (18.1) is monotone, since it is of 'convex gradient plus isometric' type.

Therefore Definition 55, applied to the case $u = x \neq 0$, $v = 0$, yields

$$\|\mathcal{R}(x)\| \leq \|x\|.$$

But $\mathcal{R}(x) = R(\hat{\lambda})x$, so we have that $|R(\lambda h)|\|x\| \leq \|x\|$. Since this holds for all $x \neq 0$ and all λ with $\mathrm{Re}(\lambda) \leq 0$, it follows that $|R(z)| \leq 1$ on the left half-plane. The maximum modulus principle (cf. Theorem 47) in turn implies strict inequality on the strict left half-plane, and A-stability follows. □

An important object called the M matrix emerges from the study of algebraic stability:

Definition 57. For an RK method with RK matrix A and weights b, define the M matrix as

$$M = \mathrm{diag}(b)\, A + A\, \mathrm{diag}(b) - bb^\top.$$

We will always assume that A and b are purely real, so notice that M is real symmetric by construction. Remarkably, if it is positive semidefinite, then the corresponding RK method is algebraically stable:

Theorem 58. *If the M matrix for an RK method is positive definite and the weights $b_1, \ldots, b_s \geq 0$, then the corresponding RK method is algebraically stable.*

Proof. The proof is, well...completely algebraic. But it requires a lot of notation.

Consider $u, v \in \mathbb{F}^n$ and fix a current time t. We will compare $\mathcal{R}(u)$ and $\mathcal{R}(v)$ for a step size h. Let $k_1, \ldots, k_s$ denote the slopes for the first of these RK steps and $r_1, \ldots, r_s$ the slopes for the second. Define 'intermediate values,'

$$u_i = u + h \sum_{i=1}^{s} a_{ij} k_i, \quad v_i = v + h \sum_{i=1}^{s} a_{ij} r_j,$$

so

$$k_i = f(u_i, t_i), \quad r_i = f(v_i, t_i),$$

where the intermediate times are defined $t_i = t + c_i h$, $i = 1, \ldots, s$. (Note that these are not the 'usual' discrete times t_n, as we have already implicitly fixed a time step n in the context of this discussion.)

Finally,

$$\mathcal{R}(u) = u + h \sum_{i=1}^{s} b_i k_i, \quad \mathcal{R}(v) = v + h \sum_{i=1}^{s} b_i r_i,$$

and with the notation set up we can finally compute:

$$\begin{aligned}\|\mathcal{R}(u)-\mathcal{R}(v)\|^2 &= \left\|(u-v)+h\sum_{i=1}^{s} b_i(k_i-r_i)\right\|^2 \\ &= \left\langle (u-v)+h\sum_{i=1}^{s} b_i(k_i-r_i), (u-v)+h\sum_{i=1}^{s} b_i(k_i-r_i)\right\rangle \\ &= \|u-v\|^2 + 2h\,\mathrm{Re}\left\langle u-v, \sum_{i=1}^{s} b_i(k_i-r_i)\right\rangle + h^2\left\|\sum_{i=1}^{s} b_i(k_i-r_i)\right\|^2 .\end{aligned}$$

Therefore, defining $z_i := k_i - r_i$, to show algebraic stability it is equivalent to show

$$2\mathrm{Re}\left\langle u-v, \sum_{i=1}^{s} b_i z_i\right\rangle + h\left\|\sum_{i=1}^{s} b_i z_i\right\|^2 \leq 0.$$

Now we substitute ... □

19 Quadratic invariants

20 Hamiltonian systems

If I have had any success in mathematical physics, it is, I think, because I have been able to dodge mathematical difficulties.

J. W. Gibbs

Part V
Stochastic differential equations and Monte Carlo sampling

Appendices

Appendix A Lagrange interpolation

Let $t_0 < \ldots < t_m \in \mathbb{R}$ (not necessarily evenly spaced), and let $x_0, \ldots, x_m \in \mathbb{R}^d$. We will discuss general vector-valued interpolation ($d \geq 1$), but this is completely equivalent to scalar-valued interpolation performed componentwise. Our goal is to find a polynomial $P(t)$ such that $P(t_i) = x_i$ for $i = 0, \ldots, m$. We need to consider a polynomial of order at least m in order to do so in general, and in fact there exists a unique such interpolating polynomial of order m, sometimes called the ***Lagrange (interpolating) polynomial***.

Usually we have in mind a scenario in which there exists a 'ground truth' function g for which $g(t_i) = x_i$ for all i, and we hope that the polynomial ℓ interpolating the data (t_i, x_i), $i = 0, \ldots, m$, is a good approximation for g.

In general, Lagrange interpolation is an ***extremely dangerous*** thing to do in the sense that the ground truth g and the interpolation P can differ dramatically even as the size m of the dataset is increased, due to Runge's phenomenon.[3]

However, if g is sufficiently smooth, the number of interpolation points m is *fixed*, and the range $[t_0, t_m]$ shrinks, then in fact ℓ approximates g with pointwise $O(|t_m - t_0|^{m+1})$ error, regardless of the choice of interpolation points t_i. First we discuss how to construct P, and then we present the proof of this error bound.

A.1 Construction and uniqueness

Consider the ***Lagrange basis polynomials*** subordinate to the choice of interpolation points $\mathbf{t} = (t_0, \ldots, t_m)$

$$\ell_j(t; \mathbf{t}) := \prod_{i \in \{0,\ldots,m\} \setminus \{j\}} \frac{t - t_i}{t_j - t_i}, \ j = 0, \ldots, m. \tag{A.1}$$

We will omit the dependence on $\mathbf{t}$ henceforth since the interpolation points will be fixed. Note that ℓ_j is a polynomial of order m, and moreover observe that

$$\ell_j(t_i) = \delta_{ij}, \tag{A.2}$$

where δ_{ij} is the Kronecker delta.

Therefore if we simply define the Lagrange interpolating polynomial $\ell : \mathbb{R} \to \mathbb{R}^d$ via

$$\ell = \sum_{j=0}^{m} \ell_j x_j,$$

then by construction $\ell(t_i) = \sum_{j=0}^{m} x_j \delta_{ij} = x_i$, achieving the desired interpolating property.

Note that the delta property (A.2) implies that the ℓ_j are linearly independent as functions, hence form a basis for the space of polynomials of order m. (Indeed, suppose $\sum_{j=0}^{m} c_j \ell_j \equiv 0$ is the zero polynomial. Then in particular, plugging in t_i to both sides, we have $c_i = 0$ for all $i = 0, \ldots, m$. This guarantees linear independence.)

In fact, the basis property implies that ℓ is the *unique* order-m polynomial interpolant, i.e., the unique polynomial P of order m achieving $P(t_i) = x_i$ for all $i = 0, \ldots, m$. Indeed,

[3] Such pathologies can be avoided if the t_i are chosen to be the Chebyshev nodes for the interval $[t_0, t_m]$, but in general, e.g., for equispaced grids, beware.

any such P can be written as $P = \sum_{j=0}^{m} \ell_j y_j$ for some $y_j \in \mathbb{R}^d$ by the basis property of the ℓ_j (applied componentwise). Then plugging t_i into both sides we see that $P(t_i) = y_i$, but $P(t_i) = x_i$ by the interpolating property, so in fact $P = \sum_{j=0}^{m} \ell_j x_j = \ell$. This establishes uniqueness as claimed.

Theorem 59. *Given distinct interpolation points $t_0, \dots, t_m \in \mathbb{R}$, the Lagrange basis polynomials $\ell_j(\,\cdot\,; \mathbf{t})$ defined as in (A.1) form a basis for the space of polynomials of order m. Moreover, given $x_0, \dots, x_m \in \mathbb{R}^d$, the Lagrange interpolating polynomial*

$$\ell(t) := \sum_{j=0}^{m} \ell_j(t; \mathbf{t}) x_j$$

is the unique polynomial of order m achieving $\ell(t_i) = x_i$ for $i = 0, \dots, m$.

A.2 Error bound

Next we establish the approximation property that is of use to us.

Theorem 60. *Suppose $g \in C^{m+1}([a, b])$, and let $t_0, \dots, t_m \in [a, b]$ be distinct interpolation points. Let $x_i = g(t_i)$ for $i = 0, \dots, m$, and let ℓ denote the Lagrange interpolating polynomial for the data (t_i, x_i), $i = 0, \dots, m$, as in the statement of Theorem 59. Then for every $t \in [a, b]$ there exists $\xi = \xi(t) \in [a, b]$ such that*

$$g(t) - \ell(t) = \frac{g^{(m+1)}(\xi)}{(m+1)!} \prod_{i=0}^{m} (t - t_i).$$

Proof. Define the remainder

$$R := g - \ell.$$

We know that R has zeros at $t_0, \dots, t_m$, and we want to 'compare' it with the degree-$(m+1)$ polynomial P with the same zeros, namely

$$P(t) := \prod_{i=0}^{m} (t - t_i).$$

Now R and P both have zeros at the t_i, hence any linear combination of them does as well. But for any fixed $t \in [a, b] \backslash \{t_0, \dots, t_m\}$, we can find a linear combination of R and P that has an *additional* zero at t, namely the function

$$h_t(s) := R(s) - \frac{R(t)}{P(t)} P(s).$$

Then h_s has (at least) $m+2$ zeros on $[a, b]$, hence by Rolle's theorem, h_s' has $m+1$ zeros on this interval, and by repeated application of Rolle's theorem, we see that $h_s^{(m+1)}$ has a zero on $[a, b]$, which we call $\xi = \xi(t)$. Then by construction

$$0 = h_t^{(m+1)}(\xi) = g_t^{(m+1)}(\xi) - \frac{R(t)}{P(t)} (m+1)!,$$

where the last equality follows from the fact that $P^{(m+1)} \equiv (m+1)!$, since P is a monic polynomial of degree $m+1$, and $\ell^{(m+1)} \equiv 0$, since ℓ is a polynomial of degree m. Solving for the remainder we have

$$R(t) = \frac{g^{(m+1)}(\xi)}{(m+1)!} P(t),$$

as was to be shown. (Note that the theorem holds trivially for $t \in \{t_0, \dots, t_m\}$.) □

The preceding theorem immediately implies the desired $O(|t_m - t_0|^{m+1})$ approximation property of the Lagrange interpolating polynomial. We turn to proving a sharp formulation of this bound.

Corollary 61. *Suppose $g \in C^{m+1}([a,b])$, and let $a = t_0 < \cdots < t_m = b$ be distinct interpolation points. Let $x_i = g(t_i)$ for $i = 0, \ldots, m$, and let ℓ denote the Lagrange interpolating polynomial for the data (t_i, x_i), $i = 0, \ldots, m$, as in the statement of Theorem 59. Further define*

$$\Delta := \max_{i=0,\ldots,m-1} |t_{i+1} - t_i|$$

to be the maximal distance between adjacent interpolation points, and let $|||\cdot|||$ denote the uniform norm $|||h||| := \sup_{t\in[a,b]} |h(t)|$ on $[a,b]$. Then

$$|||g - \ell||| \leq \frac{|||g^{(m+1)}|||}{4(m+1)} \Delta^{m+1}.$$

Proof. Let $t \in [a,b]$. By the preceding theorem, we know that there exists $\xi \in [a,b]$ such that

$$g(t) - \ell(t) = \frac{g^{(m+1)}(\xi)}{(m+1)!} \prod_{i=0}^{m} (t - t_i).$$

Then we only need to show that

$$\prod_{i=0}^{m} |t - t_i| \leq \frac{m!}{4} \Delta^{m+1}.$$

Note that for any $t \notin [a,b]\backslash\{t_0, \ldots, t_m\}$, there exists j such that $t \in [t_j, t_{j+1}]$. These values t_j, t_{j+1} are the closest of the t_i to the given t. The next closest value must be within a distance of 2Δ, then the next next closest within a distance of 3Δ, etc., with the furthest being within a distance of $m\Delta$. Moreover, the product of $(t - t_j)(t_{j+1} - t_j)$ is maximized at the midpoint $(t_j + t_{j+1})/2$, hence bounded by $\Delta^2/4$. Therefore the product of all the distances is bounded as

$$\prod_{i=0}^{m} |t - t_i| \leq (\Delta^2/4)(2\Delta)(3\Delta)\cdots(m\Delta) = \frac{m!}{4}\Delta^{m+1},$$

as was to be shown. □

Appendix B Chebyshev polynomials

The ***Chebyshev polynomials***[4] constitute an extremely useful gadget in numerical analysis, due to their ***equioscillation property*** on the interval $[-1,1]$, which can be transferred to arbitrary intervals via apprropriate shifting and scaling.

The n-th Chebyshev polynomial is defined for $t \in [-1,1]$ by

$$T_n(t) = \cos(n \cos^{-1}(t)).$$

It is not immediately obvious from the definition that T_n is actually a polynomial! However, this fact is guaranteed by the three-term recurrence

$$T_{n+1}(t) = 2tT_n(t) - T_{n-1}(t), \tag{B.1}$$

[4] To us, the terminology 'Chebyshev polynomials' will always indicate 'Chebyshev polynomials of the first kind' unless otherwise specified.

together with the obvious identities $T_0 \equiv 1$ and $T_1(t) = t$.

To see (B.1), we use the classic trigonometric identity

$$\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b).$$

Substituting $-b$ in for b and adding identities yields another identity

$$\cos(a+b) + \cos(a-b) = 2\cos(a)\cos(b),$$

or

$$\cos(a+b) = 2\cos(a)\cos(b) - \cos(a-b).$$

We can then compute

$$\begin{aligned} T_{n+1}(t) &= \cos(n\cos^{-1}(t) + \cos^{-1}(t)) \\ &= 2\cos(n\cos^{-1}(t))\cos(\cos^{-1}(t)) - \cos((n-1)\cos^{-1}(t)) \\ &= 2tT_n(t) - T_{n-1}(t), \end{aligned}$$

verifying (B.1).

Note that the zeros of T_n (called the ***Chebyshev nodes***) are therefore

$$\cos\left(\frac{2j-1}{2n}\pi\right), \quad j = 1, \ldots, n,$$

and moreover

$$|T_n(t)| \leq 1, \quad t \in [-1, 1].$$

Moreover, the local extrema of T_n, attained at

$$\cos\left(\frac{j\pi}{n}\right), \ j = 1, \ldots, n-1$$

are all ± 1 (as are the values at the interval endpoints -1 and 1). This is the equioscillation property.

Then we have the following corollary of Theorem 59, which clarifies the advantage of using Chebyshev nodes (as opposed to equispaced points) as the interpolation points in Lagrange interpolation.

Theorem 62. *Suppose $g \in C^{m+1}([-1,1])$. Let $t_0, \ldots, t_m$ be the zeros of the Chebyshev polynomial of degree $m+1$, i.e.,*

$$t_i = \cos\left(\frac{2i+1}{2m+2}\pi\right), \quad i = 0, \ldots, m.$$

Let $x_i = g(t_i)$. Then the Lagrange interpolating polynomial ℓ as in Theorem 59 satisfies

$$|||g - \ell||| \leq 2^{-m}\frac{|||g^{(m+1)}|||}{(m+1)!},$$

where $|||\cdot|||$ denotes he uniform norm on the interval $[-1,1]$.

Proof. Note that the polynomial $P(t) = \prod_{i=0}^{m}(t - t_i)$ must coincide with a scalar multiple of T_{m+1} by the fundamental theorem of algebra. The three-term recurrence (B.1) implies that the degree-n part of T_n is $2^{n-1}t^n$ for all $n \geq 1$. Therefore

$$\prod_{i=0}^{m}(t - t_i) = \frac{1}{2^m}T_{m+1}(t).$$

But $|T_{m+1}| \leq 1$ on $[-1,1]$, so the result follows from Theorem 60. □

The next corollary follows simply by shifting and scaling the interval of interpolation.

Theorem 63. *Suppose $g \in C^{m+1}(I)$, where $I := [c-r, c+r]$ with $c \in \mathbb{R}$ and $r > 0$. Define interpolation points*

$$t_i = c + r\cos\left(\frac{2i+1}{2m+2}\pi\right), \quad i = 0, \ldots, m,$$

and let $x_i = g(t_i)$. Then the Lagrange interpolating polynomial ℓ as in Theorem 59 satisfies

$$|||g - \ell||| \leq \frac{r^{m+1}}{2^m}\frac{|||g^{(m+1)}|||}{(m+1)!},$$

where $|||\cdot|||$ denotes he uniform norm on the interval I.

Appendix C Orthogonal polynomials and Gauss quadrature

Given an interval $[a, b]$, possibly with $a = -\infty$, $b = +\infty$ and a weight function $w : [a, b] \to [0, \infty)$, ***Gauss quadrature*** concerns the estimation of integrals of the form

$$\int_a^b g(t)\, w(t)\, dt$$

with quadrature rules of the form

$$\sum_{i=1}^{m} b_i g(t_i).$$

We will ask for an integration rule that is exact when g is a polynomial, up to whatever order we can achieve. Note that if the interval is infinite or semi-infinite, we must demand that the weight function $w(t)$ has sufficient decay. For example, the choice $w(t) = e^{-t^2/2}$ for the interval $(-\infty, \infty)$ will induce the so-called Gauss-Hermite quadrature.

In this course, Gauss quadrature is relevant via the study of collocation methods (cf. Section 17), where the Gauss-Legendre quadrature induced by the choice $w \equiv 1$ on the interval $[-1, 1]$ will be relevant, after suitable post-processing by shifting and scaling of the interval.

C.1 Orthogonal polynomials

Gauss quadrature is intimately linked to the theory of ***orthogonal polynomials***. For a given interval and weight function, orthogonal polynomials are defined to be a sequence of polynomials p_k of ascending degree $k = 0, 1, 2, \ldots$ (usually fixed uniquely by the constraint that they are monic) that are orthogonal with respect to the weighted inner product

$$\langle g, h\rangle = \int_a^b g(t)h(t)w(t)\, dt.$$

In our discussion we will fix our orthogonal polynomial sequence to be the unique monic orthogonal polynomial sequence.

In principle $p_k = p_k(t)$ can be constructed by applying the Gram-Schmidt procedure to the monomial sequence of polynomials $1, t, t^2, t^3, \ldots$, i.e., recursively constructing them via

$$p_k(z) = t^k - \sum_{l=0}^{k-1} \frac{\langle t^k, p_l\rangle}{\langle p_l, p_l\rangle} p_l(t).$$

By construction it should be clear that the p_k are monic orthogonal polynomials. Moreover note that as monic polynomials of ascending order, $p_0, \ldots, p_k$ form a basis for the set $\mathbb{P}_k$ polynomials of degree k, for any k.

C.2 Three-term recurrence

One important feature of orthogonal polynomial sequences, which we don't really need, is the three-term recurrence, which always permits stable evaluation of orthogonal polynomials. The derivation of the three-term-term recurrence comes from noticing that $tp_k(t)$ is a polynomial of order $k+1$, hence can be expanded in the orthogonal basis $p_0, \ldots, p_{k+1}$. Hence we can write

$$tp_k(t) = \sum_{l=0}^{k+1} \frac{\langle tp_k, p_l \rangle}{\langle p_l, p_l \rangle} p_l(t).$$

However, interestingly, many of these terms vanish, as we can compute:

$$\langle tp_k, p_l \rangle = \int_a^b p_k(t) \underbrace{[tp_l(t)]}_{\ni \mathbb{P}_{l+1}} w(t)\, dt.$$

When $l < k-1$, p_k is orthogonal to a basis for $\mathbb{P}_{l+1}$, hence orthogonal to all of $\mathbb{P}_{l+1}$, and these terms go away, yielding

$$\frac{\langle tp_k, p_{k+1} \rangle}{\langle p_{k+1}, p_{k+1} \rangle} p_{k+1}(t) = \left(t - \frac{\langle tp_k, p_k \rangle}{\langle p_k, p_k \rangle} \right) p_k(t) - \frac{\langle tp_k, p_{k-1} \rangle}{\langle p_{k-1}, p_{k-1} \rangle} p_{k-1}(t),$$

after suitable rearrangement of terms. Note that $tp_k - p_{k+1} \in \mathbb{P}_k$, hence using orthogonality

$$\langle tp_k, p_{k+1} \rangle = \langle p_{k+1}, p_{k+1} \rangle + \langle tp_k - p_{k+1}, p_{k+1} \rangle = \langle p_{k+1}, p_{k+1} \rangle,$$

and similarly $\langle tp_k, p_{k-1} \rangle = \langle tp_{k-1}, p_k \rangle = \langle p_k, p_k \rangle$, and in turn we have

$$p_{k+1}(t) = \left(t - \frac{\langle tp_k, p_k \rangle}{\langle p_k, p_k \rangle} \right) p_k(t) - \frac{\langle p_k, p_k \rangle}{\langle p_{k-1}, p_{k-1} \rangle} p_{k-1}(t),$$

or

$$p_{k+1}(t) = (t - \alpha_k) p_k(t) - \beta_k p_{k-1}(t),$$

where

$$\alpha_k := \frac{\langle tp_k, p_k \rangle}{\langle p_k, p_k \rangle}, \quad \beta_k := \frac{\langle p_k, p_k \rangle}{\langle p_{k-1}, p_{k-1} \rangle} > 0.$$

This is the classic three-term recurrence, and in fact any three-term recurrence specified by a sequence of $\alpha_k \in \mathbb{R}$ and $\beta_k \geq 0$ corresponds to a weight function, and in some sense measures on $\mathbb{R}$ are equivalent to such sequences. (This is the content of Favard's theorem.) In Section 17, we see the utility of the three-term recurrence in the context of Chebyshev polynomials.

C.3 Zeros of orthogonal polynomials

In the last subsection, we essentially proved a useful lemma, which we'll state here now formally:

Lemma 64. *For any $l < k$, p_k is orthogonal to every polynomial in $\mathbb{P}_l$.*

In fact the zeros of orthogonal polynomials will be the Gauss quadrature nodes. The following lemma characterizes some of their key properties:

Lemma 65. *All m zeros of p_m are simple, hence distinct, and lie in (a, b).*

Proof. We only need to consider $m \geq 1$. In this case

$$\langle p_m, 1 \rangle = \int_a^b p_m(t) w(t)\, dt = 0,$$

so we know that p_m changes sign at least once in (a, b).

Let $\tau_1, \ldots, \tau_k$ denote the distinct points in (a, b) where p_m changes sign. Suppose for contradiction that $k < m$, and define

$$q(t) := \prod_{j=1}^{k} (t - \tau_j) \in \mathbb{P}_k$$

for some coefficients $a_0, \ldots, a_k$.

Since p_m and q both change sign at τ_j, it follows that $p_m q$ *never* changes sign on (a, b), from which it follows that

$$\langle p_m, q \rangle = \int_a^b p_m(t) q(t) w(t)\, dt \neq 0.$$

But $p_m \perp \mathbb{P}_k$ so this is a contradiction.

We conclude that p_m has at least m zeros in (a, b). Since p_m has degree m, by root counting, they must all be simple, and p_m cannot admit any zeros outside of (a, b). □

C.4 Gauss quadrature

In Section 17.3, given nodes $c_1, \ldots, c_s$ we explain how to choose weights $b_1, \ldots, b_s$ such that the quadrature rule

$$\int_0^1 \xi(u)\, du \approx \sum_{i=1}^{s} b_i \xi(c_i)$$

is exact for polynomials $\xi \in \mathbb{P}_{s-1}$. The weights were derived by exactly integrating the Lagrange interpolant of the data $(c_i, \xi(c_i))$, $i = 1, \ldots, s$, yielding the exact formula

$$b_i := \int_0^1 \ell_i(u; c)\, du$$

for $i = 1, \ldots, m$.

Given $\mathbf{t} = (t_1, \ldots, t_m)$, the exact same procedure yields more generally a quadrature rule

$$\int_a^b g(t) w(t)\, dt \approx \sum_{i=1}^{m} b_i g(t_i)$$

that is exact for polynomials $g \in \mathbb{P}_{m-1}$, with weights defined by

$$b_i := \int_a^b \ell_i(t; \mathbf{t}) w(t)\, dt$$

for $i = 1, \ldots, m$. (**Note:** *Here we follow a one-indexing rather than zero-indexing convention for the interpolation points, by contrast with the convention of Appendix A used in the discussion of interpolation in the context of LMMs.*)

The remarkable thing is that if $t_1, \ldots, t_m \in (a, b)$ are the zeros of the m-th orthogonal polynomial p_m with respect to the weight function w on the interval $[a, b]$, then in fact this quadrature rule is ***exact for all polynomials*** $g \in \mathbb{P}_{2m-1}$.

Theorem 66. *Let $a < t_1 < \ldots < t_m < b$ be the zeros of p_m, the m-th orthogonal polynomial with respect to the weight function w on the interval $[a, b]$, and let $\mathbf{t} = (t_1, \ldots, t_m)$. Define weights*

$$b_i := \int_a^b \ell_i(t; \mathbf{t}) w(t)\, dt$$

for $i = 1, \ldots, m$. Then

$$\int_a^b g(t)w(t)\, dt = \sum_{i=1}^m b_i g(t_i)$$

for all $g \in \mathbb{P}_{2m-1}$.

Proof. Let $g \in \mathbb{P}_{2m-1}$. Since the degree of p_m is m, by polynomial long division we can write

$$g = p_m q + r,$$

where $q \in \mathbb{P}_{m-1}$, and $\deg(r) < \deg(q)$, hence also $r \in \mathbb{P}_{m-1}$. Then

$$\sum_{i=1}^m b_i g(t_i) = \sum_{i=1}^m b_i p_m(t_i) q(t_i) + \sum_{i=1}^m b_i r(t_i).$$

But $p_m(t_i) = 0$ for all $i = 1, \ldots, m$, so in fact

$$\sum_{i=1}^m b_i g(t_i) = \sum_{i=1}^m b_i r(t_i).$$

But since r is a polynomial of degree at most $m-1$, the quadrature rule defined by weights b_i and quadrature points t_i is exact, and in fact we have shown

$$\sum_{i=1}^m b_i g(t_i) = \int_a^b r(t) w(t)\, dt. \tag{C.1}$$

Meanwhile, we can compute

$$\begin{aligned}
\int_a^b g(t)w(t)\, dt &= \int_a^b p_m(t) q(t) w(t)\, dt + \int_a^b r(t) w(t)\, dt \\
&= \langle p_m, q \rangle + \sum_{i=1}^m b_i g(t_i) \\
&= \sum_{i=1}^m b_i g(t_i),
\end{aligned}$$

where we have used (C.1) and the fact that $p_m \perp \mathbb{P}_{m-1}$. □