Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 1

1 Polynomial interpolation

1.1 Lagrange and Newton polynomials

Let $f : [a,b] \to \mathbb{R}$ be a real-valued continuous function defined on some interval $[a,b]$ and let $(x_i)_{i=0}^n$ be $n+1$ distinct points in $[a,b]$. We wish to construct a polynomial p of degree n which interpolates f at these points, i.e., satisfies

$$p(x_i) = f(x_i), \quad i = \overline{0,n}, \qquad p \in \mathcal{P}_n.$$

Theorem 1.1 (Existence and uniqueness) *Given $f \in C[a,b]$ and $n+1$ distinct points $(x_i)_{i=0}^n \in [a,b]$, there is exactly one polynomial $p \in \mathcal{P}_n$ such that $p(x_i) = f(x_i)$ for all i.*

Proof. 1) There is *at least one* polynomial interpolant $p \in \mathcal{P}_n$, the one in the *Lagrange form*,

$$p(x) = \sum_{i=0}^{n} f(x_i)\,\ell_i(x) \qquad \text{with} \quad \ell_i(x) := \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j}, \quad i = \overline{0,n}, \tag{1.1}$$

the ℓ_is are called the *fundamental* Lagrange polynomials. Each ℓ_i is the product of n linear factors, hence $\ell_i \in \mathcal{P}_n$. It also equals 1 at x_i and vanishes at $x_j \neq x_i$, i.e., $\ell_i(x_j) = \delta_{ij}$. Therefore $p \in \mathcal{P}_n$ and

$$p(x_j) = \sum_{i=0}^{n} f(x_i)\,\ell_i(x_j) = f(x_j)\,.$$

2) There is *at most one* polynomial interpolant $p \in \mathcal{P}_n$ to f on $(x_i)_{i=0}^n$. For if there are two, $p, q \in \mathcal{P}_n$, then the polynomial $r := p - q$ is of degree n and vanishes at $n+1$ points, whence $r \equiv 0$. ∎

Remark 1.2 Let us introduce the so-called *nodal* polynomial

$$\omega(x) := \prod_{i=0}^{n} (x - x_i).$$

Then, in the expression (1.1) for ℓ_i, the numerator is simply $\omega(x)/(x - x_i)$ while the denominator is equal to $\omega'(x_i)$. With that we arrive to a compact Lagrange form

$$\boxed{p(x) = \sum_{i=0}^{n} f(x_i)\,\ell_i(x) = \sum_{i=0}^{n} \frac{f(x_i)}{\omega'(x_i)}\,\frac{\omega(x)}{x - x_i}} \tag{1.2}$$

The Lagrange forms (1.1)-(1.2) for the interpolating polynomials are easy to manipulate, but they are unsuitable for numerical evaluation. An alternative is the *Newton form* which has an *adaptive* nature.

Method 1.3 (The Newton form) For $k = 0, 1, \ldots, n$, let $p_k \in \mathcal{P}_k$ be the polynomial interpolant to f on $x_0, \ldots, x_k$. Then two subsequent p_{k-1} and p_k interpolate the same values $f(x_i)$ for $i \leq k-1$, hence their difference is a polynomial of degree k that vanishes at k points $x_0, \ldots, x_{k-1}$. Thus

$$p_k(x) - p_{k-1}(x) = A_k \prod_{i=0}^{k-1} (x - x_i), \tag{1.3}$$

with some constant A_k which is seen to be equal to the *leading* coefficient of p_k. It follows that $p := p_n$ can be built step by step as one constructs the sequence $(p_0, p_1, ...)$, with p_k obtained from p_{k-1} by addition the term from the right-hand side of (1.3), so that finally

$$p(x) := p_n(x) = p_0(x) + \sum_{k=1}^{n} [p_k(x) - p_{k-1}(x)] = \sum_{k=0}^{n} A_k \prod_{i=0}^{k-1} (x - x_i),$$

Definition 1.4 (Divided difference) Given $f \in C[a,b]$ and $k+1$ distinct points $(x_i)_{i=0}^k \in [a,b]$, the *divided difference* $f[x_0, ..., x_k]$ *of order* k is, by definition, the leading coefficient of the polynomial $p_k \in \mathcal{P}_k$ which interpolates f at these points. By definition, it is a symmetric function of the variables $[x_0, ..., x_k]$, and if $f(x) = x^m$, $m \leq k$, then $f[x_0, ..., x_k] = \delta_{km}$

With this definition we arrive at the Newton formula for the interpolating polynomial.

Theorem 1.5 (Newton formula) *Given $n+1$ distinct points $(x_i)_{i=0}^n$, let $p_n \in \mathcal{P}_n$ be the polynomial that interpolates f at these points. Then it may be written in the Newton form*

$$\begin{aligned} p_n(x) \;=\; & f[x_0] + f[x_0, x_1]\,(x - x_0) + f[x_0, x_1, x_2]\,(x - x_0)(x - x_1) \;+\; \cdots \\ & \cdots + f[x_0, x_1, \ldots, x_n]\,(x - x_0)(x - x_1) \cdots (x - x_{n-1})\,, \end{aligned}$$

or, more compactly,

$$\boxed{p_n(x) = \sum_{k=0}^{n} f[x_0, ..., x_k] \prod_{i=0}^{k-1} (x - x_i)} \tag{1.4}$$

To make this formula of any use, we need an expression for $f[x_0, ..., x_k]$. One such can be derived from the Lagrange formula (1.2) by identifying the leading coefficient of p. This turns to be

$$f[x_0, ..., x_n] = \sum_{i=0}^{n} \frac{f(x_i)}{\omega'(x_i)}, \qquad \omega(x) := \prod_{i=0}^{n} (x - x_i).$$

However, this expression has computational disadvantages as the Lagrange form itself. A useful way to calculate divided difference is again an adaptive (or recurrence) approach.

Theorem 1.6 (Recurrence relation) *For distinct $x_0, x_1, \ldots, x_k$ (with $k \geq 1$), we have*

$$\boxed{f[x_0, ..., x_k] = \frac{f[x_1, ..., x_k] - f[x_0, ..., x_{k-1}]}{x_k - x_0}} \tag{1.5}$$

Proof. Let $q_0, q_1 \in \mathcal{P}_{k-1}$ be the polynomials such that q_0 interpolates f on $(x_0, x_1, \ldots, x_{k-1})$ and q_1 interpolates f on $(x_1, \ldots, x_{k-1}, x_k)$

and consider the polynomial

$$p(x) := \frac{x - x_0}{x_k - x_0}\, q_1(x) + \frac{x_k - x}{x_k - x_0}\, q_0(x)\,, \qquad p \in \mathcal{P}_k\,.$$

One readily sees that $p(x_i) = f(x_i)$ for all i, hence, p is the k-th degree interpolating polynomial for f. Moreover, the leading coefficient of p is equal to the difference of those of q_1 and q_0 divided by $x_k - x_0$, and that is exactly what the recurrence (1.5) says. ∎

Method 1.7 The recursive formula (1.5) allows for fast evaluation of the *divided difference table*

x_i	$f[*] = f(*)$	$f[*,*]$	$f[*,*,*]$	$f[*,*,*,*]$
x_0	$\to f[x_0]$	$\searrow$		
		$f[x_0, x_1]$	$\searrow$	
x_1	$\to f[x_1]$	$\searrow$	$f[x_0, x_1, x_2]$	$\searrow$
		$f[x_1, x_2]$	$\searrow$	$f[x_0, x_1, x_2, x_3]$
x_2	$\to f[x_2]$	$\searrow$	$f[x_1, x_2, x_3]$	$\nearrow$
		$f[x_2, x_3]$	$\nearrow$	
x_3	$\to f[x_3]$	$\nearrow$		

This can be done in $\mathcal{O}(n^2)$ operations and the outcome is the numbers $\{f[x_0,\ldots,x_k]\}_{k=0}^n$ at the head of the columns which can be used in the Newton form (1.4).

Method 1.8 (Horner scheme) Finally, evaluation of p at a given point x using Newton formula (provided that divided differences $A_k := f[x_0,\ldots x_k]$ are known) requires just n multiplications, as long as we do it by the *Horner scheme*

$$p_n(x) = \{\cdots\{\{A_n\times(x-x_{n-1})+A_{n-1}\}\times(x-x_{n-2})+A_{n-2}\}\times(x-x_{n-3})+\cdots+A_1\}\times(x-x_0)+A_0.$$

1.2 Examples

Example 1.9 Given the data

x_i	0	1	2	3
$f(x_i)$	-3	-3	-1	9

find the interpolating polynomial $p\in\mathcal{P}_3$ in both Lagrange and Newton forms.

1a) Fundamental Lagrange polynomials

$$\begin{aligned}
\ell_0(x) &= \frac{(x-1)(x-2)(x-3)}{-6} &&= -\tfrac{1}{6}(x^3-6x^2+11x-6),\\
\ell_1(x) &= \frac{x(x-2)(x-3)}{2} &&= \tfrac{1}{2}(x^3-5x^2+6x),\\
\ell_2(x) &= \frac{x(x-1)(x-3)}{-2} &&= -\tfrac{1}{2}(x^3-4x^2+3x),\\
\ell_3(x) &= \frac{x(x-1)(x-2)}{6} &&= \tfrac{1}{6}(x^3-3x^2+2x).
\end{aligned}$$

1b) Lagrange form:

$$\begin{aligned}
p(x) &= (-3)\cdot\ell_0(x)+(-3)\cdot\ell_1(x)+(-1)\cdot\ell_2(x)+9\cdot\ell_3(x)\\
&= \left(\tfrac{1}{2}-\tfrac{3}{2}+\tfrac{1}{2}+\tfrac{3}{2}\right)x^3+\left(-3+\tfrac{15}{2}-2-\tfrac{9}{2}\right)x^2+\left(\tfrac{11}{2}-9+\tfrac{3}{2}+3\right)x-3\\
&= x^3-2x^2+x-3.
\end{aligned}$$

2a) Divided differences:

$$\begin{array}{c|c|c|c|c}
\widehat{0} & \overset{*}{-3} & & & \\
 & & \frac{(-3)-(-3)}{1-0}=\overset{*}{0} & & \\
\widehat{1} & -3 & & \frac{2-0}{2-0}=\overset{*}{1} & \\
 & & \frac{(-1)-(-3)}{2-1}=2 & & \frac{4-1}{3-0}=\overset{*}{1}\\
\widehat{2} & -1 & & \frac{10-2}{3-1}=4 & \\
 & & \frac{9-(-1)}{3-2}=10 & & \\
\widehat{3} & 9 & & &
\end{array}$$

2b) Newton form:

$$p(x) = -\overset{*}{3}+\overset{*}{0}\cdot(x-\widehat{0})+\overset{*}{1}\cdot(x-\widehat{0})(x-\widehat{1})+\overset{*}{1}\cdot(x-\widehat{0})(x-\widehat{1})(x-\widehat{2}).$$

2c) Horner scheme

$$p(x) = \left\{\left[\overset{*}{1}\cdot(x-\widehat{2})+\overset{*}{1}\right]\cdot(x-\widehat{1})+\overset{*}{0}\right\}\cdot(x-\widehat{0})-\overset{*}{3}.$$

1.3 Some further formulas

Theorem 1.10 *Let $p_n \in \mathcal{P}_n$ interpolate $f \in C[a,b]$ at $n+1$ distinct points $x_0, ..., x_n$. Then for any $x \notin (x_i)$*

$$\boxed{f(x) - p_n(x) = f[x_0, ..., x_n, x]\,\omega(x)} \tag{1.6}$$

Proof. Given $x_0, .., x_n$, let $\bar{x} := x_{n+1}$ be any other point. Then, by (9.3), the corresponding polynomials p_n and p_{n+1} are related by

$$p_{n+1}(x) = p_n(x) + f[x_0, ..., x_n, \bar{x}]\,\omega(x), \qquad \omega(x) := \prod_{i=0}^{n}(x - x_i).$$

In particular, putting $x = \bar{x}$, and noticing that $p_{n+1}(\bar{x}) = f(\bar{x})$, we obtain

$$f(\bar{x}) = p_n(\bar{x}) + f[x_0, ..., x_n, \bar{x}]\,\omega(\bar{x}),$$

the latter equality being the same as (1.6). ∎

This theorem shows the error to be "like the next term" in the Newton form. However, we cannot evaluate the right-hand side of (1.6) without knowing the number $f(x)$. But as we now show we can relate it to the $(n+1)$-st derivative of f. For this we need a version of the Rolle's theorem:

Lemma 1.11 *If $g \in C^k[a,b]$ is zero at $k + \ell$ distinct points, then $g^{(k)}$ has at least ℓ distinct zeros in $[a,b]$.*

Proof. By Rolle's theorem, if $\phi \in C^1$ is zero at two point, then ϕ' is zero at an intermediate point. So, we deduce that g' vanishes at least at $(k-1)+\ell$ distinct points. Next, applying Rolle to g', we conclude that g'' vanishes at $(k-2)+\ell$ points, and so on. ∎

Theorem 1.12 *Let $[\bar{a}, \bar{b}]$ be the smallest interval that contains $x_0, ..., x_k$ and let $f \in C^k[\bar{a}, \bar{b}]$. Then there exists $\xi \in [\bar{a}, \bar{b}]$ such that*

$$\boxed{f[x_0, ..., x_k] = \frac{1}{k!} f^{(k)}(\xi)} \tag{1.7}$$

Proof. Let $p \in \mathcal{P}_k$ be the interpolating polynomial to f on (x_i). The error function $f - p$ has at least $k+1$ zeros in $[\bar{a}, \bar{b}]$ so, by Rolle's theorem, $f^{(k)} - p^{(k)}$ must vanish at some $\xi \in [\bar{a}, \bar{b}]$, i.e.,

$$p^{(k)}(\xi) = f^{(k)}(\xi)$$

On the other hand, if $p(x) = a_k x^k +$ (lower order terms), then (for any ξ)

$$p^{(k)}(\xi) = k!\, a_n =: k!\, f[x_0, ..., x_n].$$ ∎

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 2

2 Error bounds for polynomial interpolation

Here we study the *interpolation error*

$$e_n(x) = f(x) - p_n(x), \quad p_n \in \mathcal{P}_n,$$

for the class of differentiable functions f that possess $n+1$ continuous derivatives on the interval $[a,b]$; we denote this class by $C^{n+1}[a,b]$.

Theorem 2.1 *Let $f \in C^{n+1}[a,b]$, and let $p_n \in \mathcal{P}_n$ interpolate f at $n+1$ points $(x_i)_{i=0}^n \in [a,b]$, and let $\omega(x) := \prod_{i=0}^n (x - x_i)$. Then for every $x \in [a,b]$ there exists $\xi \in [a,b]$ such that*

$$\boxed{f(x) - p_n(x) = \frac{1}{(n+1)!}\omega(x) f^{(n+1)}(\xi)} \tag{2.1}$$

Proof. If x coincides with any x_i from the interpolating set, then both sides of (2.1) vanish, hence the formula trivially holds. So, we let x be any other fixed point, and consider the function

$$\phi(t) := \big[f(t) - p(t)\big] - c_x \omega(t)\,,$$

with *some* constant c_x. For *any* c_x, $\phi(t) = 0$ at $n+1$ points $t = x_i$, and we choose *particular* c_x so that $\phi(t) = 0$ at $t = x$ as well, i.e.,

$$c_x := \frac{f(x) - p(x)}{\omega(x)}\,.$$

Then ϕ has $n+2$ distinct zeros and, by Rolle's theorem, $\phi^{(n+1)}(\xi) = 0$ for some $\xi \in [a,b]$. So,

$$0 = \phi^{(n+1)}(\xi) = [f^{(n+1)}(\xi) - p^{(n+1)}(\xi)] - c_x \omega^{(n+1)}(\xi) = f^{(n+1)}(\xi) - c_x (n+1)!$$

whence

$$c_x := \frac{f(x) - p(x)}{\omega(x)} = \frac{1}{(n+1)!} f^{(n+1)}(\xi)\,,$$

and that is the same as (2.1). ∎

The equality (2.1) with the value $f^{(n+1)}(\xi)$ for *some* ξ is of hardly any use. Usually one has a bound for $f^{(n+1)}$ in terms of some *norm*, e.g., the L_∞-norm (the *max*-norm)

$$\|g\|_\infty := \|g\|_{L_\infty[a,b]} := \max_{t\in[a,b]} |g(t)|\,.$$

Then estimate (2.1) takes the form

$$\boxed{|f(x) - p_n(x)| \le \frac{1}{(n+1)!}|\omega(x)|\,\|f^{(n+1)}\|_\infty} \tag{2.2}$$

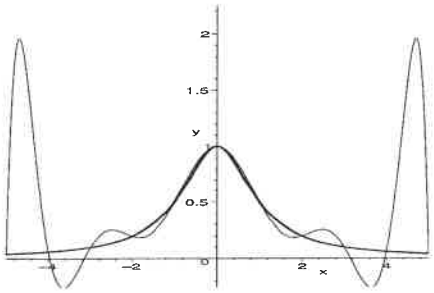
If we want to find the maximal error over the interval, then maximizing first the right- and then the left-hand side over $x \in [a,b]$ we get yet one more error bound for polynomial interpolation

$$\boxed{\|f - p_\Delta\|_\infty \le \frac{1}{(n+1)!}\|\omega_\Delta\|_\infty\,\|f^{(n+1)}\|_\infty} \tag{2.3}$$

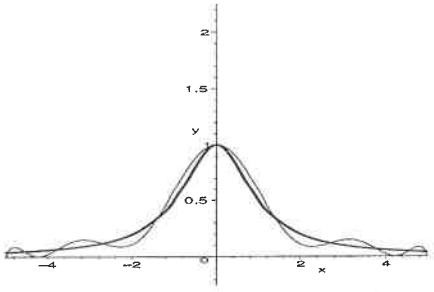
Here we put the lower index in ω_Δ in order to emphasize dependence of $\omega(x) := \prod_{i=0}^n (x - x_i)$ on the sequence of interpolating points $\Delta := (x_i)_{i=0}^n$. The choice of Δ makes a big difference!

Example 2.2

$$f(x) = \frac{1}{1+x^2}, \quad x \in [-5, 5].$$



a) Interpolation at uniform knots $\{-5+k\}_{k=0}^{10}$



b) Interpolation at Chebyshev knots $\{-5\cos\frac{2k+1}{22}\pi\}_{k=0}^{10}$

Definition 2.3 The *Chebyshev* polynomial of degree n on $[-1,1]$ is defined by

$$T_n(x) = \cos n\theta, \quad x = \cos\theta, \quad \theta \in [0, \pi].$$

(Or just $T_n(x) = \cos n \arccos x$, with $x \in [-1,1]$.) One sees at once that, on $[-1,1]$,

1) T_n takes its maximal absolute value 1 with alternating signs $n+1$ times:

$$\|T_n\|_\infty = 1, \quad T_n(t_k) = (-1)^k, \quad t_k = \cos\frac{\pi k}{n}, \quad k = \overline{0,n};$$

2) T_n has n distinct zeros: $\quad T_n(x_k^*) = 0, \quad x_k^* = \cos\frac{2k-1}{2n}\pi, \quad k = \overline{1,n}.$

Lemma 2.4 *The Chebyshev polynomials T_n satisfy the recurrence relation*

$$T_0(x) \equiv 1, \quad T_1(x) = x, \tag{2.4}$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n \geq 1. \tag{2.5}$$

In particular, T_n is indeed an algebraic polynomial of degree n with the leading coefficient 2^{n-1}.

Proof. Expressions (2.4) are straightforward, the recurrence follows via the substitution $x = \cos\theta$ into identity $\cos(n+1)\theta + \cos(n-1)\theta = 2\cos\theta\cos n\theta$ ∎

Theorem 2.5 *On the interval $[-1,1]$, among all polynomials of degree n with the leading coefficient equal to one, the Chebyshev polynomial γT_n has the smallest max-norm, i.e.,*

$$\inf_{(a_i)} \left\|x^n + a_{n-1}x^{n-1} + \cdots + a_0\right\|_\infty = \gamma \|T_n\|_\infty, \quad \gamma := 1/2^{n-1}.$$

Proof. Suppose there is a polynomial $q_n(x) = x^n + \cdots + a_0$ such that $\|q\|_\infty < \gamma$, and set

$$r := \gamma T_n - q_n.$$

1) The leading coefficients of both q_n and γT_n are equal 1, thus r is of degree at most $n-1$.

2) Further, at the points $t_k := \cos\frac{\pi k}{n}$, the Chebyshev polynomial γT_n takes the values $\pm\gamma$ alternatively, while by assumption $|q_n(t_k)| < \gamma$, hence r alternates in sign at these points, therefore it has a zero in each of n intervals (t_k, t_{k+1}), i.e. at least n zeros in the interval $[-1,1]$, a contradiction to $r \in \mathcal{P}_{n-1}$. ∎

Corollary 2.6 *For $\Delta = (x_i)_{i=0}^n \subset [-1,1]$, let $\omega_\Delta(x) = \prod_{i=0}^n (x - x_i)$. Then, for all n, we have*

$$\inf_\Delta \|\omega_\Delta\|_\infty = \|\omega_{\Delta_*}\|_\infty = 1/2^n.$$

Theorem 2.7 *For $f \in C^{n+1}[-1,1]$, the best choice of interpolating points is $\Delta_* = (x_i^*) = \left(\cos\frac{2i+1}{2n+2}\pi\right)_{i=0}^n$,*

$$\boxed{\|f - p_{\Delta_*}\|_\infty \leq \frac{1}{2^n}\frac{1}{(n+1)!}\|f^{(n+1)}\|_\infty}$$

Example 2.8 For $f(x) = e^x$, and $x \in [-1,1]$, the error of approximation provided by interpolating polynomial of degree 9 with 10 Chebyshev knots is bounded by

$$|e^x - p_9(x)| \leq \frac{1}{2^9}\frac{1}{10!}e \leq 1.5 \cdot 10^{-9}$$

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 3

3 Orthogonal polynomials

3.1 The three-term recurrence relation

Consider $\mathbb{X} = C[a,b]$, the space of all continuous real-valued functions $f : [a,b] \to \mathbb{R}$, and define a *scalar (or inner) product* on $C[a,b]$ by

$$(f,g) := (f,g)_w := \int_a^b f(x)g(x)\,w(x)\,dx. \tag{3.1}$$

Here w, the so-called *weight function*, is a fixed positive function, such that the integral $\int h(x)w(x)\,dx$ exists for all $h \in C[a,b]$.

We denote by $\mathcal{P}_n$ the space of all algebraic polynomials of degree (at most) n, i.e., $p \in \mathcal{P}_n$ if $p(x) = \sum_{k=0}^n a_k x^k$. If the *leading* coefficent a_n equals 1, then p is called a *monic* polynomial. Given a scalar product (3.1), we say that $Q_n \in \mathcal{P}_n$ is the n-th *orthogonal polynomial* if

$$(Q_n, p) = 0 \quad \forall p \in \mathcal{P}_{n-1}.$$

Note: different weights lead to different orthogonal polynomials.

Lemma 3.1 *For every $n \in \mathbb{N}$, there exists a unique monic orthogonal polynomial $Q_n \in \mathcal{P}_n$. Any $p \in \mathcal{P}_n$ is uniquely expressible as a linear combination*

$$p = \sum_{k=0}^n c_k Q_k, \quad c_k = (p, Q_k)/\|Q_k\|^2. \tag{3.2}$$

Proof. We apply the Gram-Schmidt orthogonalization algorithm for the linearly independent sequence of monomials $(1, x, \ldots, x^n, \ldots)$. Starting with $Q_0 \equiv 1$ we set

$$Q_n(x) := x^n - \sum_{k=0}^{n-1} \frac{(x^n, Q_k)}{(Q_k, Q_k)}\, Q_k(x), \quad n = 1, 2, \ldots.$$

Then, from construction, $(Q_n, Q_k) = 0$, i.e., Q_n is orthogonal to each previous Q_k. Also, we have $x^n \in \operatorname{span}(Q_k)_{k=0}^n$, hence $\mathcal{P}_n = \operatorname{span}(Q_k)_{k=0}^n$. Therefore $Q_n \perp \mathcal{P}_{n-1}$, and any $p \in \mathcal{P}_n$ has an expansion (3.2). Each coefficient c_k in (3.2) is uniquely determined by multiplying both sides (in the scalar product sense) with Q_k.

If $\widetilde{Q}_n$ is another n-th monic orthogonal polynomial, then $p := Q_n - \widetilde{Q}_n$ belongs to $\mathcal{P}_{n-1}$, therefore $(p,p) = (Q_n - \widetilde{Q}_n, p) = 0$. Hence $p \equiv 0$ (by the scalar product property), i.e., $\widetilde{Q}_n \equiv Q_n$. ∎

Remark 3.2 For practical construction of orthogonal polynomials the Gram-Schmidt is not of much help, for it suffers the same problem as the Gram-Schmidt algorithm in Euclidean space: loss of accuracy due to imprecisions in calculation of scalar products. A considerably better procedure is being provided by the next theorem.

Theorem 3.3 (The three-term recurrence relation) *Monic orthogonal polynomials satisfy the relation*

$$Q_{n+1}(x) = (x - a_n)Q_n(x) - b_n Q_{n-1}(x), \qquad n = 0, 1, \ldots \tag{3.3}$$

where $Q_{-1}(x) \equiv 0$, $Q_0(x) \equiv 1$, and

$$a_n := \frac{(xQ_n, Q_n)}{(Q_n, Q_n)}, \qquad b_n := \frac{(Q_n, Q_n)}{(Q_{n-1}, Q_{n-1})} > 0.$$

Proof. Based on (3.2), let us look at the coefficients of the expansion

$$xQ_n(x) = \sum_{k=0}^{n+1} c_k Q_k(x), \qquad c_k = \frac{(xQ_n, Q_k)}{(Q_k, Q_k)} = \frac{(Q_n, xQ_k)}{(Q_k, Q_k)}.$$

$k = n+1 \;\to\; c_{n+1} = 1$, since both $xQ_n(x)$ and $Q_{n+1}(x)$ are monic polynomials of degree $n+1$.
$k = n \;\to\; c_n = a_n$.
$k = n-1 \;\to$ Because of monicity, we have the equality $xQ_{n-1} = Q_n + p_{n-1}$ where $p_{n-1} \in \mathcal{P}_{n-1}$, so that $(Q_n, xQ_{n-1}) = (Q_n, Q_n + p_{n-1}) = (Q_n, Q_n)$, hence $c_{n-1} = b_n$.
$k < n-1 \;\to$ Then $xQ_k \in \mathcal{P}_{n-1}$, and we obtain $(Q_n, xQ_k) = 0$, thus $c_k = 0$.

It follows that $xQ_n(x) = Q_{n+1} + a_n Q_n + b_n Q_{n-1}$, and that is equivalent to (3.3). ∎

Remark 3.4 If (Q_k) have leading coefficients (α_k), then the recurrence takes the form

$$Q_{n+1}(x) = \frac{\alpha_{n+1}}{\alpha_n}(x - a_n)Q_n(x) - \frac{\alpha_{n+1}\alpha_{n-1}}{\alpha_n^2} b_n Q_{n-1}(x), \qquad n = 0, 1, \ldots$$

and, with an appropriate choice of (α_k), may become very simple.

3.2 Examples

Example 3.5 Classical examples of orthogonal (non-monic) polynomials include

Name	Notation	Interval	Weight function	Recurrence
Legendre	P_n	$[-1, 1]$	$w(x) \equiv 1$	$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x)$
Chebyshev	T_n	$[-1, 1]$	$w(x) = (1-x^2)^{-1/2}$	$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$
Laguerre	L_n	$[0, \infty)$	$w(x) = \mathrm{e}^{-x}$	$(n+1)L_{n+1}(x) = (2n+1-x)L_n(x) - nL_{n-1}(x)$
Hermite	H_n	$(-\infty, \infty)$	$w(x) = \mathrm{e}^{-x^2}$	$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x)$

Example 3.6 (The Chebyshev polynomials) The Chebyshev polynomials T_n, on line 2 in the table above, were introduced in Lecture 2 as

$$T_n(x) = \cos n \arccos x, \quad x \in [-1, 1],$$

where we also proved the three-term recurrence relation. Let us show that they are indeed orthogonal with the weight $w(x) = (1-x^2)^{-1/2}$. With the substistuion $x = \cos\theta$, $\theta \in [0, \pi]$ we have $T_n(x) = \cos n\theta$ and

$$\begin{aligned}(T_n, T_m)_w &:= \int_{-1}^{1} T_n(x)T_m(x)\frac{dx}{\sqrt{1-x^2}} = \int_0^{\pi} \cos n\theta \cos m\theta \, d\theta = \tfrac{1}{2}\int_{-\pi}^{\pi} \cos n\theta \cos m\theta \, d\theta \\ &= \tfrac{1}{4}\int_{-\pi}^{\pi} \left[\cos(n+m)\theta + \cos(n-m)\theta\right] d\theta = 0, \quad n \neq m.\end{aligned}$$

3.3 Least squares polynomial fitting

Let f be a continuous function defined on some interval $[a, b]$, and suppose we wish to approximate f by a polynomial of degree n. If we equip $C[a, b]$ with the distance $\|f-g\| := (f-g, f-g)^{1/2}$ induced by a scalar product $(f, g) = \int_a^b f(x)g(x)\,w(x)dx$, then it is natural to seek a polynomial

$$p^* := \arg\min_{p \in \mathcal{P}_n} \|f - p\|,$$

for which the distance $\|f - p^*\|$ is as small as possible. Such a polynomial is called a *(weighted) least squares approximant*.

Theorem 3.7 *Let $(Q_k)_{k=0}^n$ be polynomials orthogonal with respect to a given inner product. Then the least squares approximant to any $f \in C[a,b]$ from $\mathcal{P}_n$ is given by the formula*

$$p^* = p^*(f) = \sum_{k=0}^{n} c_k^* Q_k, \quad c_k^* = \frac{(f,Q_k)}{\|Q_k\|^2}, \tag{3.4}$$

and the value of the least squares approximation is

$$\|f - p^*\|^2 = \|f\|^2 - \sum_{k=1}^{n} \frac{(f,Q_k)^2}{\|Q_k\|^2}. \tag{3.5}$$

Remark 3.8 By orthogonality of Q_k, the norm of the extremal p^* is equal to

$$\|p^*\|^2 = \|\sum_{k=0}^{n} c_k^* Q_k\|^2 = \sum_{k=0}^{n} |c_k^*|^2 \|Q_k\|^2 = \sum_{k=1}^{n} \frac{(f,Q_k)^2}{\|Q_k\|^2},$$

hence formula (3.5) takes the form

$$\|f - p^*\|^2 = \|f\|^2 - \|p^*\|^2,$$

which is a reminiscent of the Pythagoras theorem.

Proof. Let $p = \sum_{k=0}^{n} c_k Q_k$. Then

$$F(c) := (f-p, f-p) = (f - \sum_{k=0}^{n} c_k Q_k, f - \sum_{k=0}^{n} c_k Q_k) = \|f\|^2 - 2\sum_{k=0}^{n} c_k (f, Q_k) + \sum_{k=0}^{n} c_k^2 \|Q_k\|^2 \tag{3.6}$$

The right-hand side is a quadratic polynomial in each c_k, therefore it attains its minimum when

$$\frac{\partial F}{\partial c_k}\Big|_{c_k = c_k^*} = -2(f, Q_k) + 2c_k^* \|Q_k\|^2 = 0, \quad k = \overline{0,n},$$

hence conclusion (3.4). Putting optimal c_k^* into (3.6) we obtain (3.5).

Method 3.9 Suppose we want to approximate $f \in C[a,b]$ with a prescribed accuracy ϵ, i.e., to find $n = n(\epsilon)$ such that

$$\|f - p^*\| \le \epsilon, \qquad p^* \in \mathcal{P}_n.$$

From (3.5), it follows that the required value n can be calculated by summing the terms of $\frac{(f,Q_k)^2}{\|Q_k\|^2}$ until we reach the bound

$$\sum_{k=1}^{n} \frac{(f,Q_k)^2}{\|Q_k\|^2} \ge \|f\|^2 - \epsilon^2.$$

The next theorem assures that this bound can be reached.

Theorem 3.10 (The Parseval identity) *If $[a,b]$ is finite, then $\sum_{k=0}^{\infty} \frac{(f,Q_k)^2}{\|Q_k\|^2} = \|f\|^2$.*

Proof (incomplete). Let

$$\sigma_n^2 := \|p^*\|^2 = \sum_{k=0}^{n} \frac{(f,Q_k)^2}{\|Q_k\|^2},$$

hence

$$\inf_{p \in \mathcal{P}_n} \|f - p\|^2 = \|f - p^*\|^2 = \|f\|^2 - \sigma_n^2.$$

According to the *Weierstrass theorem*, any function in $C[a,b]$ can be approximated arbitrarily close by a polynomial, hence $\lim_{n\to\infty} \inf_{p\in\mathcal{P}_n} \|f-p\|^2 = 0$ and we deduce that $\sigma_n^2 \to \|f\|^2$ as $n \to \infty$. ∎

Remark 3.11 Analogous identity is valid for the trigonometric system (and actually for any complete orthogonal system).

This page is void

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 4

4 Approximation of linear functionals

Given a vector space $\mathbb{X}$ (e.g., $\mathbb{R}^n$, or $C^m[a,b]$), a *linear functional* is any linear mapping

$$\lambda : \mathbb{X} \to \mathbb{R} \qquad \Big(\text{so } \lambda(\alpha f + \beta g) = \alpha\lambda(f) + \beta\lambda(g), \quad \forall f, g \in \mathbb{X} \text{ and } \forall \alpha, \beta \in \mathbb{R}\Big).$$

We will treat the space $\mathbb{X} = C^{n+1}[a,b]$, and the functionals

$$1) \quad \lambda(f) = f^{(k)}(\bar{x}), \quad (\bar{x}, k \text{ being fixed}), \qquad 2) \quad \lambda(f) = \int_a^b f(x)\, w(x) dx\,. \tag{4.1}$$

Our goal is to find an approximation of the form

$$\lambda(f) \approx \sum_{i=0}^{N} a_i f(x_i), \qquad f \in C^{n+1}[a,b]\,. \tag{4.2}$$

For the functionals 1)–2) in (4.1), this is called *numerical differentiation* and *numerical integration*, respectively.

Method 4.1 A suggestive approximation method is to interpolate f by $p \in \mathcal{P}_n$ and take $\lambda(f) \approx \lambda(p)$. This is called the *interpolating formula* (of degree n), in this case $N = n$. We have already seen an interpolating formula, namely that of Lagrange for the functional $\lambda(f) = f(\bar{x})$:

$$f(\bar{x}) \approx p(\bar{x}) = \sum_{i=0}^{n} \ell_i(\bar{x}) f(x_i)\,.$$

By linearity of λ, we have $\lambda(p) = \sum_{i=0}^n \lambda(\ell_i) p(x_i)$, thus the interpolating formula has the form

$$\lambda(f) \approx \sum_{i=0}^{n} \lambda(\ell_i) f(x_i). \tag{4.3}$$

Here $(\ell_i)_{i=0}^n$ are the fundamental Lagrange polynomials of degree n, $\ell_i(x_j) = \delta_{ij}$.

Method 4.2 Another method is to require from the formula (4.2) to be *exact on* $\mathcal{P}_n$, that is to become equality for any $f \in \mathcal{P}_n$. In this case the number N of terms (almost) need not to be restricted, and if $N > n$, then it is certainly not a polynomial of degree n that substitutes the function. If $N < n$, then the formula is of *high accuracy*. However, if $N = n$, then both methods are the same.

Lemma 4.3 *The formula* $\lambda(f) \approx \sum_{i=0}^n a_i f(x_i)$ *is interpolating* $\quad\Leftrightarrow\quad$ *it is exact on* $\mathcal{P}_n$.

Proof. The interpolating formula (4.3) is exact on $\mathcal{P}_n$ by definition. Conversely, if the formula in the lemma is exact on $\mathcal{P}_n$, take $f(x) = \ell_j(x)$ to obtain $\lambda(\ell_j) = \sum_{i=0}^n a_i \ell_j(x_i) = a_j$, i.e., (4.3). ∎

4.1 Numerical integration

A formula of numerical integration (with some $w(x) \geq 0$)

$$\lambda(f) := I(f) := \int_a^b f(x)\, w(x)\, dx \approx \sum_{i=0}^{n} a_i f(x_i), \qquad f \in C[a,b],$$

is called a *quadrature formula* with *nodes* (x_i) and *weights* (a_i). By Lemma 4.3, for any $n+1$ *fixed* (x_i), the interpolating formula $a_i = I(\ell_i)$ is exact on $\mathcal{P}_n$. Can one find a *quadrature of higher accuracy* which is exact on $\mathcal{P}_m$ with some $m > n$? Since we are free in choosing $n+1$ nodes (x_i), we may hope to increase the degree of accuracy from degree n up to $2n+1$. More is impossible.

Claim 4.4 *No quadrature formula with $n+1$ nodes is exact for all $p \in \mathcal{P}_m$ if $m \geq 2n+2$.*

Proof. Take $p(x) = \prod_{i=0}^{n}(x-x_i)^2 \geq 0$. Then $p \in \mathcal{P}_{2n+2}$, and $I(p) > 0$, but $\sum_{i=0}^{n} a_i p(x_i) = 0$ for any a_is. Hence the integral and the quadrature do not match. ∎

Our next goal is to show that $m = 2n+1$ can be attained. For this we need

Lemma 4.5 *Let Q_{n+1} be orthogonal to all $p_n \in \mathcal{P}_n$ on $[a,b]$. Then all the zeros of Q_{n+1} are real, distinct and lie in the interval (a,b).*

Proof. Denote by k the number of the sign changes of Q_{n+1} in (a,b) and assume that $k \leq n$. If $k = 0$ set $p_k \equiv 1$, and if $1 \leq k \leq n$ set $p_k(x) := \prod_{i=1}^{k}(x-t_i)$ where t_is are the points where a sign change of Q_{n+1} occurs. Then $p_k \in \mathcal{P}_k$, $k \leq n$, hence $(Q_{n+1}, p_k) = 0$. On the other hand, by construction, $Q_{n+1}(x)p_k(x)$ does not change sign throughout $[a,b]$ and vanishes at a finite number of points, hence $|(Q_{n+1}, p_k)| := \left|\int_a^b Q_{n+1}(x)p_k(x)w(x)dx\right| = \int_a^b |Q_{n+1}(x)p_k(x)|\, w(x)dx > 0$, a contradiction. Thus, $k \geq n+1$ (hence $k = n+1$) and the statement follows. ∎

Theorem 4.6 *Let a quadrature with $n+1$ nodes $(x_i)_{i=0}^n$ be exact on $\mathcal{P}_n$ (i.e. interpolating). Then it is exact on $\mathcal{P}_{2n+1}$ if and only if its nodes $(x_i)_{i=0}^n$ are the zeros of the $(n+1)$-st orthogonal polynomial Q_{n+1}.*

Proof. 1) If a quadrature with $n+1$ nodes (x_i) is exact for all $p \in \mathcal{P}_{2n+1}$, then letting $Q_{n+1}(x) := \prod_{i=0}^{n}(x-x_i) \in \mathcal{P}_{n+1}$ and taking any $q_n \in \mathcal{P}_n$ we find

$$\int_a^b Q_{n+1}(x)q_n(x)w(x)dx = \sum_{i=0}^{n} a_i[Q_{n+1}(x_i)q_n(x_i)] = 0$$

i.e., Q_{n+1} is orthogonal to all $q_n \in \mathcal{P}_n$.

2) Conversely, let $(x_i)_{i=0}^n$ be the zeros of orthogonal Q_{n+1}. Given any $p_{2n+1} \in \mathcal{P}_{2n+1}$, we can represent it uniquely as

$$p_{2n+1}(x) = Q_{n+1}(x)r_n(x) + s_n(x), \quad \text{with some} \quad r_n, s_n \in \mathcal{P}_n$$

(in fact, s_n interpolates p_{2n+1} at the points (x_i)). Since Q_{n+1} is orthogonal to r_n, we have

$$I(p_{2n+1}) = \int_a^b p_{2n+1}(x)\, w(x)dx = \int_a^b s_n(x)\, w(x)dx = I(s_n).$$

On the other hand, because $Q_{n+1}(x_i) = 0$,

$$\sum_{i=0}^{n} a_i p_{2n+1}(x_i) = \sum_{i=1}^{n} a_i s_n(x_i).$$

But $s_n \in \mathcal{P}_n$, while the quadrature is exact on $\mathcal{P}_n$, hence $I(s_n) = \sum a_i s_n(x_i)$, i.e., the right-hand sides of two previous equalities coincide, therefore the left-hand sides coincide too, i.e., $I(p_{2n+1}) = \sum a_i p_{2n+1}(x_i)$. ∎

Definition 4.7 A quadrature with $n+1$ nodes that is exact on $\mathcal{P}_{2n+1}$ is called *Gaussian quadrature.*

Example 4.8 For $[a,b] = [-1,1]$ and $w(x) \equiv 1$, the underlying orthogonal polynomials are the *Legendre polynomials.* The corresponding weights and nodes of the Gaussian quadratures are as follows.

$P_1(x) = x$	$x_0 = 0$	$a_0 = 2;$
$P_2(x) = \frac{3}{2}(x^2 - \frac{1}{3})$	$x_0 = -\frac{1}{\sqrt{3}},\ x_1 = \frac{1}{\sqrt{3}}$	$a_0 = 1,\ a_1 = 1$
$P_3(x) = \frac{5}{2}(x^3 - \frac{3}{5}x)$	$x_0 = -\sqrt{\frac{3}{5}},\ x_1 = 0,\ x_2 = \sqrt{\frac{3}{5}}$	$a_0 = \frac{5}{9},\ a_1 = \frac{8}{9},\ a_2 = \frac{5}{9}$

Example 4.9 For $[a,b] = [-1,1]$ and $w(x) = (1-x^2)^{-1/2}$, the orthogonal polynomials are the *Chebyshev polynomials* and the qudrature rule is particularly attractive:

$$T_{n+1}(x) = \cos(n+1)\arccos x, \qquad x_i = \cos\tfrac{2i+1}{2n+2}\pi, \qquad a_i = \tfrac{\pi}{n+1}, \qquad i = 0, \ldots, n.$$

Example 4.10 *(Simplest quadrature formulae ($w \equiv 1$)).*

the rectangle rule:	$\boxed{I(f) \approx (b-a)f(a)}$	1-point, non-Gaussian exact on constants
the midpoint rule:	$\boxed{I(f) \approx (b-a)f(\frac{a+b}{2})}$	1-point, Gaussian, exact on linear fns
the trapezoidal rule:	$\boxed{I(f) \approx (b-a)[\frac{1}{2}f(a) + \frac{1}{2}f(b)]}$	2-point, non-Gaussian, exact on linear fns
the Simpson rule:	$\boxed{I(f) \approx (b-a)[\frac{1}{6}f(a) + \frac{2}{3}f(\frac{a+b}{2}) + \frac{1}{6}f(b)]}$	3-point, non-Gaussian, but of higher accuracy (exact on cubics)

4.2 Numerical differentiation

Consider the interpolating formulae for numerical differentiation

$$\lambda(f) = f^{(k)}(\bar{x}) \approx \textstyle\sum_{i=0}^{n} a_i f(x_i), \qquad a_i = \lambda(\ell_i) = \ell_i^{(k)}(\bar{x}),$$

which are exact on the polynomials of degree n. The simplest ones are with $n = k$, i.e., $f^{(k)}(\bar{x}) \approx p^{(k)}(\bar{x})$ where p is the interpolating polynomial of degree k. But $p^{(k)}(\bar{x})$ is $k!$ times the leading coefficient of p, i.e., $k!f[x_0, \ldots, x_k]$, and we obtain the simplest rules

$$f^{(k)}(\bar{x}) \approx k!f[x_0, \ldots, x_k]\,.$$

Example 4.11

the forward difference:	$\boxed{f'(x) \approx f[x, x+h] = \dfrac{f(x+h) - f(x)}{h}}$	2-point, exact on linear fns
the central difference:	$\boxed{f'(x) \approx f[x-h, x+h] = \dfrac{f(x+h) - f(x-h)}{2h}}$	2-point, of higher accuracy, exact on quadratics
the 2-nd central difference:	$\boxed{f''(x) \approx 2f[x-h, x, x+h] = \dfrac{f(x+h) - 2f(x) + f(x-h)}{h^2}}$	3-point, of higher accuracy, exact on cubics

Example 4.12 $n = 2$, $k = 1$, $[a, b] = [0, 2]$. (Of course, one can transform any formula to any interval.)

$$f'(0) \approx p_2'(0) = -\tfrac{3}{2}f(0) + 2f(1) - \tfrac{1}{2}f(2)\,. \tag{4.4}$$

Here (and in any other formula), given the nodes (x_i), in our case $(0, 1, 2)$, we can find the corresponding coefficients (a_i) in two ways.

1) We determine fundamental lagrange polynomials ℓ_i (such that $\ell_i(x_j) = \delta_{ij}$),

$$\ell_0(x) = \tfrac{1}{2}(x-1)(x-2), \quad \ell_1(x) = -x(x-2), \quad \ell_2(x) = \tfrac{1}{2}x(x-1),$$

and set $a_i = \lambda(\ell_i)$:

$$a_0 = \ell_0'(0) = -\tfrac{3}{2}, \quad a_1 = \ell_1'(0) = 2, \quad a_2 = \ell_2'(0) = -\tfrac{1}{2}.$$

2) Sometimes it is easier to solve the system of linear equations which arises if we require the formula to be exact on monomials (x^m).

$$\begin{cases} a_0 + a_1 + a_2 = 0 & (f(x) = 1), \\ \quad a_1 + 2a_2 = 1 & (f(x) = x), \\ \quad a_1 + 4a_2 = 0 & (f(x) = x^2). \end{cases} \quad \Rightarrow \quad a_2 = -\tfrac{1}{2}, \quad a_1 = 2, \quad a_0 = -\tfrac{3}{2}\,.$$

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 5

5 Error of approximation

Given a linear functional λ and an approximation formula $\lambda(f) \approx \sum_{i=0}^{N} a_i f(x_i)$, we are interested in the error (which is a linear functional as well)

$$e_\lambda(f) := \lambda(f) - \sum_{i=0}^{N} a_i f(x_i)\,.$$

If λ acts on $f \in C^{n+1}[a,b]$, then it is natural to seek an estimate in terms of $\|f^{(n+1)}\|_\infty$, i.e.,

$$|e_\lambda(f)| \le c_\lambda \|f^{(n+1)}\|_\infty, \quad \forall f \in C^{n+1}[a,b]. \tag{5.1}$$

Since $f^{(n+1)} \equiv 0$ on $\mathcal{P}_n$, such an estimate can exist only if

$$e_\lambda(f) = 0 \quad \text{on} \quad \mathcal{P}_n,$$

i.e., the approximation formula *must be exact on* $\mathcal{P}_n$ (e.g., it can be interpolating of degree n).

If (5.1) holds with some c_λ and moreover, for any $\epsilon > 0$, there is an $f_\epsilon \in C^{n+1}[a,b]$ such that

$$|e_\lambda(f_\epsilon)| > (c_\lambda - \epsilon)\,\|f_\epsilon^{(n+1)}\|_\infty\,,$$

then the constant c_λ in inequality (5.1) is called *least* or *sharp*. The next section gives a general approach to obtaining *sharp* estimates for the error functionals e_λ that vanish on $\mathcal{P}_n$.

5.1 The Peano kernel theorem

Our point of departure is the *Taylor formula with an integral remainder term*,

$$f(x) = \sum_{i=0}^{n} \frac{1}{i!}(x-a)^i f^{(i)}(a) + \frac{1}{n!}\int_a^x (x-t)^n f^{(n+1)}(t)\,dt, \tag{5.2}$$

where the sum is the *Taylor polynomial* q_n to f (at the point a). One can can verify (5.2) by integration by parts. It is standard to write the remainder in the slightly different form, which makes the range of integration independent of x:

$$R(x) = \frac{1}{n!}\int_a^b (x-t)_+^n\, f^{(n+1)}(t)\,dt, \qquad \text{with} \quad (x-t)_+^n := \big(\max\{x-t,0\}\big)^n.$$

Let μ be a linear functional on C^{n+1}. Then

$$\mu(f) = \mu(q_n) + \mu(R) = \mu(q_n) + \frac{1}{n!}\,\mu\Big(\int_a^b (x-t)_+^n\, f^{(n+1)}(t)\,dt\Big).$$

Let us put formally μ under the integration sign (that is exchange action of μ and of $\int_a^b$), so that, for any *fixed* value of t, it will act on the function

$$g_t(\bullet) := (\bullet - t)_+^n\,,$$

a function, say, of x which (for a given t) is defined as $g_t(x) := (x-t)_+^n$, $x \in [a,b]$. To each value $t \in \mathbb{R}$ there corresponds a value $\mu(g_t) \in \mathbb{R}$, thus we have a function

$$K_\mu(t) := \mu(g_t) := \mu\big((\bullet - t)_+^n\big), \qquad t \in \mathbb{R}\,.$$

It is called the *Peano kernel* of the functional μ. So, formally, we may write

$$\mu(f) = \mu(q_n) + \frac{1}{n!}\int_a^b K_\mu(t) f^{(n+1)}(t)dt. \tag{5.3}$$

Definition 5.1 Denote by Λ_0 the set of linear functionals which are linear combinations of the functionals of two types

$$1)\ \mu(f) = f^{(k)}(\bar{x}) \quad (0 \le k \le n), \quad \bar{x} \in [a,b], \qquad 2)\ \mu(f) = \int_a^{\bar{x}} f(x)\, w(x) dx, \quad \bar{x} \in [a,b].$$

Lemma 5.2 *(without proof) If $\mu \in \Lambda_0$, then equality (5.3) is valid (i.e., exchange of μ and $\int_a^b$ is justified).*

Theorem 5.3 (The Peano kernel theorem) *Let $\lambda \in \Lambda_0$ and let $\lambda(f) \approx \sum_{i=0}^{N} a_i f(x_i)$ be an approximation formula which is exact on $\mathcal{P}_n$. Then the error functional e_λ (and any other functional from Λ_0 that vanishes on $\mathcal{P}_n$) has the integral representation*

$$e_\lambda(f) = \frac{1}{n!} \int_a^b K_{e_\lambda}(t) f^{(n+1)}(t)\, dt\,. \tag{5.4}$$

Proof. The formula follows from (5.3), where we put e_λ instead of μ, and use assumptions $e_\lambda \in \Lambda_0$ and $e_\lambda(q_n) = 0$. ∎

5.2 Sharp error bounds

Theorem 5.4 *Under assumptions of the previous theorem, for any $f \in C^{n+1}$, we have the sharp inequality*

$$|e_\lambda(f)| \le c_\lambda \, \|f^{(n+1)}\|_\infty\,, \qquad c_\lambda = \frac{1}{n!} \, \|K_{e_\lambda}\|_1\,, \tag{5.5}$$

where $\|K_{e_\lambda}\|_1 := \int_a^b |K_{e_\lambda}(t)|\, dt$, and $\|f^{(n+1)}\|_\infty := \max_{t\in[a,b]} |f^{(n+1)}(t)|$.

Proof. Applying the inequality

$$\Big| \int_a^b g(t)h(t)dt \Big| \le \int_a^b |g(t)|\, dt \cdot \max_{t\in[a,b]} |h(t)| =: \|g\|_1 \|h\|_\infty \tag{5.6}$$

to the right-hand side of (5.4), we obtain (5.5).

Let us show that the constant c_λ in (5.5) is *sharp* (the least possible). In (5.6), equality occurs if we take $h(t) = \operatorname{sgn} g(t)$. Therefore, we get equality in (5.5) taking f_0 with $f_0^{(n+1)}(t) = \operatorname{sgn} K_{e_\lambda}(t)$. ∎

Remark 5.5 A minor problem in the above proof of sharpness of c_λ is that the sign-function $f_0^{(n+1)}$ that provides equality in (5.5) is not continuous, whereas we consider $f \in C^{n+1}$.

Informally, we just *allow* the functions with piecewise continuous $(n+1)$-st derivative, like $|x|^{n+1}$, to be considered together with $f \in C^{n+1}$.

Formally, we can approximate a discontinuous sign-function $f_0^{(n+1)}$ by a continuous function $f_\epsilon^{(n+1)}$ changing its values on arbitrarily small intervals, say of of length ϵ', around the jumps. This will change the integral value in (5.4) by ϵ'', hence for such f_ϵ we get the estimate $e_\lambda(f_\epsilon) > (c_\lambda - \epsilon)\|f_\epsilon^{(n+1)}\|_\infty$. Thus, c_λ in (5.5) is the least possible constant on C^{n+1} as well.

Finally, here are two general results on the sharp constants in numerical integration and numerical differentiation.

Theorem 5.6 (Error of Gaussian quadrature) *Given $n \in \mathbb{N}$ and $w(x) \ge 0$, let $(a_i)_{i=0}^n$ and $(x_i)_{i=0}^n$ be the weights and nodes of the Gaussian quadrature, respectively, i.e., (x_i) are zeros of orthogonal polynomial Q_{n+1}. Then we have the sharp inequality*

$$\Big| \int_a^b f(x)\, w(x) dx - \sum_{i=0}^{n} a_i f(x_i) \Big| \le \frac{1}{(2n+2)!} \, \|Q_{n+1}\|_2^2 \, \|f^{(2n+2)}\|_\infty\,.$$

Theorem 5.7 (Shadrin) *For any $n \in \mathbb{N}$ and any $\Delta = (x_i)_{i=0}^n$, let $\ell_\Delta \in \mathcal{P}_n$ be the polynomial that interpolates f on Δ. Then, for any $1 \le k \le n$, we have the sharp inequality*

$$\|f^{(k)} - \ell_\Delta^{(k)}\|_\infty \le \frac{1}{(n+1)!} \, \|\omega_\Delta^{(k)}\|_\infty \, \|f^{(n+1)}\|_\infty\,.$$

5.3 Examples

Example 5.8 Take formula (4.4):

$$f'(0) \approx p_2'(0) = -\tfrac{3}{2}f(0) + 2f(1) - \tfrac{1}{2}f(2).$$

It is exact on $\mathcal{P}_2$, hence the error on $C^3[0,2]$ can be expressed as

$$e(f) := f'(0) - \left[-\tfrac{3}{2}f(0) + 2f(1) - \tfrac{1}{2}f(2)\right] = \frac{1}{2!}\int_0^2 K(t)f'''(t)dt.$$

Here, with $g_t(x) := (x-t)_+^2$, the Peano kernel K has the form

$$\begin{aligned} K(t) = e(g_t) &= 2(0-t)_+^1 + \tfrac{3}{2}(0-t)_+^2 - 2(1-t)_+^2 + \tfrac{1}{2}(2-t)_+^2 \\ &= \begin{cases} -2(1-t)^2 + \frac{1}{2}(2-t)^2 \\ \qquad\qquad \frac{1}{2}(2-t)^2 \end{cases} = \begin{cases} 2t - \frac{3}{2}t^2, & t \in [0,1], \\ \frac{1}{2}(2-t)^2, & t \in [1,2]. \end{cases} \end{aligned}$$

So, $K(t) \geq 0$, and integration gives the value

$$\|K\|_1 := \int_0^2 |K(t)|\,dt = \left(\int_0^1 + \int_1^2\right) K(t)\,dt = \frac{1}{2} + \frac{1}{6} = \frac{2}{3}.$$

Therefore, we have the estimate

$$|e(f)| \leq \tfrac{1}{2!}\,\tfrac{2}{3}\,\|f'''\|_\infty = \tfrac{1}{3}\,\|f'''\|_\infty.$$

The constant $c = \frac{1}{3}$ is sharp: since $K(t) \geq 0$, we will get equality for $f''' \equiv \text{const}$, e.g. for $f(x) = x^3$.

Example 5.9 The Simpson's rule,

$$\int_{-1}^1 f(t)\,dt \approx \int_{-1}^1 p_2(t)\,dt = \tfrac{1}{3}f(-1) + \tfrac{4}{3}f(0) + \tfrac{1}{3}f(1),$$

is exact on $\mathcal{P}_2$, hence the error on $C^3[-1,1]$ is

$$e(f) = \int_{-1}^1 f(t)\,dt - \tfrac{1}{3}f(-1) - \tfrac{4}{3}f(0) - \tfrac{1}{3}f(1) = \frac{1}{2!}\int_{-1}^1 K(t)f'''(t)\,dt,$$

where with $g_t(x) := (x-t)_+^2$

$$\begin{aligned} K(t) = e(g_t) &= \int_{-1}^1 (x-t)_+^2 dx - \tfrac{1}{3}(-1-t)_+^2 - \tfrac{4}{3}(0-t)_+^2 - \tfrac{1}{3}(1-t)_+^2 \\ &= \begin{cases} \frac{1}{3}(1-t)^3 - \frac{4}{3}t^2 - \frac{1}{3}(1-t)^2 \\ \frac{1}{3}(1-t)^3 \qquad\quad - \frac{1}{3}(1-t)^2 \end{cases} = \begin{cases} -\frac{1}{3}\,t\,(1+t)^2, & t \in [-1,0] \\ -\frac{1}{3}\,t\,(1-t)^2, & t \in [0,1] \end{cases} \end{aligned}$$

Now, $K(t)$ changes its sign at $t = 0$, so its L_1-norm has the value

$$\|K\|_1 := \int_{-1}^1 |K(t)|\,dt = \left(\int_{-1}^0 - \int_0^1\right) K(t)\,dt = 2 \cdot \tfrac{1}{3}\left(\tfrac{1}{2} - \tfrac{2}{3} + \tfrac{1}{4}\right) = \tfrac{1}{18},$$

so that

$$e(f) \leq \tfrac{1}{2!}\,\tfrac{1}{18}\,\|f'''\|_\infty = \tfrac{1}{36}\,\|f'''\|_\infty.$$

The constant $c = \frac{1}{36}$ is sharp again: since $K(t)$ changes its sign at one point $t = 0$, we get equality for $f'''(t) = C\,\mathrm{sgn}\,t$, e.g. for $f(x) = |x|^3$. For this f its third derivative has a jump, but for any $\epsilon > 0$ it can be approximated by $f_\epsilon \in C^3$ for which the constant would be $c > \frac{1}{36} - \epsilon$, i.e.

$$e(f_\epsilon) > \left(\frac{1}{36} - \epsilon\right)\|f_\epsilon'''\|_\infty.$$

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 6

6 Ordinary differential equations - basics

Problem 6.1 We wish to approximate the exact solution of the *ordinary differential equation (ODE)*

$$y' = f(t, y), \qquad 0 \le t \le T, \tag{6.1}$$

where $y \in \mathbb{R}^d$ and the function $f : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ is sufficiently 'smooth'. In principle, it is enough for f satisfy the Lipschitz condition with respect to the second argument to ensure that the solution exists and is unique, namely there exists $\lambda > 0$ such that

$$\|f(t, x) - f(t, y)\| \le \lambda \|x - y\|, \qquad t \in [0, T], \quad x, y \in \mathbb{R}^d. \tag{6.2}$$

Yet, for simplicity, we henceforth assume that f is analytic: in other words, we are always able to expand locally into Taylor series.

The equation (6.1) is accompanied by the initial condition $y(0) = y_0$.

For a small *time step* $h > 0$, we let $t_n = nh$ and our purpose is to approximate $y_{n+1} \approx y(t_{n+1})$, from $y_0, y_1, \ldots, y_n$ and equation (6.1).

Definition 6.2 (One-step method) A one-step method for solving (6.1) is a map $y_{n+1} = \varphi(t_n, y_n)$, i.e. an algorithm which allows y_{n+1} to depend only on t_n, y_n, h and f.

Method 6.3 (Euler's method) We approximate y' by the first difference $y'(t) \approx \frac{1}{h}(y(t+h) - y(t))$, thus obtaining the *Euler method*

$$y_{n+1} = y_n + hf(t_n, y_n), \qquad n = 0, 1, \ldots \tag{6.3}$$

On $[t_n, t_{n+1}]$, it approximates $y(t)$ with a straight line with slope $f(t_n, y_n)$.

Definition 6.4 Let $T > 0$ be given, and suppose that, for every $h > 0$, a method produces the sequence $y_n = y_{n,h}$, where $0 \le n \le \lfloor T/h \rfloor$. We say that the method *converges*, if $\max_n \|y_n - y(t_n)\| \to 0$ as $h \to 0$. It follows that if $nh \to t$ as $h \to 0$ and $n \to \infty$, then we have convergence $y_{n,h} \to y(t)$ to the exact solution of (6.1), uniformly for $t \in [0, T]$.

Theorem 6.5 *Suppose that f satisfies the Lipschitz condition (6.2). Then the Euler method (6.3) converges, and the error $e_n = y(t_n) - y_n$ admits the estimate*

$$\|e_n\| \le c_0 h\,. \tag{6.4}$$

Proof. The Taylor series for the exact solution y provides $y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{1}{2}h^2 y''(\tau_n)$, where τ_n is a point in the interval $[t_n, t_{n+1}]$. Therefore, using the identity $y'(t) = f(t, y(t))$, we may write

$$y(t_{n+1}) = y(t_n) + hf\big(t_n, y(t_n)\big) + \tfrac{1}{2}h^2 y''(\tau_n). \tag{6.5}$$

By subtracting formula (6.3) from this equation, and assuming that $\frac{1}{2}\|y''\|_\infty = c$, we find that the errors $e_n = y(t_n) - y_n$ satisfy the following relations

$$\begin{aligned} \|e_{n+1}\| &\le \|e_n\| + h\|f\big(t_n, y(t_n)\big) - f(t_n, y_n)\| + ch^2 \\ &\overset{(*)}{\le} \|e_n\| + \lambda h\,\|y(t_n) - y_n\| + ch^2 = (1 + \lambda h)\|e_n\| + ch^2\,, \end{aligned}$$

where in $(*)$ we used the Lipschitz condition on $\boldsymbol{f}$. Consequently, by induction,

$$\|\boldsymbol{e}_{n+1}\| \le (1+\lambda h)^m \|\boldsymbol{e}_{n+1-m}\| + ch^2 \sum_{i=0}^{m-1} (1+\lambda h)^i, \qquad m = 1, 2, \ldots, n+1.$$

In particular, letting $m = n+1$ and bearing in mind that $\boldsymbol{e}_0 = \boldsymbol{0}$, we have

$$\|\boldsymbol{e}_{n+1}\| \le ch^2 \sum_{i=0}^{n} (1+\lambda h)^i = ch^2 \frac{(1+\lambda h)^{n+1} - 1}{(1+\lambda h) - 1} \le \frac{ch}{\lambda} (1+\lambda h)^{n+1}.$$

Since $1 + \lambda h \le \mathrm{e}^{\lambda h}$ and $(n+1)h \le T$, we obtain that $(1+\lambda h)^{n+1} \le \mathrm{e}^{\lambda T}$, therefore

$$\|\boldsymbol{e}_n\| \le \frac{c\mathrm{e}^{\lambda T}}{\lambda} h \to 0 \quad (h \to 0), \quad \text{uniformly for} \quad 0 \le nh \le T,$$

and the theorem is true. □

Definition 6.6 The *local truncation error* of a general numerical method $\boldsymbol{y}_{n+1} = \varphi_h(t_n, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_n)$ for solving (6.1) is the error of the method on the true solution, i.e., the value $\boldsymbol{\eta}_{n+1}$ such that

$$\boldsymbol{y}(t_{n+1}) = \varphi_h(t_n, \boldsymbol{y}(t_0), \boldsymbol{y}(t_1), \ldots, \boldsymbol{y}(t_n)) + \boldsymbol{\eta}_{n+1}.$$

The *order* of the method is the largest integer $p \ge 0$ such that

$$\boldsymbol{\eta}_{n+1} = \mathcal{O}(h^{p+1})$$

for all $h > 0$, $n \ge 0$ and all sufficiently smooth functions $\boldsymbol{f}$ in (6.1). Note that, unless $p \ge 1$, the method is an unsuitable approximation to (6.1): in particular, $p \ge 1$ is necessary for convergence.

Remark 6.7 (The order of Euler's method) From (6.5), it follows that

$$\boldsymbol{\eta}_{n+1} = \tfrac{1}{2} h^2 \boldsymbol{y}''(\tau_n) = \mathcal{O}(h^2)$$

and we deduce that Euler's method is of order 1 (which is justified by (6.4))

Definition 6.8 (Theta methods) For $\theta \in [0,1]$, we consider methods of the form

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\left[\theta \boldsymbol{f}(t_n, \boldsymbol{y}_n) + (1-\theta)\boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1})\right], \qquad n = 0, 1, \ldots. \tag{6.6}$$

1) If $\theta = 1$, we recover Euler's method and the choices $\theta = 0$ and $\theta = \frac{1}{2}$ are known as

$$\begin{aligned} &\text{Backward Euler:} && \boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1}), \\ &\text{Trapezoidal rule:} && \boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \tfrac{1}{2}h\left[\boldsymbol{f}(t_n, \boldsymbol{y}_n) + \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_{n+1})\right]. \end{aligned}$$

2) If $\theta \in [0, 1)$, then the theta method (6.6) is *implicit*: each time step requires the solution of a system of d (in general, nonlinear) algebraic equations for the unknown vector $\boldsymbol{y}_{n+1}$.

Solution of nonlinear algebraic equations $\boldsymbol{F}(\boldsymbol{y}) = 0$ can be done by iteration. For example,

$$\begin{aligned} &\text{Direct iteration:} && \boldsymbol{y}^{[k+1]} = \boldsymbol{y}^{[k]} - \boldsymbol{F}(\boldsymbol{y}^{[k]}), \\ &\text{Newton–Raphson (N-R):} && \boldsymbol{y}^{[k+1]} = \boldsymbol{y}^{[k]} - [\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{y}^{[k]})]^{-1} \boldsymbol{F}(\boldsymbol{y}^{[k]}), \\ &\text{Modified N-R:} && \boldsymbol{y}^{[k+1]} = \boldsymbol{y}^{[k]} - [\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{y}^{[0]})]^{-1} \boldsymbol{F}(\boldsymbol{y}^{[k]}). \end{aligned}$$

Remark 6.9 (The order of the theta method) It follows from (6.6) and Taylor's theorem that the local truncation error of the theta method is

$$\begin{aligned} &\boldsymbol{y}(t_{n+1}) - \boldsymbol{y}(t_n) - h\big[\theta \boldsymbol{y}'(t_n) + (1-\theta)\boldsymbol{y}'(t_{n+1})\big] \\ &\quad = \big[h\boldsymbol{y}'(t_n) + \tfrac{1}{2}h^2\boldsymbol{y}''(t_n) + \tfrac{1}{6}h^3\boldsymbol{y}'''(t_n)\big] \\ &\qquad - \theta h\, \boldsymbol{y}'(t_n) - (1-\theta)h\left[\boldsymbol{y}'(t_n) + h\boldsymbol{y}''(t_n) + \tfrac{1}{2}h^2\boldsymbol{y}'''(t_n)\right] + \mathcal{O}(h^4) \\ &\quad = (\theta - \tfrac{1}{2})h^2\boldsymbol{y}''(t_n) + (\tfrac{1}{2}\theta - \tfrac{1}{3})h^3\boldsymbol{y}'''(t_n) + \mathcal{O}(h^4). \end{aligned}$$

Therefore the theta method is of order 1, except that the trapezoidal rule is of order 2.

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 7

7 Multistep methods

7.1 Order of the multistep methods

Definition 7.1 (Multistep methods) It is often useful to use several past solution values in computing a new value. Thus, assuming that $\boldsymbol{y}_n, \boldsymbol{y}_{n+1}, \dots, \boldsymbol{y}_{n+s-1}$ are available, where $s \geq 1$, we say that

$$\sum_{m=0}^{s} a_m \boldsymbol{y}_{n+m} = h \sum_{m=0}^{s} b_m \boldsymbol{f}_{n+m}, \qquad n = 0, 1, \dots, \tag{7.1}$$

where $a_s = 1$, and $\boldsymbol{f}_{n+m} = \boldsymbol{f}(t_{n+m}, \boldsymbol{y}_{n+m})$, is an s-step method. If $b_s = 0$, the method is *explicit*, otherwise it is *implicit*. If $s \geq 2$, we need to obtain extra *starting values* $\boldsymbol{y}_1, \dots, \boldsymbol{y}_{s-1}$ by different time-stepping method.

Theorem 7.2 *The multistep method (7.1) is of order $p \geq 1$ if and only if*

$$\sum_{m=0}^{s} a_m = 0, \qquad \sum_{m=0}^{s} m^k a_m = k \sum_{m=0}^{s} m^{k-1} b_m, \qquad k = 1..p. \tag{7.2}$$

Proof. Substituting the exact solution and expanding into Taylor series about t_n, we obtain

$$\begin{aligned}\sum_{m=0}^{s} a_m \boldsymbol{y}(t_{n+m}) - h \sum_{m=0}^{s} b_m \boldsymbol{y}'(t_{n+m}) &= \sum_{m=0}^{s} a_m \sum_{k=0}^{\infty} \frac{(mh)^k}{k!} \boldsymbol{y}^{(k)}(t_n) - h \sum_{m=0}^{s} b_m \sum_{k=1}^{\infty} \frac{(mh)^{k-1}}{(k-1)!} \boldsymbol{y}^{(k)}(t_n) \\ &= \Big(\sum_{m=0}^{s} a_m \Big) \boldsymbol{y}(t_n) + \sum_{k=1}^{\infty} \frac{h^k}{k!} \Big(\sum_{m=0}^{s} m^k a_m - k \sum_{m=0}^{s} m^{k-1} b_m \Big) \boldsymbol{y}^{(k)}(t_n).\end{aligned}$$

Thus, to obtain the local truncation error of order $\mathcal{O}(h^{p+1})$ regardless of the choice of $\boldsymbol{y}$, it is necessary and sufficient that the coefficients at h^k vanish for $k \leq p$, i.e., that (7.2) are satisfied. □

Remark 7.3 Since the Taylor expansion of polynomials of degree p contains only $\mathcal{O}(h^k)$ terms with $k \leq p$, the multistep method is of order p iff

$$\sum_{m=0}^{s} a_m Q(t_{n+m}) = h \sum_{m=0}^{s} b_m Q'(t_{n+m}), \quad \forall Q \in \mathcal{P}_p\,.$$

In particular, taking $Q(x) = x^k$ for $k = 0..p$ and $t_{n+m} = m$, $h = 1$, we obtain (7.2).

Given a multistep method (7.1), we define two polynomials of degree s:

$$\rho(w) = \sum_{m=0}^{s} a_m w^m, \quad \sigma(w) = \sum_{m=0}^{s} b_m w^m.$$

Theorem 7.4 *The multistep method (7.1) is of order $p \geq 1$ if and only if*

$$\rho(e^z) - z\sigma(e^z) = \mathcal{O}(z^{p+1}), \qquad z \to 0. \tag{7.3}$$

Proof. Expanding again into Taylor series,

$$\begin{aligned}\rho(e^z) - z\sigma(e^z) &= \sum_{m=0}^{s} a_m e^{mz} - z \sum_{m=0}^{s} b_m e^{mz} = \sum_{m=0}^{s} a_m \sum_{k=0}^{\infty} \frac{1}{k!} m^k z^k - z \sum_{m=0}^{s} b_m \sum_{k=0}^{\infty} \frac{1}{k!} m^k z^k \\ &= \sum_{k=0}^{\infty} \frac{z^k}{k!} \sum_{m=0}^{s} m^k a_m - \sum_{k=1}^{\infty} \frac{z^k}{(k-1)!} \sum_{m=0}^{s} m^{k-1} b_m \\ &= \left(\sum_{m=0}^{s} a_m \right) + \sum_{k=1}^{\infty} \frac{z^k}{k!} \left(\sum_{m=0}^{s} m^k a_m - k \sum_{m=0}^{s} m^{k-1} b_m \right).\end{aligned}$$

The theorem follows from (7.2). □

The reason that (7.3) is equivalent to (7.2) (and that their proofs are almost identical) is due to the (informal) relation between the Taylor series and the exponent e^{hD}, where D is the operator of differentiation. Namely

$$\boldsymbol{f}(x+h) = \Big(I + hD + \frac{1}{2} h^2 D^2 + \cdots \Big) \boldsymbol{f}(x) = e^{hD} \boldsymbol{f}(x).$$

Example 7.5 The 2-step *Adams–Bashforth method* is

$$\boldsymbol{y}_{n+2} - \boldsymbol{y}_{n+1} = h\Big[\tfrac{3}{2}\boldsymbol{f}_{n+1} - \tfrac{1}{2}\boldsymbol{f}_n\Big]. \tag{7.4}$$

Therefore $\rho(w) = w^2 - w$, $\sigma(w) = \frac{3}{2}w - \frac{1}{2}$ and

$$\rho(e^z) - z\sigma(e^z) = [1 + 2z + 2z^2 + \tfrac{4}{3}z^3] - [1 + z + \tfrac{1}{2}z^2 + \tfrac{1}{6}z^3] - \tfrac{3}{2}z[1 + z + \tfrac{1}{2}z^2] + \tfrac{1}{2}z + \mathcal{O}(z^4) = \tfrac{5}{12}z^3 + \mathcal{O}(z^4).$$

Hence the method is of order 2.

7.2 Convergence

Definition 7.6 (Convergence of a multistep method) Let a multistep method (7.1) be applied to the usual ODE (6.1), and let $\widehat{e}(h)$ and $e(h)$ be the numbers

$$\widehat{e}(h) := \max \|\boldsymbol{y}(t_i) - \boldsymbol{y}_i\| : i = 0...s-1\}, \qquad e(h) := \max \|\boldsymbol{y}(t_i) - \boldsymbol{y}_i\| : i = 0...N\}.$$

The method is defined to be convergent if, for every ODE whose right-hand side satisfies the Lipshitz condition, the limits $h \to 0$ and $\widehat{e}(h) \to 0$ imply $e(h) \to 0$.

Example 7.7 (Absence of convergence) Consider the 2-step method

$$\boldsymbol{y}_{n+2} + 4\boldsymbol{y}_{n+1} - 5\boldsymbol{y}_n = h(4\boldsymbol{f}_{n+1} + 2\boldsymbol{f}_n) \tag{7.5}$$

Now $\rho(w) = w^2 + 4w - 5$, $\sigma(w) = 4w + 2$ and it is easy to verify that the method is of order 3. Let us apply it, however, to the trivial ODE $y' = 0$, $y(0) = 1$. Hence a single step reads $y_{n+2} + 4y_{n+1} - 5y_n = 0$ and the general solution of this recursion is $y_n = c_1 1^n + c_2(-5)^n$, where c_1, c_2 are arbitrary constants, which are determined by $y_0 = 1$ and our value of y_1. If $y_1 \neq 1$, i.e. if there is a small error in our starting values, then $c_2 \neq 0$, thus $|y_n| \to \infty$ and we cannot recover the exact solution $y(t) \equiv 1$.

As a more convincing example, we may consider ODE $y' = -y$, $y(0) = 1$, with the exact solution $y(t) = e^{-t}$. Even if we take the exact $y_1 = e^{-h}$, the sequence (y_n) will grow like $h^4(-5)^n$. So, it is not only the order that matters.

Definition 7.8 (Root condition) We say that a polynomial obeys the *root condition* if all its zeros reside in $|w| \leq 1$ and all zeros of unit modulus are simple. (If ρ obeys the root condition, the method (7.1) is sometimes said to be *zero-stable*.)

Theorem 7.9 (The Dahlquist equivalence theorem) *The multistep method (7.1) is convergent if and only if it is of order $p \geq 1$ and the polynomial ρ obeys the root condition.*

For the method (7.4) we have $\rho(w) = w(w-1)$, and the root condition is obeyed. However, for (7.5) we obtain $\rho(w) = (w+5)(w-1)$, the root condition fails and we deduce that there is no convergence.

7.3 Construction

Technique 7.10 A useful procedure to generate multistep methods which are convergent and of high order is as follows. According to (7.2), order $p \geq 1$ implies $\rho(1) = 0$. Choose an arbitrary s-degree polynomial ρ that obeys the root condition and such that $\rho(1) = 0$. To maximize order, we let σ be the polynomial of degree s for implicit methods (alternatively, of degree $(s-1)$ for explicit methods) arising from the truncation of the Taylor expansion of $\frac{\rho(w)}{\log w}$ about the point $w = 1$. Thus, for example, for an *implicit method*,

$$\sigma(w) = \frac{\rho(w)}{\log w} + \mathcal{O}(|w-1|^{s+1}) \quad \Leftrightarrow \quad \rho(e^z) - z\sigma(e^z) = \mathcal{O}(z^{s+2})$$

and (7.3) implies order at least $s+1$ (for explicit methods order s).

Example 7.11 The choice $\rho(w) = w^{s-1}(w-1)$ corresponds to *Adams methods:*

1) *Adams–Bashforth methods* if $b_s = 0$, hence explicit, with the order s (e.g. (7.4) for $s = 2$),

2) *Adams–Moulton methods* if $b_s \neq 0$, hence implicit, but with the order $(s+1)$.

For example, letting $s = 2$ and $\xi = w - 1$, we obtain the 3rd-order Adams–Moulton method by expanding

$$\begin{aligned}
\frac{w(w-1)}{\log w} &= \frac{\xi + \xi^2}{\log(1+\xi)} = \frac{\xi + \xi^2}{\xi - \frac{1}{2}\xi^2 + \frac{1}{3}\xi^3 - \cdots} = \frac{1 + \xi}{1 - \frac{1}{2}\xi + \frac{1}{3}\xi^2 - \cdots} \\
&= (1+\xi)\Big[1 + (\tfrac{1}{2}\xi - \tfrac{1}{3}\xi^2) + (\tfrac{1}{2}\xi - \tfrac{1}{3}\xi^2)^2 + \mathcal{O}(\xi^3)\Big] = 1 + \tfrac{3}{2}\xi + \tfrac{5}{12}\xi^2 + \mathcal{O}(\xi^3) \\
&= 1 + \tfrac{3}{2}(w-1) + \tfrac{5}{12}(w-1)^2 + \mathcal{O}|w-1|^3 = -\tfrac{1}{12} + \tfrac{2}{3}w + \tfrac{5}{12}w^2 + \mathcal{O}(|w-1|^3).
\end{aligned}$$

Therefore the 2-step, 3rd-order Adams–Moulton method is

$$\boldsymbol{y}_{n+2} - \boldsymbol{y}_{n+1} = h\Big[\tfrac{5}{12}\boldsymbol{f}_{n+2} + \tfrac{2}{3}\boldsymbol{f}_{n+1} - \tfrac{1}{12}\boldsymbol{f}_n\Big].$$

(If we cut expansion at $1 + \frac{3}{2}\xi = 1 + \frac{3}{2}(w-1) = \frac{3}{2}w - \frac{1}{2}$, we get explicit 2-step 2nd-order method (7.4).)

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 8

8 Multistep methods (contd)

8.1 Construction (contd)

Technique 8.1 A useful procedure to generate multistep methods which are convergent and of high order is as follows. According to (7.2), order $p \geq 1$ implies $\rho(1) = 0$. Choose an arbitrary s-degree polynomial ρ that obeys the root condition and such that $\rho(1) = 0$. To maximize order, we let σ be the polynomial of degree s for implicit methods (alternatively, of degree $(s-1)$ for explicit methods) arising from the truncation of the Taylor expansion of $\frac{\rho(w)}{\log w}$ about the point $w = 1$. Thus, for example, for an *implicit method*,

$$\sigma(w) = \frac{\rho(w)}{\log w} + \mathcal{O}(|w-1|^{s+1}) \quad \Leftrightarrow \quad \rho(e^z) - z\sigma(e^z) = \mathcal{O}(z^{s+2})$$

and (7.3) implies order at least $s+1$ (for explicit methods order s).

Example 8.2 The choice $\rho(w) = w^{s-1}(w-1)$ corresponds to *Adams methods:*

1) *Adams–Bashforth methods* if $b_s = 0$, hence explicit, with the order s (e.g. (7.4) for $s = 2$),

2) *Adams–Moulton methods* if $b_s \neq 0$, hence implicit, but with the order $(s+1)$.

For example, letting $s = 2$ and $\xi = w - 1$, we obtain the 3rd-order Adams–Moulton method by expanding

$$\begin{aligned}\frac{w(w-1)}{\log w} &= \frac{\xi+\xi^2}{\log(1+\xi)} = \frac{\xi+\xi^2}{\xi-\frac{1}{2}\xi^2+\frac{1}{3}\xi^3-\cdots} = \frac{1+\xi}{1-\frac{1}{2}\xi+\frac{1}{3}\xi^2-\cdots} \\ &= (1+\xi)\Big[1+(\tfrac{1}{2}\xi-\tfrac{1}{3}\xi^2)+(\tfrac{1}{2}\xi-\tfrac{1}{3}\xi^2)^2+\mathcal{O}(\xi^3)\Big] = 1+\tfrac{3}{2}\xi+\tfrac{5}{12}\xi^2+\mathcal{O}(\xi^3) \\ &= 1+\tfrac{3}{2}(w-1)+\tfrac{5}{12}(w-1)^2+\mathcal{O}|w-1|^3 = -\tfrac{1}{12}+\tfrac{2}{3}w+\tfrac{5}{12}w^2+\mathcal{O}(|w-1|^3).\end{aligned}$$

Therefore the 2-step 3rd-order Adams–Moulton method is

$$y_{n+2} - y_{n+1} = h\Big[\tfrac{5}{12}f_{n+2} + \tfrac{2}{3}f_{n+1} - \tfrac{1}{12}f_n\Big].$$

(If we cut expansion at $1+\frac{3}{2}\xi = 1+\frac{3}{2}(w-1) = \frac{3}{2}w - \frac{1}{2}$, we get explicit 2-step 2nd-order method (7.4).)

Method 8.3 (BDF) As another exercise in applying Technique 8.1 (and for future applications) let us construct s-step s-order methods such that $\sigma(w) = b_s w^s$ for some non-zero b_s, say $b_s = 1$. In other words,

$$\sum_{m=0}^{s} a_m y_{n+m} = h f_{n+s}, \qquad n = 0, 1, \ldots. \tag{8.1}$$

Such methods are called *backward differentiation formulae (BDF)*, and their coefficients (a_m) can be obtained from the following expression.

Lemma 8.4 *The polynomial $\rho(w) = \sum_{m=0}^{s} a_m w^m$ of the s-step s-order BDF method (8.1) has the form*

$$\rho(w) = \sum_{k=1}^{s} \frac{1}{k} w^{s-k}(w-1)^k. \tag{8.2}$$

Proof. We need to fulfill the s-order condition $\rho(e^z) - z\sigma(e^z) = \mathcal{O}(z^{s+1})$ which, with $w = e^z$ and $\sigma(w) = w^s$, becomes

$$\rho(w) = w^s \log w + \mathcal{O}(|w-1|^{s+1}).$$

But we have $\log w = -\log\left(\frac{1}{w}\right) = -\log\left(1 - \frac{w-1}{w}\right)$, and $\log(1-x) = -\sum_{k=1}^{\infty}\frac{1}{k}x^k$, therefore

$$w^s \log w = w^s \sum_{k=1}^{s} \tfrac{1}{k}\left(\tfrac{w-1}{w}\right)^k + \mathcal{O}(|w-1|^{s+1}),$$

and the first term is a polynomial of degree s, hence the result. □

Example 8.5 For $s=2$, we obtain $\frac{3}{2}y_{n+2}-2y_{n+1}+\frac{1}{2}y_n = hf_{n+2}$, or with the standard normalization

$$y_{n+2}-\tfrac{4}{3}y_{n+1}+\tfrac{1}{3}y_n = \tfrac{2}{3}hf_{n+2}\,.$$

Warning. We cannot take it for granted that BDF methods are convergent (i.e. that $\rho(w)$ in (8.2) satisfies the root condition). In fact, this is the case if and only if $s \le 6$, thus they cannot be used outside this range.

Method 8.6 Let us look at construction of both Adams methods and BDF formulae from a different (although equivalent) point of view, namely using the polynomial criterion for the p-order methods:

$$\sum_{m=0}^{s} a_m Q(m) = \sum_{m=0}^{s} b_m Q'(m) \qquad \forall Q \in \mathcal{P}_p\,.$$

1) For Adams methods, the right-hand side is $Q(s)-Q(s-1) = \int_{s-1}^{s} Q'(t)\,dt$, therefore letting $q=Q'$ we are looking for the quadrature formula (of numerical integration)

$$\int_{s-1}^{s} q(t)\,dt = \sum_{m=0}^{s} b_m q(m)\,,$$

which should be valid for all polynomials q of degree s (if $b_s \neq 0$) or of degree $s-1$ (if $b_s = 0$). By the Lagrange interpolating formula, we can write q as $q(t) = \sum_{m=0}^{s} \ell_m(t) q(m)$, where ℓ_m are the fundamental Lagrange polynomials satisfying $\ell_m(k) = \delta_{mk}$, and it follows that

$$b_m = \int_{s-1}^{s} \ell_m(t)\,dt\,, \qquad \ell_m(t) = \frac{\omega(t)}{(t-m)\omega'(m)}, \qquad \omega(t) = \prod_{m=0}^{s}(t-m)\,.$$

2) For BDF, we are looking for the formulae of numerical (backward) differentiation instead,

$$\sum_{m=0}^{s} a_m Q(m) = Q'(s)\,,$$

which should be valid for all polynomials of degree s. Again, from the Lagrange interpolating formula $Q(t) = \sum_{m=0}^{s} \ell_m(t) Q(m)$, it follows that $a_m = \ell_m'(s)$, and using explicit form of ℓ_m given above we derive

$$a_s = \sum_{m=1}^{s} \frac{1}{m}, \qquad a_m = \frac{(-1)^{s-m}}{s-m}\binom{s}{m}.$$

8.2 Higher order and convergence (non-examinable)

Lemma 8.7 *There is no s-step methods of order $2s+1$. The implicit and explicit methods of order $2s$ and $2s-1$, respectively, do exist and are unique.*

Example 8.8 For $s=1$ and $s=2$ the highest order (implicit) methods have the form

1) one-step order 2: $y_{n+1}-y_n = h\left(\frac{1}{2}f_{n+1}+\frac{1}{2}f_n\right)$ [trapezoidal rule],

2) two-step order 4: $y_{n+2}-y_n = h\left(\frac{1}{3}f_{n+2}+\frac{4}{3}f_{n+1}+\frac{1}{3}f_n\right)$ [Simpson rule].

Obviously, the root condition for both methods is fulfilled. However, for $s \ge 3$, the s-step methods of order $2s$ do not satisfy the root condition, and that makes them not very useful for numerical implementation. In fact, we should not go very far in our search for the higher order multistep methods because of the following restriction.

Theorem 8.9 (The first Dalquist barrier) *If an s-step method converges, then it has the order*

$$p \le \begin{cases} s & \text{for explicit methods,} \\ s+1 & \text{if } s \text{ is odd,} \\ s+2 & \text{if } s \text{ is even.} \end{cases}$$

So, in Example 8.2, the Adams-Bashforth methods are optimal in the sense that they attain the bound $p=s$ for explicit methods, and the Adams-Moulton methods for odd s are also optimal, with $p=s+1$.

An optimal implicit method with even number of steps s with order $p=s+2$ is given for example by $\rho(w) = w^s - 1$. This method corresponds to the quadrature (where $q=Q'$, with $Q \in \mathcal{P}_{s+2}$)

$$\int_0^s q(t)dt = \sum_{m=0}^{s} b_m q(m) \qquad \forall q \in \mathcal{P}_{s+1}\,.$$

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 9

9 Runge-Kutta methods

Revision 9.1 (Qudrature formulae) We may approximate the integral by the quadrature

$$\int_0^1 f(t)\,dt \approx \sum_{i=1}^{s} b_i f(c_i), \tag{9.1}$$

where, for given knots c_i, the weights b_i are chosen to make this quadrature exact for all polynomials of degree $s-1$, that is $b_i = \int_0^1 \ell_i(t)\,dt$, where ℓ_i are fundamental Lagrange polynomials, i.e., $\ell_i(c_j) = \delta_{ij}$. If $\omega(t) = \prod_{i=1}^{s}(t-c_i)$ is orthogonal to all polynomials of degree $s-1$ on $[0,1]$, then we obtain the Gaussian quadrature: the formula that is exact for all polynomials of degree $2s-1$.

Method 9.2 (Explicit Runge-Kutta methods) Suppose that we wish to solve the ODE $y' = f(t)$, with $y(0) = y_0$. The exact solution is $y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t)\,dt$ and we can approximate the integral by the quadrature (9.1) scaling it to the interval $[t_n, t_{n+1}]$. In that way, we obtain the time-stepping scheme

$$y_{n+1} = y_n + h\textstyle\sum_{i=1}^{s} b_i f(t_n + c_i h)\,, \qquad h = t_{n+1} - t_n\,.$$

Can we generalize this to genuine ODEs of the form $\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y})$? Formally, $\boldsymbol{y}(t_{n+1}) = \boldsymbol{y}(t_n) + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{y}(t))\,dt$, and this can be approximated as before by

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\textstyle\sum_{i=1}^{s} b_i \boldsymbol{f}(t_n + c_i h, \boldsymbol{y}(t_n + c_i h))\,, \tag{9.2}$$

except that, of course, the vectors $\boldsymbol{y}(t_n + c_i h)$ are unknown. But they can be approximated by another quadrature using approximate values of $\boldsymbol{y}(t_n + c_j h)$ obtained earlier:

$$\boldsymbol{y}(t_n + c_i h) = \boldsymbol{y}(t_n) + \textstyle\int_{t_n}^{t_n + c_i h} \boldsymbol{f}(t, \boldsymbol{y}(t))\,dt \;\approx\; \boldsymbol{y}(t_n) + h\sum_{j=1}^{i-1} a_{ij} \boldsymbol{f}(t_n + c_j h, \boldsymbol{y}(t_n + c_j h)),$$

so that applying $\boldsymbol{f}$ to both sides of this formula we obtain

$$\boldsymbol{f}(t_n + c_i h, \boldsymbol{y}(t_n + c_i h) \approx \boldsymbol{f}\Big(t_n + c_i h, \boldsymbol{y}(t_n) + h\textstyle\sum_{j=1}^{i-1} a_{ij} \boldsymbol{f}(t_n + c_j h, \boldsymbol{y}(t_n + c_j h))\Big)\,. \tag{9.3}$$

Finally, from (9.2)-(9.3), letting $\boldsymbol{k}_i \approx \boldsymbol{f}(t_n + c_i h, \boldsymbol{y}(t_n + c_i h))$, we arrive at the general form of an *s-stage explicit Runge–Kutta method* (RK):

$$\begin{aligned} \boldsymbol{k}_i &= \boldsymbol{f}\big(t_n + c_i h, \boldsymbol{y}_n + h\textstyle\sum_{j=1}^{i-1} a_{ij} \boldsymbol{k}_j\big), \qquad i = 1..s, \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + h\textstyle\sum_{i=1}^{s} b_i \boldsymbol{k}_i. \end{aligned}$$

Here are the same formulas with more details:

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}(t_n, \boldsymbol{y}_n), && c_1 = 0, \\ \boldsymbol{k}_2 &= \boldsymbol{f}(t_n + c_2 h, \boldsymbol{y}_n + h a_{21} \boldsymbol{k}_1), && c_2 = a_{21}, \\ \boldsymbol{k}_3 &= \boldsymbol{f}(t_n + c_3 h, \boldsymbol{y}_n + h(a_{31}\boldsymbol{k}_1 + a_{32}\boldsymbol{k}_2)), && c_3 = a_{31} + a_{32}, \\ &\vdots && \vdots \\ \boldsymbol{k}_s &= \boldsymbol{f}\big(t_n + c_s h, \boldsymbol{y}_n + h\textstyle\sum_{j=1}^{s-1} a_{sj} \boldsymbol{k}_j\big), && c_s = \sum_{j=1}^{s-1} a_{sj}, \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + h\textstyle\sum_{i=1}^{s} b_i \boldsymbol{k}_i\,, && \textstyle\sum b_i = 1\,. \end{aligned}$$

Here, equalities $c_i = \sum_{j=1}^{i-1} a_{i,j}$ (as well as $\sum b_i = 1$) simply say that quadratures are exact on constants: a reasonable (although not necessary for $a_{i,j}$) requirement. A more sophisticated choice of the RK coefficients a_{ij} is motivated at the first instance by order considerations.

9.1 Order of Runge-Kutta methods

Example 9.3 (Two-stage methods of order 2) Let us analyse the order of two-stage methods

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}(t_n, \boldsymbol{y}_n), \\ \boldsymbol{k}_2 &= \boldsymbol{f}(t_n + c_2 h, \boldsymbol{y}_n + c_2 h \boldsymbol{k}_1), \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + h\left(b_1 \boldsymbol{k}_1 + b_2 \boldsymbol{k}_2\right). \end{aligned}$$

Taylor-expanding $\boldsymbol{k}_2$ about $(t_n, \boldsymbol{y}_n)$, and using $\boldsymbol{k}_1 = \boldsymbol{f}$, we obtain

$$\boldsymbol{k}_2 = \boldsymbol{f}(t_n + c_2 h, \boldsymbol{y}_n + c_2 h \boldsymbol{f}) = \boldsymbol{f} + c_2 h \left[\boldsymbol{f}_t + \boldsymbol{f}_y \boldsymbol{f}\right] + \mathcal{O}(h^2).$$

But $\boldsymbol{y}'(t) = \boldsymbol{f}(t, \boldsymbol{y})$, hence $\boldsymbol{y}'' = \boldsymbol{f}_t + \boldsymbol{f}_y \boldsymbol{y}' = \boldsymbol{f}_t + \boldsymbol{f}_y \boldsymbol{f}$, therefore, substituting the exact solution $\boldsymbol{y}_n = \boldsymbol{y}(t_n)$ into the scheme, we obtain $\boldsymbol{k}_1 = \boldsymbol{y}'$ and $\boldsymbol{k}_2 = \boldsymbol{y}' + c_2 h \boldsymbol{y}'' + \mathcal{O}(h^2)$. Consequently, the RK method produces

$$\boldsymbol{y}(t_{n+1}) = \boldsymbol{y} + (b_1 + b_2) h \boldsymbol{y}' + b_2 c_2 h^2 \boldsymbol{y}'' + \mathcal{O}(h^3),$$

and we deduce that it is of order 2 if

$$b_1 + b_2 = 1, \qquad b_2 c_2 = \tfrac{1}{2}.$$

It is easy to demonstrate that no such method may be of order ≥ 3 (e.g. by applying it to $y' = y$).

Method 9.4 (General Runge-Kutta method) A general s-stage *Runge–Kutta method* is

$$\begin{aligned} \boldsymbol{k}_i &= \boldsymbol{f}\big(t_n + c_i h, \boldsymbol{y}_n + h \textstyle\sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j\big), \qquad i = 1..s, \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + h \textstyle\sum_{i=1}^{s} b_i \boldsymbol{k}_i. \end{aligned}$$

The explicit RK method corresponds to the case when $a_{ij} = 0$ for $i \leq j$. Otherwise, an RK method is *implicit*.

Example 9.5 Consider the 2-stage implicit method

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}\Big(t_n, \boldsymbol{y}_n + \tfrac{1}{4} h (\boldsymbol{k}_1 - \boldsymbol{k}_2)\Big), \\ \boldsymbol{k}_2 &= \boldsymbol{f}\Big(t_n + \tfrac{2}{3} h, \boldsymbol{y}_n + \tfrac{1}{12} h (3\boldsymbol{k}_1 + 5\boldsymbol{k}_2)\Big), \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + \tfrac{1}{4} h (\boldsymbol{k}_1 + 3\boldsymbol{k}_2). \end{aligned}$$

We analyse its order with respect to scalar *autonomous* equations of the form $y' = f(y)$. For brevity, we use the convention that all functions are evaluated at $y = y(t_n)$. Thus,

$$\begin{aligned} k_1 &= f + \tfrac{1}{4} h f_y (k_1 - k_2) + \tfrac{1}{32} h^2 f_{yy} (k_1 - k_2)^2 + \mathcal{O}(h^3), \\ k_2 &= f + \tfrac{1}{12} h f_y (3k_1 + 5k_2) + \tfrac{1}{288} h^2 f_{yy} (3k_1 + 5k_2)^2 + \mathcal{O}(h^3). \end{aligned}$$

Substitution $k_1 = f + \mathcal{O}(h)$, $k_2 = f + \mathcal{O}(h)$ in the above equations (h-terms) yields

$$\begin{aligned} k_1 &= f + \mathcal{O}(h^2), \\ k_2 &= f + \tfrac{2}{3} h f_y f + \mathcal{O}(h^2). \end{aligned}$$

Substituting again, we obtain

$$\begin{aligned} k_1 &= f - \tfrac{1}{6} h^2 f_y^2 f + \mathcal{O}(h^3), \\ k_2 &= f + \tfrac{2}{3} h f_y f + h^2 \left(\tfrac{5}{18} f_y^2 f + \tfrac{2}{9} f_{yy} f^2\right) + \mathcal{O}(h^3), \\ y(t_{n+1}) &= y + hf + \tfrac{1}{2} h^2 f_y f + \tfrac{1}{6} h^3 (f_y^2 f + f_{yy} f^2) + \mathcal{O}(h^4). \end{aligned}$$

But $y' = f$ implies $y'' = f_y f$ and $y''' = f_y^2 f + f_{yy} f^2$ and we deduce from Taylor's theorem that the method is at least of order 3. (It is easy to verify that it isn't of order 4, for example applying it to the equation $y' = y$.)

Theorem 9.6 (High-order RK methods) *Let $\omega(t) = \prod_{i=1}^{s} (t - c_i)$ be orthogonal on the interval $[0, 1]$ to all polynomials of degree $r - 1 \leq s - 1$, let ℓ_i be the fundamental Lagrange polynomials of degree $s - 1$ for the knots c_i, and let*

$$a_{ij} = \textstyle\int_0^{c_i} \ell_j(t)\, dt, \qquad b_i = \int_0^1 \ell_j(t)\, dt \tag{9.4}$$

Then the (highly implicit) s-stage RK method with parameters (9.4) is of order $s + r$.

Thus, the highest order of an s-stage implicit RK method is $2s$, it corresponds to collocation at zeros of Legendre polynomial (Gauss-Legendre RK method). Construction of explicit methods with high order is a kind of art.

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 10

10 Stiff ODEs: linear stability and A-stability

Consider the linear scalar ODE

$$y' = \lambda y, \qquad y(0) = 1\,, \quad \text{where} \quad \lambda < 0.$$

The exact solution is $y(t) = e^{\lambda t}$ which decays to zero as $t \to \infty$, so we would like from a numerical method to maintain the same property for the sequence (y_n). However, if we solve this ODE with forward Euler's method, then $y_{n+1} = y_n + \lambda h y_n = (1 + \lambda h)^{n+1}$, and in order to have $y_n \to 0$, we should require

$$y_n \to 0 \quad \Leftrightarrow \quad |1 + \lambda h| < 1 \quad \Leftrightarrow \quad h < \frac{2}{|\lambda|}.$$

For large $\lambda < 0$, say $\lambda = -100$, this could be a severe restriction on h.

It is important to realise that this restriction, necessary to recovery of correct asymptotic behaviour, has *nothing* to do with local accuracy, since, for large n, the genuine solution $y(t_n)$ is exceedingly small. So, this restriction is solely to prevent an unbounded growth in the numerical solution.

Definition 10.1 (Stiff ODEs) We say that the ODE $\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y})$ is *stiff* if (for some methods) we need to depress h to maintain *stability* well beyond requirements of accuracy.

Definition 10.2 (Linear stability domain) Suppose that a numerical method, applied to $y' = \lambda y$, $y(0) = 1$, with constant h, produces the solution sequence $\{y_n\}$. We call the set

$$\mathcal{D} = \{z := \lambda h \in \mathbb{C} : \lim_{n\to\infty} y_n = 0\}$$

the *linear stability domain* of the method.

Noting that the set of $\lambda \in \mathbb{C}$ for which $y(t) = e^{\lambda t} \to 0$ as $t \to \infty$ is the left half-plane $\mathbb{C}^- = \{z \in \mathbb{C} : \operatorname{Re} z < 0\}$, we say that the method is *A-stable* if $\mathbb{C}^- \subseteq \mathcal{D}$.

Example 10.3 1) For Euler's method, as we have already seen $y_n \to 0$ iff $|1 + \lambda h| < 1$, therefore its linear stability domain is $\mathcal{D} = \{z \in \mathbb{C} : |1 + z| < 1\}$.

2) Solving $y' = \lambda y$ with the *trapezoidal rule*: $y_{n+1} = y_n + h(\frac{1}{2} f_n + \frac{1}{2} f_{n+1})$, we obtain $y_{n+1} = [(1 + \frac{1}{2}\lambda h)/(1 - \frac{1}{2}\lambda h)]\, y_n$ thus, by induction, $y_n = [(1 + \frac{1}{2}\lambda h)/(1 - \frac{1}{2}\lambda h)]^n y_0$. Therefore

$$z \in \mathcal{D} \qquad \Leftrightarrow \qquad \left|\frac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}\right| < 1 \qquad \Leftrightarrow \qquad \operatorname{Re} z < 0,$$

and we deduce that $\mathcal{D} = \mathbb{C}^-$. Hence, the method is A-stable.

3) For *backward Euler*: $y_{n+1} = y_n + \lambda h y_{n+1}$, we get $y_{n+1} = (1 - \lambda h)^{-1} y_n$, whence $\mathcal{D} = \{z \in \mathbb{C} : |1 - z| > 1\}$, and the method is also A-stable.

Note that A-stability does not mean that *any* step size will do. We need to choose h small enough to ensure the right accuracy, but we don't want to depress it much further to prevent instability.

10.1 Stability of multistep methods

By definition, the number $z = \lambda h$ is in the linear stability domain $\mathcal{D}$ of a multistep method

$$\sum_{m=0}^{s} a_m y_{n+m} = h \sum_{m=0}^{s} b_m f_{n+m} \tag{10.1}$$

if the sequence (y_n) which is a solution to the recurrence relation arising from (10.1) with $f_n = \lambda y_n$ satisfies $y_n \to 0$. The latter is true if and only if the roots of the (characteristic) equation

$$p(x) := \rho(x) - z\sigma(x) = \sum_{m=0}^{s} a_m x^m - z \sum_{m=0}^{s} b_m x^m = 0 \tag{10.2}$$

are less than one in absolute values.

If $z = \lambda h$ is on the boundary of the linear stability domain $\mathcal{D}$, then the characteristic polynomial p has a root x of modulus 1, i.e. $x = e^{it}$. Substituting such an x into (10.2), we get that the boundary of $\mathcal{D}$ is the curve $z(t) = \frac{\rho(e^{it})}{\sigma(e^{it})}$.

If we need to determine the real stability interval $I = \mathcal{D} \cap \mathbb{R}$, then it is highly likely that $z_1 = \frac{\rho(1)}{\sigma(1)}$ and $z_2 = \frac{\rho(-1)}{\sigma(-1)}$ are the end-points of I. Further, since $\rho(1) = 0$, this interval for many methods is $I = (\frac{\rho(-1)}{\sigma(-1)}, 0)$. Still some further work is needed to prove that this is indeed the case.

Example 10.4 Consider the 3-step 3-rd-order Adams-Bashforth method

$$y_{n+3} - y_{n+2} = h\Big(\tfrac{23}{12} f_{n+2} - \tfrac{4}{3} f_{n+1} + \tfrac{5}{12} f_n\Big).$$

With $f = \lambda y$, the characteristic polynomial is

$$p(x) = \Big(x^3 - x^2\Big) - z\Big(\tfrac{23}{12}x^2 - \tfrac{4}{3}x + \tfrac{5}{12}\Big).$$

We have then

$$p(1) = -z, \qquad p(-1) = -2 - \tfrac{11}{3}\, z,$$

so we guess that $(-\frac{6}{11}, 0)$ is the stability interval $I = \mathcal{D} \cap \mathbb{R}$.

1) If $z > 0$, then

$$p(1) = -z < 0, \qquad p(+\infty) = +\infty,$$

hence p has a root $x_* > 1$, thus no stability.

2) If $z < -\frac{6}{11}$, then

$$p(-1) = -2 - \frac{11}{3}\, z > 0, \qquad p(-\infty) = -\infty,$$

hence p has a root $x_* < -1$, and there is no stability either.

3) For $z \in (-\frac{6}{11}, 0)$, we have

$$p(-1) = -2 - \tfrac{11}{3} z < 0, \qquad p(0) = -\tfrac{5}{12} z > 0, \qquad p(\tfrac{1}{2}) = -\tfrac{1}{8} - \tfrac{11}{48} z < 0, \qquad p(1) = -z > 0,$$

hence all three roots of p are real and reside within $(-1, 1)$, therefore the method is stable for $z = \lambda h \in (-\frac{6}{11}, 0)$.

Theorem 10.5 (The second Dahlquist barrier) *No multistep method of order $p \geq 3$ may be A-stable. (The $p = 2$ barrier for A-stability is attained by the trapezoidal rule.)*

However, some high-order multistep methods are still of good use. Stability properties of BDF, say, are satisfactory for most stiff equations. The point is that, in many stiff linear systems in applications, the eigenvalues are not just in $\mathbb{C}^-$ but also well away from $i\mathbb{R}$. All BDF methods of order $p \leq 6$ (i.e., all convergent BDF methods) share the feature that their linear stability domain $\mathcal{D}$ includes a wedge about $(-\infty, 0)$: such methods are said to be *A_0-stable.*

10.2 Stability of Runge-Kutta methods

Lemma 10.6 *No explicit Runge-Kutta method may be A-stable (or even A_0-stable).*

Proof. See Exercise 6 on Example Sheet 2. □

Unlike multistep or explicit Runge-Kutta methods, implicit high-order RK may be A-stable. For an s-stage method, its linear stability domain $\mathcal{D}$ is determined from the relation

$$y_{n+1} = r(z)y_n, \qquad r(z) = \frac{p(z)}{q(z)}, \quad p, q \in \mathcal{P}_s\,,$$

where $r(z)$ is a rational function of degree s, and the inequality $|r(z)| < 1$ for $z \in \mathbb{C}^-$ is possible.

Example 10.7 Consider the 2-stage 3rd-order method from Example 9.5

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}\Big(t_n, \boldsymbol{y}_n + \tfrac{1}{4}h(\boldsymbol{k}_1 - \boldsymbol{k}_2)\Big), \\ \boldsymbol{k}_2 &= \boldsymbol{f}\Big(t_n + \tfrac{2}{3}h, \boldsymbol{y}_n + \tfrac{1}{12}h(3\boldsymbol{k}_1 + 5\boldsymbol{k}_2)\Big), \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + \tfrac{1}{4}h(\boldsymbol{k}_1 + 3\boldsymbol{k}_2). \end{aligned}$$

Applying it to $y' = \lambda y$, we have

$$\begin{aligned} hk_1 &= h\lambda\Big(y_n + \tfrac{1}{4}hk_1 - \tfrac{1}{4}hk_2\Big), \\ hk_2 &= h\lambda\Big(y_n + \tfrac{1}{4}hk_1 + \tfrac{5}{12}hk_2\Big). \end{aligned}$$

This is a linear system, whose solution (with $z = \lambda h$) is

$$\begin{bmatrix} hk_1 \\ hk_2 \end{bmatrix} = \begin{bmatrix} 1 - \frac{1}{4}z & \frac{1}{4}z \\ -\frac{1}{4}z & 1 - \frac{5}{12}z \end{bmatrix}^{-1} \begin{bmatrix} zy_n \\ zy_n \end{bmatrix} = \frac{1}{\det(A)} \begin{bmatrix} 1 - \frac{5}{12}z & -\frac{1}{4}z \\ \frac{1}{4}z & 1 - \frac{1}{4}z \end{bmatrix} \begin{bmatrix} zy_n \\ zy_n \end{bmatrix}.$$

We need $\frac{1}{4}(hk_1 + 3hk_2) = \frac{z(1-\frac{1}{6}z)}{\det(A)}\, y_n$, therefore

$$y_{n+1} = y_n + \tfrac{1}{4}h(k_1 + 3k_2) = \Big(1 + \frac{z(1 - \frac{1}{6}z)}{1 - \frac{2}{3}z + \frac{1}{6}z^2}\Big)\, y_n = \frac{1 + \frac{1}{3}z}{1 - \frac{2}{3}z + \frac{1}{6}z^2}\, y_n.$$

Let

$$r(z) = \frac{1 + \frac{1}{3}z}{1 - \frac{2}{3}z + \frac{1}{6}z^2}\,.$$

Then $y_{n+1} = r(z)y_n$, therefore, by induction, $y_n = r(z)^n y_0$ and we deduce that

$$\mathcal{D} = \{z \in \mathbb{C} \,:\, |r(z)| < 1\}\,.$$

We wish to prove that $|r(z)| < 1$ for every $z \in \mathbb{C}^-$, since this is equivalent to A-stability. This will be done by a technique that can be applied to other RK methods. According to the *maximum modulus principle* from Complex Methods, if g is analytic in a complex domain Ω then $|g|$ attains its maximum on $\partial\Omega$. We let $g = r$. This is a rational function, hence its only singularities are the poles $2 \pm i\sqrt{2}$ and g is analytic in $\Omega = \mathbb{C}^- = \{z \in \mathbb{C} \,:\, \mathrm{Re}\, z < 0\}$. Therefore, it attains its maximum modulus either at $z = \infty$ or on $\partial\Omega = i\mathbb{R}$. Clearly, $r(z) \to 0$ as $z \to \infty$, and

$$\text{A-stability} \quad \Leftrightarrow \quad |r(z)| < 1, \quad z \in \mathbb{C}^- \quad \Leftrightarrow \quad |r(it)| \le 1, \quad t \in \mathbb{R}.$$

In turn,

$$|r(it)|^2 \le 1 \quad \Leftrightarrow \quad \big|1 - \tfrac{2}{3}it - \tfrac{1}{6}t^2\big|^2 - \big|1 + \tfrac{1}{3}it\big|^2 \ge 0.$$

But the last expression is $(1 - \frac{1}{6}t^2)^2 + \frac{4}{9}t^2 - (1 + \frac{1}{9}t^2) = \frac{1}{36}t^4 \ge 0$, and it follows that the method is A-stable.

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 11

11 Numerical implementation

Problem 11.1 The step size h is not some preordained quantity: it is a parameter of the method (in reality, many parameters, since we may vary it from step to step). The basic input of a well-written computer package for ODEs is not the step size but the *error tolerance:* the level of precision, as required by the user. The choice of $h > 0$ is an important tool at our disposal to keep a local estimate of the error beneath the required tolerance in the solution interval. In other words, we need not just a *time-stepping algorithm,* but also mechanisms for *error control* and for amending the step size.

Technique 11.2 (The Milne device) Suppose that we wish to monitor the error of the trapezoidal rule (TR),

$$y_{n+1} = y_n + h\left[\tfrac{1}{2}f(t_n, y_n) + \tfrac{1}{2}f(t_n, y_{n+1})\right].$$

We already know that the order is 2, more precisely (see Remark 6.9) the truncation error satisfies

$$y(t_{n+1}) - y^{\mathrm{TR}}_{n+1} = c_{\mathrm{TR}}h^3y'''(t_n) + \mathcal{O}(h^4), \qquad c_{\mathrm{TR}} = -\frac{1}{12}, \tag{11.1}$$

where the number $c_{\mathrm{TR}} = -\frac{1}{12}$ is called the *error constant* of TR. Unfortunately, this error estimate does not help much because the value $y'''(t_n)$ is unknown. However, each multistep method (but not RK!) has its own error constant. For example, the 2-nd order 2-step Adams–Bashforth method (AB)

$$y_{n+1} - y_n = h\left[\tfrac{3}{2}f(t_n, y_n) - \tfrac{1}{2}f(t_{n-1}, y_{n-1})\right],$$

has the error constant $c_{\mathrm{AB}} = \frac{5}{12}$ (see Example 7.5), i.e.,

$$y(t_{n+1}) - y^{\mathrm{AB}}_{n+1} = c_{\mathrm{AB}}h^3y'''(t_n) + \mathcal{O}(h^4), \quad c_{\mathrm{AB}} = \frac{5}{12}. \tag{11.2}$$

The idea behind the *Milne device* is to use two multistep methods of the same order, one explicit and the second implicit (e.g., two methods just mentioned), to estimate the local error of the implicit method. Subtracting (11.2) from (11.1), we obtain the estimate $h^3y'''(t_n) \approx -\frac{y^{\mathrm{AB}}_{n+1} - y^{\mathrm{TR}}_{n+1}}{c_{\mathrm{AB}} - c_{\mathrm{TR}}}$, therefore

$$y(t_{n+1}) - y^{\mathrm{TR}}_{n+1} \approx -\frac{c_{\mathrm{TR}}}{c_{\mathrm{AB}} - c_{\mathrm{TR}}}\left(y^{\mathrm{AB}}_{n+1} - y^{\mathrm{TR}}_{n+1}\right) = \frac{1}{6}\left(y^{\mathrm{AB}}_{n+1} - y^{\mathrm{TR}}_{n+1}\right), \tag{11.3}$$

and we use the right hand side as an estimate of the local error.

Note that TR is a far better method than AB: it is A-stable, hence its *global* behaviour is superior. We employ AB *solely* to estimate the local error (well, and for some other things). This adds very little to the overall cost of TR, since AB is an explicit method.

Implementation 11.3 To implement the *Milne device,* we work with a *pair* of multistep methods of the same order, one explicit *(predictor)* and the other implicit *(corrector)*, e.g.

$$\begin{aligned} &\text{Predictor:} && y^{\mathrm{P}}_{n+2} = y_{n+1} + h\left[\tfrac{5}{12}f_{n-1} - \tfrac{4}{3}f_n + \tfrac{23}{12}f_{n+1}\right], \\ &\text{Corrector:} && y^{\mathrm{C}}_{n+2} = y_{n+1} + h\left[-\tfrac{1}{12}f_n + \tfrac{2}{3}f_{n+1} + \tfrac{5}{12}f_{n+2}\right], \end{aligned} \tag{11.4}$$

the third-order Adams–Bashforth and Adams–Moulton methods respectively. Depending on whether the error tolerance ϵ has been achieved (from the estimate similar to (11.3)) we amend the step size h (this can be done with polynomial interpolation).

The predictor is employed not just to estimate the error of the corrector, but also to provide *an initial guess in the solution of the implicit corrector equations*, which then can be solved, say, by simple (direct) iteration (see p. 20). In the latter case, the formulas are as follows

$$y_{n+2}^{(1)} = y_{n+1} + h\Big[\tfrac{5}{12}f_{n-1} - \tfrac{4}{3}f_n + \tfrac{23}{12}f_{n+1}\Big], \tag{11.5}$$

$$y_{n+2}^{(\ell+1)} = y_{n+1} + h\Big[-\tfrac{1}{12}f_n + \tfrac{2}{3}f_{n+1} + \tfrac{5}{12}f(t_{n+2}, y_{n+2}^{(\ell)})\Big], \quad \ell = 1, 2, \ldots \tag{11.6}$$

Usually, one stops the iteration when $\|y_{n+2}^{(\ell+1)} - y_{n+2}^{(\ell)}\| < \epsilon$ is achieved.

Lemma 11.4 (A condition for convergence) *Given an implicit multistep method, say (11.4), let a simple iteration method (11.5)-(11.6) be applied to determine y_{n+2}. Further, let f satisfy a Lipschitz condition with a constant L, and let h satisfy $hL|b_2| < 1$. Then the sequence $(y_{n+2}^{(\ell)})$ converges to a solution y_{n+2} of (11.4).*

Proof. The simple iteration provides the relation

$$\begin{aligned}\|y_{n+2}^{(\ell+2)} - y_{n+2}^{(\ell+1)}\| &= h|b_2|\|f(t_{n+2}, y_{n+2}^{(\ell+1)}) - f(t_{n+2}, y_{n+2}^{(\ell)})\| \le hL|b_2|\|y_{n+2}^{(\ell+1)} - y_{n+2}^{(\ell)}\| \\ &\le (hL|b_2|)^{\ell}\|y_{n+2}^{(1)} - y_{n+2}^{(0)}\|.\end{aligned}$$

Thus, $hL|b_2| < 1$ implies that $(y_{n+2}^{(\ell)})$ is a Cauchy sequence, so it converges to a limit, y_{n+2} say. Further, $y_{n+2}^{(\ell+1)} \to y_{n+2}$ and $y_{n+2}^{(\ell)} \to y_{n+2}$ in (11.6) show that y_{n+2} satisfies (11.4). □

Technique 11.5 (Embedded Runge–Kutta methods) The situation is more complicated with RK, since no single error constant determines local growth of the error. The approach of *embedded RK* requires, again, two (typically explicit) methods: an RK method of s stages and order p, say, and another method, of $s + m$ stages, $m \ge 1$, and order $p + 1$, such that *the first s stages of both methods are identical.* (This means that the cost of implementing the higher-order method is marginal, once we have computed the lower-order approximation.) For example, consider

$$\left.\begin{aligned} k_1 &= f(t_n, y_n), \\ k_2 &= f\big(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}hk_1\big), \\ y_{n+1}^{[1]} &= y_n + hk_2; \end{aligned}\right\} \text{order 2}$$

$$\left.\begin{aligned} k_3 &= f(t_n + h, y_n - hk_1 + 2hk_2), \\ y_{n+1}^{[2]} &= y_n + \tfrac{1}{6}h(k_1 + 4k_2 + k_3)\,. \end{aligned}\right\} \text{order 3}$$

We thus estimate $y(t_{n+1}) - y_{n+1}^{[1]} \approx y_{n+1}^{[2]} - y_{n+1}^{[1]}$. (It might look paradoxical, at least at first glance, but the only purpose of the higher-order method is to provide error control for the lower-order one!)

Technique 11.6 (The Zadunaisky device) Suppose that the ODE $y' = f(t, y)$, $y(0) = y_0$, is solved by an arbitrary numerical method of order p and that we have stored (not necessarily equidistant) past solution values $y_n, y_{n-1}, \ldots, y_{n-p}$. We form an interpolating pth degree polynomial (with vector coefficients) q such that $q(t_{n-i}) = y_{n-i}$, $i = 0, 1, \ldots, p$, and consider the differential equation

$$z' = f(t, z) + q'(t) - f(t, q), \qquad z(t_n) = y_n. \tag{11.7}$$

There are two important observations with regard to (11.7)

(1) Since $q(t) - y(t) = \mathcal{O}(h^{p+1})$, the term $q'(t) - f(t, q)$ is usually small (because $y'(t) - f(t, y(t)) \equiv 0$). Therefore, (11.7) is a small perturbation of the original ODE.

(2) The exact solution of (11.7) is known: $z(t) = q(t)$. Now, having produced y_{n+1} with our numerical method, we proceed to evaluate z_{n+1} as well, *using exactly the same method and implementation details.* We then evaluate the error in z_{n+1}, namely $z_{n+1} - q(t_{n+1})$, and use it as an estimate of the error in y_{n+1}.

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 12

12 LU factorization

12.1 Introduction

Problem 12.1 The basic problem of numerical analysis is: solve

$$Ax = b. \tag{12.1}$$

The formula

$$\det A = \textstyle\sum_i (-1)^{\sigma(i)} a_{1,i_1} a_{2,i_2} \dots a_{n,i_n}$$

gives an opportunity to solve (12.1) explicitly by Cramer's rule with the small detail that it costs $n!$ multiplications. Given a computer with 10^9 flops/sec, this way of solving an $n \times n$ system will take

a) $n = 10$, $t = 10^{-4}$ sec, b) $n = 20$, $t = 17$ min, c) $n = 30$, $t = 400\,000$ years.

Thus we have to look at methods from a more constructive and practical point of view.

Example 12.2 (Triangular matrices) An *upper triangular* matrix U has $u_{ij} = 0$ if $i > j$. Then $\det U = \prod u_{ii}$. Also, we can solve $Ux = y$ by the so-called *back–substitution*

$$x_n = y_n/u_{nn}, \quad . \quad x_i = (y_i - \textstyle\sum_{j=i+1}^{n} u_{ij} x_j)/u_{ii}, \quad i = n-1, \dots, 1.$$

Similarly, a *lower triangular* matrix L (s.t. $\ell_{ij} = 0$ if $i < j$) allows us to solve $Ly = b$ directly by the *forward substitution*. Hence, if we manage to factorize A as

$$A = LU,$$

then solution of the system $Ax = b$ can be split into two cases

$$Ax = L\underbrace{Ux}_{y} = b \quad \Rightarrow \quad 1)\ Ly = b, \quad 2)\ Ux = y, \tag{12.2}$$

both being solved sufficiently easily, as we have seen.

Example 12.3 (Orthogonal matrices) An orthogonal matrix Q satisfies $Q^T Q = I$, i.e., $Q^{-1} = Q^T$, so that a solution to $Qx = y$ is simply $x = Q^T y$. If

$$A = QR$$

where Q is orthogonal and R is upper triangular, then again one can solve the system $Ax = b$ in two simple steps

$$Ax = QRx = b \quad \Rightarrow \quad 1)\ Qy = b, \quad 2)\ Rx = y.$$

12.2 LU factorization

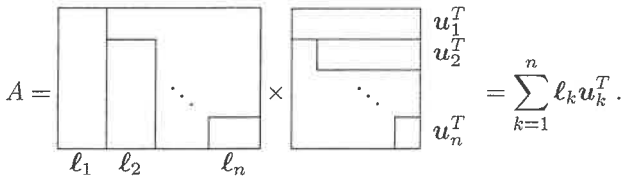
Definition 12.4 By the LU factorization of a (nonsingular) matrix A we understand a representation of A as a product

$$A = LU,$$

where L is a lower triangular matrix with unit diagonal and U is a nonsingular upper triangular matrix.

Remark 12.5 Nonsingularity of U (and A) is a natural requirement in all practical applications. Theoretically it is not necessary; see Question 1 on Examples Sheet 3.

Method 12.6 We present a method of the LU factorization that is based on the direct approach. If $A = LU$, and $\boldsymbol{\ell}_k$ and $\boldsymbol{u}_k^T$ are the columns and the rows of the matrices L and U respectively, then



$$= \sum_{k=1}^{n} \boldsymbol{\ell}_k \boldsymbol{u}_k^T .$$

Since the leading $k-1$ elements of $\boldsymbol{\ell}_k$ and $\boldsymbol{u}_k^T$ are zero for $k \geq 2$, it follows that the first $k-1$ columns and rows of the matrix $\boldsymbol{\ell}_k \boldsymbol{u}_k^T$ are zeros as well. Hence $\boldsymbol{u}_1^T$, which is the 1-st row of the matrix $\boldsymbol{\ell}_1 \boldsymbol{u}_1^T$, coincides with the 1-st row of A, while $\boldsymbol{\ell}_1 \cdot u_{11} = \boldsymbol{\ell}_1 \cdot a_{11}$ coincides with the 1-st column of A. Substracting and letting

$$A^{(1)} := A - \boldsymbol{\ell}_1 \boldsymbol{u}_1^T$$

we obtain similarly that $\boldsymbol{u}_2^T$ and $\boldsymbol{\ell}_2 \cdot a_{22}^{(1)}$ are the 2-nd row and the 2-nd column of $A^{(1)}$ respectively, so that we proceed further with

$$A^{(2)} := A^{(1)} - \boldsymbol{\ell}_2 \boldsymbol{u}_2^T ,$$

and so on.

Algorithm 12.7 Notice that we can use the matrix A itself to store the elements of L and U, in the subdiagonal and in the upper triangular portions of A, respectively. So, the previous method allows computing of the LU factorization as follows.

```
for k=1 to n-1 do
    for i=k+1 to n do
        a(i,k) := a(i,k)/a(i,i);     #--   l(i,k) := a(i,k)/a(i,i)
        for m=k+1 to n do            #--   u(k,m) := a(k,m)
            a(i,m) := a(i,m) - a(i,k)*a(k,m)
                                     #--   a(i,m) := a(i,m) - l(i,k)*u(k,m)
        end
    end
end
```

Example 12.8

$$\begin{bmatrix} 2 & -1 & -3 & 3 \\ 4 & 0 & -3 & 1 \\ 6 & 1 & -1 & 6 \\ -2 & -5 & 4 & 1 \end{bmatrix} \to \begin{bmatrix} 2 & -1 & -3 & 3 \\ \boxed{2} & 2 & 3 & -5 \\ 3 & 4 & 8 & -3 \\ -1 & -6 & 1 & 4 \end{bmatrix} \to \begin{bmatrix} 2 & -1 & -3 & 3 \\ \boxed{2} & \boxed{2} & 3 & -5 \\ 3 & 2 & 2 & 7 \\ -1 & -3 & 10 & -11 \end{bmatrix} \to \begin{bmatrix} 2 & -1 & -3 & 3 \\ \boxed{2} & \boxed{2} & 3 & -5 \\ 3 & \boxed{2} & \boxed{2} & 7 \\ -1 & -3 & 5 & -46 \end{bmatrix}$$

Application 12.9 One can use the LU factorization in the following problems.

1. Solution of linear systems (see (12.2)).
2. Calculation of the determinant: $\det A = \det L \det U = \prod u_{ii}$.
3. Calculation of the inverse of A: $A^{-1} = U^{-1} L^{-1}$.

The cost. It is only multiplications and divisions that matter. At the k-th step of the LU algorithm we perform $(n-k)$ divisions to determine the components of $\boldsymbol{\ell}_k$ and $(n-k)^2$ multiplications for finding $\boldsymbol{\ell}_k \boldsymbol{u}_k^T$. Hence

$$N_{LU} = \textstyle\sum_{k=1}^{n-1} [(n-k)^2 + (n-k)] = \frac{1}{3} n^3 + \mathcal{O}(n^2).$$

Solving $L\boldsymbol{y} = \boldsymbol{b}$ by forward substitution we use k multiplications/divisions to determine y_k, and similarly for back substitutions, thus

$$N_F = N_B \sim \tfrac{1}{2} n^2.$$

Remark 12.10 At the k-th step of the LU-algorithm the multipliers l_{ik} are chosen so that the outcome of the operation $A^{(k)} = A^{(k-1)} - \boldsymbol{\ell}_k \boldsymbol{u}_k^T$ is that the kth column of $A^{(k)}$ is zero. This is equivalent to Gaussian elimination of the unknown x_k from the system $A\boldsymbol{x} = \boldsymbol{b}$. The only yet important difference between LU and Gaussian elimination is that we do not consider the right hand side $\boldsymbol{b}$ until the factorization is complete. This is useful e.g. when there are many right hand sides, in particular if not all the $\boldsymbol{b}$'s are known at the outset (as, say, in the multigrid method). In Gaussian elimination the solution for each new $\boldsymbol{b}$ would require $\mathcal{O}(n^3)$ computational operations, whereas with LU factorization $\mathcal{O}(n^3)$ operations are required for the initial factorization, but then the solution for each new $\boldsymbol{b}$ only requires $\mathcal{O}(n^2)$ operations (for back- and forward-substitutions).

12.3 Pivoting

The LU algorithm fails if $a_{kk}^{(k-1)} = 0$. A remedy is to exchange rows of $A^{(k-1)}$ by picking a suitable *pivotal equation*, i.e., the equation which is used to eliminate one unknown from certain of the other equation.

This technique is called *column pivoting* and it means that, having obtained $A^{(k-1)}$, we exchange two rows of $A^{(k-1)}$ so that the element of largest magnitude in the k-th column is in the *pivotal position* (k, k). In other words,

$$|a_{k,k}^{(k-1)}| = \max_{j=k\ldots n} |a_{j,k}^{(k-1)}|.$$

The exchange of rows k and m can be regarded as the pre-multiplication of the relevant matrix $A^{(k-1)}$ by a permutation matrix P_{km}. Since

$$P_{km}A^{(k-1)} = P_{km}A - \sum_{i=1}^{k-1}(P_{km}\boldsymbol{\ell}_i)\boldsymbol{u}_i^T,$$

the same exchange is required in the portion of L that has been formed already (i.e., in the first $k-1$ columns). Also, we need to record the permutation of rows to solve for the right-hand side and/or to compute the determinant.

Example 12.11

$$\begin{bmatrix} 2 & 1 & 1 \\ \bullet 4 & 1 & 0 \\ -2 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 1 & 0 \\ 2 & 1 & 1 \\ -2 & 2 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & 1 \\ -\frac{1}{2} & \bullet\frac{5}{2} & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 1 & 0 \\ -\frac{1}{2} & \frac{5}{2} & 1 \\ \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 4 & 1 & 0 \\ -\frac{1}{2} & \frac{5}{2} & 1 \\ \frac{1}{2} & \frac{1}{5} & \frac{4}{5} \end{bmatrix},$$

$$PA = LU \quad \Rightarrow \quad A = P^T LU, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & & \\ -\frac{1}{2} & 1 & \\ \frac{1}{2} & \frac{1}{5} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 4 & 1 & 0 \\ & \frac{5}{2} & 1 \\ & & \frac{4}{5} \end{bmatrix}.$$

An important advantage of column pivoting is that every element of L has magnitude at most one. This avoids not just division by zero (what is not the case in the previous example) but also tends to reduce the chance of very large numbers occuring during the factorization, a phenomenon that might lead to *ill conditioning* and to accumulation of *round-off error*.

12.4 Examples

Example 12.12 (LU factorization)

$$\begin{bmatrix} -3 & 2 & 3 & -1 \\ 6 & -2 & -6 & 0 \\ -9 & 4 & 10 & 3 \\ 12 & -4 & -13 & -5 \end{bmatrix} \xrightarrow[\substack{4^{\text{th}} - 1^{\text{st}} \cdot (-4)}]{\substack{2^{\text{nd}} - 1^{\text{st}} \cdot (-2) \\ 3^{\text{rd}} - 1^{\text{st}} \cdot \ 3}} \begin{bmatrix} -3 & 2 & 3 & -1 \\ -2 & 2 & 0 & -2 \\ 3 & -2 & 1 & 6 \\ -4 & 4 & -1 & -9 \end{bmatrix}$$

$$\xrightarrow{\substack{3^{\text{rd}} - 2^{\text{nd}} \cdot (-1) \\ 4^{\text{th}} - 2^{\text{nd}} \cdot \ 2}} \begin{bmatrix} -3 & 2 & 3 & -1 \\ -2 & 2 & 0 & -2 \\ 3 & -1 & 1 & 4 \\ -4 & 2 & -1 & -5 \end{bmatrix} \xrightarrow{4^{\text{th}} - 3^{\text{rd}} \cdot (-1)} \begin{bmatrix} -3 & 2 & 3 & -1 \\ -2 & 2 & 0 & -2 \\ 3 & -1 & 1 & 4 \\ -4 & 2 & -1 & -1 \end{bmatrix}$$

i.e.

$$A = LU, \quad L = \begin{bmatrix} 1 & & & \\ -2 & 1 & & \\ 3 & -1 & 1 & \\ -4 & 2 & -1 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} -3 & 2 & 3 & -1 \\ & 2 & 0 & -2 \\ & & 1 & 4 \\ & & & -1 \end{bmatrix}$$

Example 12.13 (Forward and back substitutions) Then the solution to the system

$$A\boldsymbol{x} = \boldsymbol{b}, \qquad \boldsymbol{b} = [3, -2, 2, 0]^T$$

proceeds in two steps

$$A\boldsymbol{x} = L\underbrace{U\boldsymbol{x}}_{\boldsymbol{y}} = \boldsymbol{b} \quad \Rightarrow \quad 1) \quad L\boldsymbol{y} = \boldsymbol{b}, \quad 2) \quad U\boldsymbol{x} = \boldsymbol{y}.$$

1) Forward substitution

$$\begin{bmatrix} 1 & & & \\ -2 & 1 & & \\ 3 & -1 & 1 & \\ -4 & 2 & -1 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \\ 2 \\ 0 \end{bmatrix} \begin{matrix} \downarrow \\ \downarrow \\ \downarrow \\ \downarrow \end{matrix} \quad \Rightarrow \quad y = \begin{bmatrix} 3 \\ 4 \\ -3 \\ 1 \end{bmatrix},$$

2) Back substitution

$$\begin{bmatrix} -3 & 2 & 3 & -1 \\ & 2 & 0 & -2 \\ & & 1 & 4 \\ & & & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \\ -3 \\ 1 \end{bmatrix} \begin{matrix} \uparrow \\ \uparrow \\ \uparrow \\ \uparrow \end{matrix} \quad \Rightarrow \quad x = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix},$$

Example 12.14 (LU factorization with pivoting)

$$\begin{bmatrix} -3 & 2 & 3 & -1 \\ 6 & -2 & -6 & 0 \\ -9 & 4 & 10 & 3 \\ \bullet 12 & -4 & -13 & -5 \end{bmatrix} \underset{\rightarrow}{\begin{bmatrix} 4 \\ 2 \\ 3 \\ 1 \end{bmatrix}} \begin{bmatrix} 12 & -4 & -13 & -5 \\ 6 & -2 & -6 & 0 \\ -9 & 4 & 10 & 3 \\ -3 & 2 & 3 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 12 & -4 & -13 & -5 \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{5}{2} \\ -\frac{3}{4} & \bullet 1 & \frac{1}{4} & -\frac{3}{4} \\ -\frac{1}{4} & 1 & -\frac{1}{4} & -\frac{9}{4} \end{bmatrix}$$

$$\underset{\rightarrow}{\begin{bmatrix} 4 \\ 3 \\ 2 \\ 1 \end{bmatrix}} \begin{bmatrix} 12 & -4 & -13 & -5 \\ -\frac{3}{4} & 1 & \frac{1}{4} & -\frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{5}{2} \\ -\frac{1}{4} & 1 & -\frac{1}{4} & -\frac{9}{4} \end{bmatrix} \rightarrow \begin{bmatrix} 12 & -4 & -13 & -5 \\ -\frac{3}{4} & 1 & \frac{1}{4} & -\frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{5}{2} \\ -\frac{1}{4} & 1 & -\frac{1}{2} & -\frac{3}{2} \end{bmatrix} \rightarrow \begin{bmatrix} 12 & -4 & -13 & -5 \\ -\frac{3}{4} & 1 & \frac{1}{4} & -\frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} & \frac{5}{2} \\ -\frac{1}{4} & 1 & -1 & 1 \end{bmatrix}$$

i.e.

$$A = P^T LU, \quad P = \begin{bmatrix} & & & 1 \\ & & 1 & \\ & 1 & & \\ 1 & & & \end{bmatrix}, \quad L = \begin{bmatrix} 1 & & & \\ -\frac{3}{4} & 1 & & \\ \frac{1}{2} & 0 & 1 & \\ -\frac{1}{4} & 1 & 1 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 12 & -4 & -13 & -5 \\ & 1 & \frac{1}{4} & -\frac{3}{4} \\ & & \frac{1}{2} & \frac{5}{2} \\ & & & 1 \end{bmatrix}$$

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 13

13 LU factorization (cont.)

13.1 Existence and uniqueness of the LU factorization

Definition 13.1 A square matrix A is called *strictly regular* if all its leading submatrices $A_k = (a_{ij})_{i,j=1}^{k}$ (the matrix A itself as well) are nonsingular.

Theorem 13.2 (Existence) *A matrix A admits an LU factorization $\Leftrightarrow$ it is strictly regular.*

Proof. ($\Rightarrow$) This part follows from the scheme (where we partition matrices into blocks):

$$\left[\begin{array}{c|c} A_k & B_{k,n-k} \\ \hline C_{n-k,k} & D_{n-k} \end{array}\right] = A = LU = \left[\begin{array}{c|c} L_k & O_{k,n-k} \\ \hline X_{n-k,k} & L'_{n-k} \end{array}\right] \cdot \left[\begin{array}{c|c} U_k & Y_{k,n-k} \\ \hline O_{n-k,k} & U'_{n-k} \end{array}\right].$$

Multiplying blocks we obtain $A_k = L_k U_k$, and since L_k, U_k are nonsingular, so is A_k.

($\Leftarrow$) We use induction. For $n = 1$ the result is trivial. Assume the result for $(n-1) \times (n-1)$ matrices. Partition the $n \times n$ matrix A as

$$A = \left[\begin{array}{c|c} A_{n-1} & \boldsymbol{b} \\ \hline \boldsymbol{c}^T & a_{nn} \end{array}\right].$$

Let us show that we can get

$$A = LU \quad \text{with} \quad L = \left[\begin{array}{c|c} L_{n-1} & \boldsymbol{o} \\ \hline \boldsymbol{x}^T & 1 \end{array}\right], \quad U = \left[\begin{array}{c|c} U_{n-1} & \boldsymbol{y} \\ \hline \boldsymbol{o}^T & u_{nn} \end{array}\right],$$

where L_{n-1}, U_{n-1}, $\bar{x}^T$, $\bar{y}$ and u_{nn} are to be determined. Multiplying out these block matrices we see that we want to have

$$A = \left[\begin{array}{c|c} A_{n-1} & \boldsymbol{b} \\ \hline \boldsymbol{c}^T & a_{nn} \end{array}\right] = \left[\begin{array}{c|c} L_{n-1}U_{n-1} & L_{n-1}\boldsymbol{y} \\ \hline \boldsymbol{x}^T U_{n-1} & \boldsymbol{x}^T\boldsymbol{y} + 1 \cdot u_{nn} \end{array}\right].$$

In virtue of induction assumption, L_{n-1} and U_{n-1} exist and are non-singular. Then $L_{n-1}\boldsymbol{y} = \boldsymbol{b}$, $\boldsymbol{x}^T U_{n-1} = \boldsymbol{c}^T$ can be solved to get

$$\boldsymbol{y} = L_{n-1}^{-1}\boldsymbol{b}, \qquad \boldsymbol{x}^T = \boldsymbol{c}^T U_{n-1}^{-1}, \qquad u_{nn} = a_{nn} - \boldsymbol{x}^T\boldsymbol{y}.$$

So, there exists a factorization $A = LU$ with $\ell_{ii} = 1$, but we need also U to be nonsingular, which it is because $\det U = \det A$ and $\det A \neq 0$ by strict regularity.

Theorem 13.3 (Uniqueness) *The LU factorization is unique, i.e.*

$$A = L_1U_1 = L_2U_2 \quad \textit{implies} \quad L_1 = L_2, \quad U_1 = U_2.$$

Proof. Actually, this fact follows from Method 12.6 because at each of its step the vectors ℓ_k and u_k are uniquely determined. Here is an independent proof.

The equality $L_1U_1 = L_2U_2$ implies $L_2^{-1}L_1 = U_2U_1^{-1}(=: V$, say). The inverse of a lower triangular matrix and the product of such matrices are lower triangular matrices. The same is true for upper triangular matrices. Consequently, V is simultaneously lower and upper triangular, hence it is diagonal. Since $L_2^{-1}L_1$ has unit diagonal, we obtain $V = I$. □

Remark 13.4 Both theorems are for the LU factorization with a *nonsingular* U. If we required only $\ell_{ii} = 1$, then *some* singular matrices A may be LU factorized, and in many ways:

$$\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & \frac{1}{2} \end{bmatrix}.$$

(See Question 3 about LU factorization with arbitrary U). There is no LU factorization (whatsoever) for regular but not strictly regular matrices, e.g. for $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. On the other hand, *every* regular matrix A admits an LU factorization with pivoting, i.e. for every nonsingular matrix A there exists a permutation matrix P such that $PA = LU$.

Corollary 13.5 *A strictly regular matrix A has a unique factorization $A = LDU$ where both L and U have unit diagonal.*

Corollary 13.6 *A strictly regular symmetric matrix A admits the unique representation $A = LDL^T$.*

Proof. By strict regularity, $A = LDU$, and by symmetry,

$$LDU = A = A^T = U^T D L^T.$$

Since the LDU factorization is unique, $U = L^T$ □

13.2 Symmetric positive definite and diagonally dominant matrices

Here we consider two important classes of matrices which are strictly regular hence possess an LU factorization.

Definition 13.7 A matrix A is called (symmetric) *positive definite* (SPD-matrix) if $\boldsymbol{x}^T A \boldsymbol{x} > 0$ for all $\boldsymbol{x} \neq 0$ and $A = A^T$.

Theorem 13.8 *Let A be a real $n \times n$ symmetric matrix. It is positive definite if and only if it has the LDL^T factorization in which the diagonal elements of D are all positive.*

Proof. Suppose that $A = LDL^T$ with $D > 0$, and let $\boldsymbol{x} \in \mathbb{R}^n \setminus 0$. Since L is nonsingular, $\boldsymbol{y} := L^T \boldsymbol{x} \neq 0$. Then $\boldsymbol{x}^T A \boldsymbol{x} = \boldsymbol{y}^T D \boldsymbol{y} = \sum_{k=1}^n d_{kk} y_k^2 > 0$, hence A is positive definite.

Conversely, if A is (symmetric) positive definite, then it is strictly regular. Indeed, if $A_k \boldsymbol{x} = 0$ for some $k \leq n$ and some $\boldsymbol{x} \in \mathbb{R}^k$, then for $\boldsymbol{x}_* = (x_1, \ldots, x_k, 0, \ldots, 0) \in \mathbb{R}^n$ we obtain $\boldsymbol{x}_*^T A \boldsymbol{x}_* = \boldsymbol{x}^T A_k \boldsymbol{x} = 0$, a contradiction to the positive definiteness of A. Thus, by Corollary 13.6 it admits an LDL^T factorization. Take $\boldsymbol{x}$ such that $L^T \boldsymbol{x} = \boldsymbol{e}_k = [0..1..0]^T$. Then $0 < \boldsymbol{x}^T A \boldsymbol{x} = \boldsymbol{e}_k^T D \boldsymbol{e}_k = d_{kk}$. □

Method 13.9 One can check if a symmetric matrix is positive definite by trying to form its LDL^T factorization.

Example 13.10 The matrix below is positive definite.

$$\begin{bmatrix} 2 & 6 & -2 \\ 6 & 21 & 0 \\ -2 & 0 & 16 \end{bmatrix} \to \begin{bmatrix} 2 & 6 & -2 \\ \boxed{3} & 3 & 6 \\ -1 & 6 & 14 \end{bmatrix} \to \begin{bmatrix} 2 & 6 & -2 \\ \boxed{3} & \boxed{3} & 6 \\ -1 & \boxed{2} & 2 \end{bmatrix} \Rightarrow L = \begin{bmatrix} 1 & & \\ 3 & 1 & \\ -1 & 2 & 1 \end{bmatrix}, \quad D = \begin{bmatrix} 2 & & \\ & 3 & \\ & & 2 \end{bmatrix}.$$

The only application of this method is during exam. Normally, we know whether the matrix in question is positive definite by some a priori reasons. Say, a Gram matrix with $a_{ij} = [\boldsymbol{x}^{(i)}]^T \boldsymbol{x}^{(j)}$, where vectors $\boldsymbol{x}^{(i)}$ are linearly independent, is positive definite.

Corollary 13.11 (Cholesky[1] factorization) *A positive definite matrix A admits the Cholesky factorization*

$$A = \tilde{L} \tilde{L}^T,$$

where $\tilde{L}$ is a lower triangular matrix.

Proof. Since $A = LDL^T$ with a positive diagonal matrix D, we can write

$$A = LDL^T = (LD^{1/2})(D^{1/2}L^T) =: \widetilde{L}\widetilde{L}^T.$$

Definition 13.12 A matrix A is called *strictly diagonally dominant* (by rows) if

$$|a_{ii}| > \sum_{j\neq i} |a_{ij}| \quad \text{for all } i, \qquad \text{e.g.} \quad A = \begin{bmatrix} 6 & -2 & 1 \\ 1 & 4 & 2 \\ 2 & 1 & -5 \end{bmatrix}.$$

Theorem 13.13 *If A is strictly diagonally dominant (by rows), then it is srictly regular, hence the LU factorization exists.*

Proof. Take any $x = (x_1, ..., x_n) \neq 0$ and let x_i be its largest absolute value component, i.e., $|x_i| \geq |x_j|$. Then, for the i-th component of Ax, the strict diagonal dominance gives the value

$$|(Ax)_i| = |a_{ii}x_i + \sum_{j\neq i} a_{ij}x_j| \geq |a_{ii}|\,|x_i| - \underbrace{\sum_{j\neq i} |a_{ij}|}_{<|a_{ii}|}\underbrace{|x_j|}_{\leq |x_i|} > 0.$$

Hence $Ax \neq 0$ for any x, i.e. A is nonsingular. The leading submatrices of a strictly diagonally dominant matrix are (even more) strictly diagonally dominant, hence the strict regularity of A. □

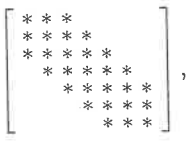
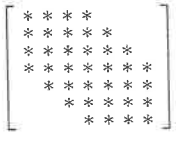
13.3 Sparse and band matrices

Definition 13.14 A matrix $A \in \mathbb{R}^n$ is called a *sparse* matrix if nearly all elements of A are zero. Most useful examples are band matrices and block band matrices

Definition 13.15 A matrix A is called a *band matrix* if there exists an integer $r < n$ such that

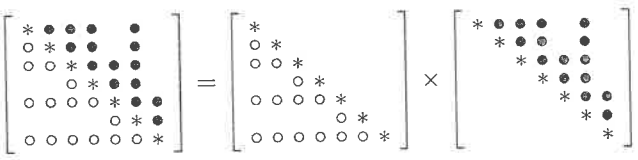
$$a_{ij} = 0 \quad \text{for all} \quad |i - j| > r.$$

In other words, all nonzero elements of A reside in a band of width $2r+1$ along the main diagonal.

$r = 1$  , $r = 2$  , $r = 3$  .

It is frequently required to solve *very* large systems $Ax = b$ with sparse matrices (say, $n = 10^5$ is considered small in this context). The LU factorization of such A makes sense only if L and U inherit much of the sparsity of A (so that the cost of computing Ux, say, is comparable with that of Ax). To this end the following theorem is useful.

Theorem 13.16 *Let $A = LU$ be the LU factorization (without pivoting) of a sparse matrix. Then*
1) *all leading zeros in the rows of A to the left of diagonal are inherited by L,*
2) *all leading zeros in the columns of A above the diagonal are inherited by U.*



[1] Andre-Lois Cholesky (1875-1918) was a geodesist in the French army's artillery, who died in battle. He developed his method to solve the problems of the "precision levelling of Algeria and Tunesia" in 1910-12. His mathematical work was posthumously published on his behalf in 1924.

Proof. Let $a_{i,1} = a_{i,2} = \cdots = 0$ be the leading zeros in the i-th row. Then, since $u_{kk} \neq 0$, we obtain

$$\begin{aligned} 0 = a_{i,1} &= \ell_{i,1}u_{11} && \Rightarrow \quad \ell_{i,1} = 0, \\ 0 = a_{i,2} &= \ell_{i,1}u_{12} + \ell_{i,2}u_{22} && \Rightarrow \quad \ell_{i,2} = 0, \\ 0 = a_{i,3} &= \ell_{i,1}u_{13} + \ell_{i,2}u_{23} + \ell_{i,3}u_{33} && \Rightarrow \quad \ell_{i,3} = 0, \quad \text{and so on.} \end{aligned}$$

Similarly for the leading zeros in the j-th column.

Corollary 13.17 *If A is a band matrix and $A = LU$, then $\ell_{ij} = u_{ij} = 0$ for all $|i - j| > r$, i.e. L and U are the band matrices with the same band width as A.*

The cost (Question 5). In the case of a banded A, we need just
1) $\mathcal{O}(nr^2)$ operations to factorize and
2) $\mathcal{O}(nr)$ operations to solve a linear system (after factorization).
This must be compared with $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ operations respectively for an $n \times n$ *dense* matrix. If $r \ll n$ this represents a very substantial saving!

Method 13.18 Theorem 13.16 suggests that for a factorization of a sparse but not nicely structured matrix A one might try to reorder its rows and columns by a preliminary calculation so that many of the zero elements become leading zero elements in rows and columns, thus reducing the fill-in in L and U.

Example 13.19 The LU factorization of

$$A = \begin{bmatrix} 5 & 1 & 1 & 1 & 1 \\ 1 & 1 & & & \\ 1 & & 1 & & \\ 1 & & & 1 & \\ 1 & & & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ \frac{1}{5} & 1 & & & \\ \frac{1}{5} & -\frac{1}{4} & 1 & & \\ \frac{1}{5} & -\frac{1}{4} & -\frac{1}{3} & 1 & \\ \frac{1}{5} & -\frac{1}{4} & -\frac{1}{3} & -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 5 & 1 & 1 & 1 & 1 \\ & \frac{4}{5} & -\frac{1}{5} & -\frac{1}{5} & -\frac{1}{5} \\ & & \frac{3}{4} & -\frac{1}{4} & -\frac{1}{4} \\ & & & \frac{2}{3} & -\frac{1}{3} \\ & & & & \frac{1}{2} \end{bmatrix}$$

has significant fill-in. However, exchanging the first and the last rows and columns yields

$$PA = \begin{bmatrix} 1 & & & & 1 \\ & 1 & & & 1 \\ & & 1 & & 1 \\ & & & 1 & 1 \\ 1 & 1 & 1 & 1 & 5 \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & 1 \\ & 1 & & & 1 \\ & & 1 & & 1 \\ & & & 1 & 1 \\ & & & & 1 \end{bmatrix}.$$

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 14

14 QR factorization

14.1 Inner product spaces

Definition 14.1 An *inner product* on a real vector space $\mathbb{X}$ is a function $(\cdot,\cdot) : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ which satisfies the following axioms

1) $(x,x) \geq 0 \quad \forall x \in \mathbb{X}, \quad$ with equality only if $x = 0$
2) $(x,y) = (y,x) \quad \forall x, y \in \mathbb{X},$
3) $(\alpha x, y) = \alpha(x,y) \quad \forall \alpha \in \mathbb{R} \quad \forall x, y \in \mathbb{X},$
4) $(x+y,z) = (x,z) + (y,z) \quad \forall x, y, z \in \mathbb{X}.$

A vector space with an inner product is called an *inner product* space. If $(x,y) = 0$, then the vectors x, y are called *orthogonal*. A set of vectors $(x_i) \in \mathbb{X}$ is called *orthonormal* if

$$(x_i, x_j) = \delta_{ij} := \begin{cases} 0, & i = j; \\ 1, & i \neq j. \end{cases}$$

For $x \in \mathbb{X}$, the function $\|x\| := (x,x)^{1/2}$ is called the *norm* of x (induced by the given inner product), and we have the Cauchy–Schwarz inequality

$$(x,y) \leq \|x\| \, \|y\| \quad \forall x, y \in \mathbb{X}.$$

Example 14.2 For $\mathbb{X} = \mathbb{R}^m$, the following rule defines the so-called Euclidean inner product

$$(\boldsymbol{u}, \boldsymbol{v}) := \boldsymbol{u}^T \boldsymbol{v} := \sum_{i=1}^{m} u_i v_i, \qquad \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^m.$$

Example 14.3 Another example, which we used in Lecture 3 in construction of orthogonal polynomials, is the inner product on the space $C[a,b]$ of real-valued continuous functions on $[a,b]$ with respect to a fixed positive *weight function* w. It is given by the rule

$$(f,g)_w := \int_a^b [f(x)g(x)]\, w(x)\, dx, \qquad f, g \in C[a,b].$$

14.2 Orthogonal matrices

Definition 14.4 An $m \times n$ matrix Q $(m \geq n)$ is called *orthogonal* if $Q^T Q = I$, e.g., $Q = \frac{1}{3}\begin{bmatrix} 2 & 2 & 1 \\ 1 & -2 & 2 \\ 2 & -1 & -2 \end{bmatrix}$.

Thus, the columns $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n \in \mathbb{R}^m$ of the orthogonal Q satisfy $\boldsymbol{q}_i^T \boldsymbol{q}_j = \delta_{ij}$, i.e., they are *orthonormal* with respect to the Euclidean inner product in $\mathbb{R}^m$. For a square Q we get

$$Q^{-1} = Q^T, \quad I = Q^T Q = Q Q^T = (Q^T)^T Q^T,$$

therefore, Q^T is also an orthogonal matrix, so that the rows of Q are orthonormal as well. In particular, the column and the row elements of orthogonal $Q \in \mathbb{R}^{n \times n}$ satisfy $\sum_{i=1}^n q_{ij}^2 = \sum_{j=1}^n q_{ij}^2 = 1$. It follows also that a square orthogonal Q is nonsingular. Moreover

$$1 = \det I = \det(QQ^T) = \det Q \det Q^T = (\det Q)^2 \quad \Rightarrow \quad \det Q = \pm 1.$$

14.3 QR factorization

Definition 14.5 The *QR factorization* of an $m \times n$ matrix A is the representation

$$A = QR,$$

where Q is an $m \times m$ *orthogonal* matrix and R is an $m \times n$ *upper triangular* matrix,

$$\underbrace{\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}}_{n} = \underbrace{\begin{bmatrix} q_{11} & \cdots & q_{1n} & \cdots & q_{1m} \\ \vdots & & \vdots & & \vdots \\ q_{n1} & & q_{nn} & & q_{nm} \\ \vdots & & \vdots & & \vdots \\ q_{m1} & \cdots & q_{mn} & \cdots & q_{mm} \end{bmatrix}}_{m \geq n} \underbrace{\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & & \vdots \\ & & \ddots & \vdots \\ & & & r_{nn} \\ 0 & \cdots & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}}_{n} \Bigg\} m \geq n .$$

Due to the bottom zero element of R, the columns $\boldsymbol{q}_{n+1}, ..., \boldsymbol{q}_m$ are inessential for the representation itself, hence we can safely write

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} q_{11} & \cdots & q_{1n} \\ \vdots & & \vdots \\ q_{n1} & \cdots & q_{nn} \\ \vdots & & \vdots \\ q_{m1} & \cdots & q_{mn} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & & \vdots \\ & & \ddots & \vdots \\ & & & r_{nn} \end{bmatrix}. \tag{14.1}$$

The latter formula is called the *reduced* QR factorization.

Theorem 14.6 *Every matrix A has a QR factorization. If A is (square) non-singular, then a factorization $A = QR$ where R has a positive main diagonal is unique.*

Proof. Three algorithms of the QR factorization are given below:
1) the Gram–Schmidt orthogonalization (for the reduced version),
2) the Givens rotations and
3) the Householder reflections.
For the second part, let $A = QR$ be non-singular. Then $A^TA = R^TR$ is positive definite, and there is a unique Cholesky factorization $A = \widetilde{L}\widetilde{L}^T$ with $\widetilde{L}$ having a positive main diagonal. Thus, $R^T = \widetilde{L}$ is uniquely determined. □

Applications:
1) Solution to the linear least squares problem (Lecture 16);
2) QR-algorithm for determining eigenvalues (Part II Numerical Analysis).

14.4 The Gram-Schmidt orthogonalization

Let $A = QR$ be sought. If we denote the columns of A and Q by $(\boldsymbol{a}_j)$ and $(\boldsymbol{q}_j)$ respectively, then it follows from (14.1) that, given $(\boldsymbol{a}_k) \in \mathbb{R}^m$, we want to find an orthonormal set $(\boldsymbol{q}_j) \in \mathbb{R}^m$ that satisfies

$$\boldsymbol{a}_k = \textstyle\sum_{j=1}^{k} r_{jk}\boldsymbol{q}_j, \quad k = 1...n.$$

The latter problem can be solved for a general inner product space $\mathbb{X}$.

Theorem 14.7 *Let $\mathbb{X}$ be an inner product space, and let elements $\boldsymbol{a}_1, ..., \boldsymbol{a}_n \in \mathbb{X}$ be linearly independent. Then there exist elements $\boldsymbol{q}_1, ..., \boldsymbol{q}_n \in \mathbb{X}$ such that*

$$\textstyle\sum_{j=1}^{k} r_{jk}\boldsymbol{q}_j = \boldsymbol{a}_k, \qquad k = 1...n, \tag{14.2}$$

$$(\boldsymbol{q}_i, \boldsymbol{q}_j) = \delta_{ij}, \qquad i, j = 1...n. \tag{14.3}$$

Proof. From the first equation of (14.2), since $\boldsymbol{a}_1 \neq 0$,

$$r_{11}\boldsymbol{q}_1 = \boldsymbol{a}_1, \quad \|\boldsymbol{q}_1\| = 1 \quad \Rightarrow \quad r_{11}^2 = (\boldsymbol{a}_1, \boldsymbol{a}_1) \quad \Rightarrow \quad r_{11} = \|\boldsymbol{a}_1\|, \quad \boldsymbol{q}_1 = \boldsymbol{a}_1/\|a_1\|.$$

Suppose that the elements $\boldsymbol{q}_j$ which satisfy (14.2)-(14.3) are constructed for all $j < k$. The next element $\boldsymbol{q}_k$ should be the part of the orthonormal set $\boldsymbol{q}_1, ..., \boldsymbol{q}_{k-1}, \boldsymbol{q}_k$, and it should satisfy

$$r_{kk}\boldsymbol{q}_k + \textstyle\sum_{j=1}^{k-1} r_{jk}\boldsymbol{q}_j = \boldsymbol{a}_k. \tag{14.4}$$

Multiplying both sides of (14.4) by $\boldsymbol{q}_j$ (in the scalar product sense) and using orthonormality, we find the first $(k-1)$ coefficients

$$r_{jk} = (\boldsymbol{q}_j, \boldsymbol{a}_k), \quad j = 1, \ldots, k-1.$$

Substituting them back to (14.4) we obtain

$$r_{kk}\boldsymbol{q}_k = \boldsymbol{a}_k - \textstyle\sum_{j=1}^{k-1}(\boldsymbol{q}_j, \boldsymbol{a}_k)\boldsymbol{q}_j =: \boldsymbol{b}_k \quad \Rightarrow \quad r_{kk} = \|\boldsymbol{b}_k\|, \quad \boldsymbol{q}_k = \boldsymbol{b}_k/\|\boldsymbol{b}_k\|.$$

By construction, each $\boldsymbol{q}_j$ is a linear combination of $(\boldsymbol{a}_i)_{i\leq j}$, hence $\boldsymbol{b}_k := \boldsymbol{a}_k - \sum_{j=1}^{k-1} c_j \boldsymbol{a}_j$ is non-zero since (a_i) are linearly independent, i.e., $\boldsymbol{q}_k$ is well-defined. □

Algorithm 14.8 We may put the previous construction in the following algorithm

```
for k = 1 to n do
    b[k] := a[k];
    for j = 1 to k-1 do     #--  void if k=1
        r(j,k) := <q[j],a[k]>;
        b[k] := b[k] - r(j,k)*q[j]
    end
    r(k,k) := sqrt(<b[k],b[k]>);
    q[k] := b[k]/r(k,k)
end
```

Example 14.9 QR factorization by Gram-Schmidt of $A = \begin{bmatrix} 2 & 4 & 5 \\ 1 & -1 & 1 \\ 2 & 1 & -1 \end{bmatrix}$.

$$r_{11} = \|\boldsymbol{a}_1\| = 3, \quad \boldsymbol{q}_1 = \boldsymbol{a}_1/r_{11} = \tfrac{1}{3}\begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix};$$

$$r_{12} = (\boldsymbol{q}_1, \boldsymbol{a}_2) = 3, \qquad \boldsymbol{b}_2 = \boldsymbol{a}_2 - r_{12}\boldsymbol{q}_1 = \begin{bmatrix} 4 \\ -1 \\ 1 \end{bmatrix} - 3 \cdot \tfrac{1}{3}\begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ -2 \\ -1 \end{bmatrix},$$

$$r_{22} = \|\boldsymbol{b}_2\| = 3, \quad \boldsymbol{q}_2 = \boldsymbol{b}_2/r_{22} = \tfrac{1}{3}\begin{bmatrix} 2 \\ -2 \\ -1 \end{bmatrix};$$

$$r_{13} = (\boldsymbol{q}_1, \boldsymbol{a}_3) = 3, \quad r_{23} = (\boldsymbol{q}_2, \boldsymbol{a}_3) = 3, \qquad \boldsymbol{b}_3 = \boldsymbol{a}_3 - r_{13}\boldsymbol{q}_1 - r_{23}\boldsymbol{q}_2 = \begin{bmatrix} 5 \\ 1 \\ -1 \end{bmatrix} - 3 \cdot \tfrac{1}{3}\begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} - 3 \cdot \tfrac{1}{3}\begin{bmatrix} 2 \\ -2 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix},$$

$$r_{33} = \|\boldsymbol{b}_3\| = 3, \quad \boldsymbol{q}_3 = \boldsymbol{b}_3/r_{33} = \tfrac{1}{3}\begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}.$$

So,

$$A = \begin{bmatrix} 2 & 4 & 5 \\ 1 & -1 & 1 \\ 2 & 1 & -1 \end{bmatrix} = \underbrace{\frac{1}{3}\begin{bmatrix} 2 & 2 & 1 \\ 1 & -2 & 2 \\ 2 & -1 & -2 \end{bmatrix}}_{Q} \cdot \underbrace{\begin{bmatrix} 3 & 3 & 3 \\ & 3 & 3 \\ & & 3 \end{bmatrix}}_{R}$$

14.5 Further properties of orthogonal matrices

The Gram-Schmidt algorithm is very unstable: round-off errors accumulate rapidly and, even for moderate values of n, the computed matrix Q is no longer orthogonal. Much better algorithms are based on constructing Q as a composition of certain elementary orthogonal mappings.

Lemma 14.10 *A matrix Q is orthogonal* $\Leftrightarrow$ $\|Q\boldsymbol{x}\| = \|\boldsymbol{x}\|$ $\forall \boldsymbol{x} \in \mathbb{R}^m$ *(i.e., the mapping Q preserves Euclidean norm $\|\boldsymbol{x}\| = \sqrt{(\boldsymbol{x}, \boldsymbol{x})}$)*

Proof. Set $S := Q^T Q$. Then

$$\|Q\boldsymbol{x}\|^2 = (Q\boldsymbol{x}, Q\boldsymbol{x}) = (Q^T Q\boldsymbol{x}, \boldsymbol{x}) =: (S\boldsymbol{x}, \boldsymbol{x}).$$

If Q is orthogonal, then $S = I$, i.e. $\|Q\boldsymbol{x}\|^2 = (\boldsymbol{x}, \boldsymbol{x}) = \|\boldsymbol{x}\|^2$. Conversely, if Q is norm-preserving, then

$$(S\boldsymbol{x}, \boldsymbol{x}) = (\boldsymbol{x}, \boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathbb{R}^m \quad (\text{and } S = S^T).$$

Taking $\boldsymbol{x} = \boldsymbol{e}_i$, we get $s_{ii} = (S\boldsymbol{e}_i, \boldsymbol{e}_i) = 1$, while the choice $\boldsymbol{x} = \boldsymbol{e}_i + \boldsymbol{e}_j$ will provide $s_{ii} + 2s_{ij} + s_{jj} = 2$, whence $s_{ij} = 0$, i.e., $S = I$, using symmetry of S. □

A mapping which preserves the distance between any two points (as orthogonal mapping Q does) is called an *isometry*. It is clear that the composition of two isometrical mappings is an isometry itself. So, the following is true.

Lemma 14.11 *If P, Q are orthogonal, so is PQ.*

The simplest isometries in $\mathbb{R}^m$ are *rotations* and *reflections*. So, the idea of the next two algorithms is quite clear geometrically: given a sequence of vectors (a_i) we can always determine a sequence of elementary rotations (reflections) in $\mathbb{R}^m$ that brings the coordinates of (a_i) to the triangular form.

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 15

15 QR factorization (contd)

15.1 Further properties of orthogonal matrices

The Gram-Schmidt algorithm is very unstable: round-off errors accumulate rapidly and, even for moderate values of n, the computed matrix Q is no longer orthogonal. Much better algorithms are based on constructing Q as a composition of certain elementary orthogonal mappings.

Lemma 15.1 *A matrix Q is orthogonal* $\Leftrightarrow$ $\|Qx\| = \|x\|$ $\forall x \in \mathbb{R}^m$ *(i.e., the mapping Q preserves Euclidean length $\|x\| = \sqrt{(x,x)}$)*

A mapping which preserves the distance between any two points (as orthogonal mapping Q does) is called an *isometry*. It is clear that the composition of two isometrical mappings is an isometry itself. So, the following is true.

Lemma 15.2 *If P, Q are orthogonal, so is PQ.*

The simplest isometries in $\mathbb{R}^m$ are *rotations* and *reflections*. So, the idea of the next two algorithms is quite clear geometrically: given a sequence of vectors (a_i) we can always determine a sequence of elementary rotations (reflections) in $\mathbb{R}^m$ that brings the coordinates of (a_i) to the triangular form.

15.2 Givens rotations

Consider a problem: given $a, b \in \mathbb{R}$, find $c, s \in \mathbb{R}$ such that, with some r,

$$c^2 + s^2 = 1, \qquad \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

A solution always exists and is given by

$$c = \tfrac{a}{\sqrt{a^2+b^2}} =: \cos\phi, \qquad s = \tfrac{b}{\sqrt{a^2+b^2}} =: \sin\phi.$$

Generally, for any vector $a_p \in \mathbb{R}^m$, we can find a matrix $\Omega^{[p,q]}$ such that

$$\Omega^{[p,q]} a_p = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & c & & s & \\ & & & \ddots & & \\ & & -s & & c & \\ & & & & & \ddots \\ & & & & & & 1 \\ & & \uparrow & & \uparrow & \\ & & p & & q & \end{bmatrix} \begin{bmatrix} a_{1p} \\ \vdots \\ a_{pp} \\ \vdots \\ a_{qp} \\ \vdots \\ a_{np} \end{bmatrix} = \begin{bmatrix} a_{1p} \\ \vdots \\ r \\ \vdots \\ 0 \\ \vdots \\ a_{np} \end{bmatrix} \begin{matrix} \\ \\ \leftarrow p \\ \\ \leftarrow q \\ \\ \\ \end{matrix} \qquad \begin{aligned} c &= \tfrac{a_{pp}}{\sqrt{a_{pp}^2+a_{qp}^2}}, \\ s &= \tfrac{a_{qp}}{\sqrt{a_{pp}^2+a_{qp}^2}}. \end{aligned}$$

Such a matrix $\Omega^{[p,q]}$ is called a *Givens rotation*. The mapping $\Omega^{[p,q]}$ rotates the vectors in the two-dimensional plane spanned by e_p and e_q (clockwise by the angle ϕ), hence it is orthogonal.

Lemma 15.3 *Let A be an $m \times n$ matrix and, for any $1 \le p \le q \le m$, let $\widetilde{A} = \Omega^{[p,q]} A$. Then*

1) $\widetilde{a}_{qp} = 0$;
2) *the p-th and the q-th rows of $\widetilde{A}$ are linear combinations of the p-th and q-th rows of A;*
3) *all other rows of $\widetilde{A}$ remains the same as those of A.*

Proof. Follows immediately from the form of $\Omega^{[p,q]}$. □

From this lemma it follows that we can subsequently annihilate the subdiagonal elements, column by column, and this implies the next statement.

Theorem 15.4 *For any matrix A, there exist Givens matrices $\Omega^{[p,q]}$ such that*

$$R := \left(\Omega^{[m-1,m]}\right) \cdots \left(\Omega^{[2,m]} \cdots \Omega^{[2,3]}\right) \left(\Omega^{[1,m]} \cdots \Omega^{[1,3]}\Omega^{[1,2]}\right) A$$

is an upper triangular matrix.

Method 15.5

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \overset{\Omega^{[1,2]}}{\rightarrow} \begin{bmatrix} \bullet & \bullet & \bullet \\ 0 & \bullet & \bullet \\ * & * & * \\ * & * & * \end{bmatrix} \overset{\Omega^{[1,3]}}{\rightarrow} \begin{bmatrix} \bullet & \bullet & \bullet \\ 0 & * & * \\ 0 & \bullet & \bullet \\ * & * & * \end{bmatrix} \overset{\Omega^{[1,4]}}{\rightarrow} \begin{bmatrix} \bullet & \bullet & \bullet \\ 0 & * & * \\ 0 & * & * \\ 0 & \bullet & \bullet \end{bmatrix} \overset{\Omega^{[2,3]}}{\rightarrow} \begin{bmatrix} * & * & * \\ 0 & \bullet & \bullet \\ 0 & 0 & \bullet \\ 0 & * & * \end{bmatrix} \overset{\Omega^{[2,4]}}{\rightarrow} \begin{bmatrix} * & * & * \\ 0 & \bullet & \bullet \\ 0 & 0 & * \\ 0 & 0 & \bullet \end{bmatrix} \overset{\Omega^{[3,4]}}{\rightarrow} \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & \bullet \\ 0 & 0 & 0 \end{bmatrix}$$

The $\bullet$-elements have changed through a single rotation while the $*$-elements remain the same.

The cost. There are less than mn rotations and each rotation replaces two rows by their linear combinations, hence the total cost of computing R is $\mathcal{O}(mn^2)$.

If the matrix Q is required in an explicit form, set $\Omega = I$ and, for each successive rotation, replace Ω by $\Omega^{[p,q]}\Omega$. The final Ω is the product of all the rotations, in correct order, and we let $Q = \Omega^T$. The extra cost is $\mathcal{O}(m^2n)$.

If only one vector $Q^T\boldsymbol{b}$ is required, we multiply the vector by successive rotations, the cost being $\mathcal{O}(mn)$.

15.3 Householder transformations

Definition 15.6 Given any nonzero $\boldsymbol{u} \in \mathbb{R}^m$, the $m \times m$ matrix

$$H = H_{\boldsymbol{u}} = I - \frac{2}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T$$

is called a *Householder transformation*, or a *Householder reflection*. Since

$$H\boldsymbol{u} = -\boldsymbol{u}, \qquad H\boldsymbol{v} = \boldsymbol{v} \quad \text{if} \quad \boldsymbol{u}^T\boldsymbol{v} = 0,$$

this transformation reflects any vector $\boldsymbol{x} \in \mathbb{R}^m$ with respect to the $(m-1)$-dimensional hyperplane orthogonal to $\boldsymbol{u}$. Each such matrix H is symmetric and (since reflection is an isometry) orthogonal.

Lemma 15.7 *For any two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^m$ of equal length, the Householder transformation $H_{\boldsymbol{u}}$ with $\boldsymbol{u} = \boldsymbol{a} - \boldsymbol{b}$ reflects $\boldsymbol{a}$ onto $\boldsymbol{b}$,*

$$\boldsymbol{u} = \boldsymbol{a} - \boldsymbol{b}, \quad \|\boldsymbol{a}\| = \|\boldsymbol{b}\| \quad \Rightarrow \quad H_{\boldsymbol{u}}\boldsymbol{a} = \boldsymbol{b}.$$

In particular, for any $\boldsymbol{a} \in \mathbb{R}^m$, the choice $\boldsymbol{b} = \gamma\boldsymbol{e}_1$ implies

$$\boldsymbol{u} = \boldsymbol{a} - \gamma\boldsymbol{e}_1, \quad \gamma = \pm\|\boldsymbol{a}\| \quad \Rightarrow \quad H_{\boldsymbol{u}}\boldsymbol{a} = \gamma\boldsymbol{e}_1.$$

Proof. Draw a picture.

Example 15.8 To reflect

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \overset{H_{\boldsymbol{u}}}{\rightarrow} \begin{bmatrix} \gamma \\ 0 \\ 0 \end{bmatrix} = \gamma\boldsymbol{e}_1$$

with some $\boldsymbol{u}$ and γ, we should set $\gamma = \|\boldsymbol{a}\|$ or $\gamma = -\|\boldsymbol{a}\|$ to have equal lengths of vectors $\boldsymbol{a}$ and $\gamma \boldsymbol{e}_1$, and take

$$\boldsymbol{u} = \begin{bmatrix} a_1 - \|a\| \\ a_2 \\ a_3 \end{bmatrix} \quad \text{or} \quad \boldsymbol{u} = \begin{bmatrix} a_1 + \|a\| \\ a_2 \\ a_3 \end{bmatrix},$$

respectively. For example, we can reflect

$$\boldsymbol{a} = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} \to \begin{bmatrix} \gamma \\ 0 \\ 0 \end{bmatrix} = \gamma \boldsymbol{e}_1$$

either with

$$\gamma = 3 \quad \Rightarrow \quad \boldsymbol{u} = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix} \quad \Rightarrow \quad H_{\boldsymbol{u}} \boldsymbol{a} = \frac{1}{3} \begin{bmatrix} 2 & 1 & 2 \\ 1 & 2 & -2 \\ 2 & -2 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \end{bmatrix},$$

or with

$$\gamma = -3 \quad \Rightarrow \quad \boldsymbol{u} = \begin{bmatrix} 5 \\ 1 \\ 2 \end{bmatrix} \quad \Rightarrow \quad H_{\boldsymbol{u}} \boldsymbol{a} = \frac{1}{15} \begin{bmatrix} -10 & -5 & -10 \\ -5 & 14 & -2 \\ -10 & -2 & 11 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -3 \\ 0 \\ 0 \end{bmatrix}.$$

Choosing sign of γ. For calculations by hands, it is recommended to choose $\operatorname{sgn} \gamma = \operatorname{sgn} a_1$ as it leads to smaller numbers involved. However, numerically, it is the opposite choice that provides a better stability.

Example 15.9 To reflect

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \overset{H_{\boldsymbol{u}}}{\to} \begin{bmatrix} a_1 \\ a_2 \\ \gamma \\ 0 \end{bmatrix} = \boldsymbol{b}$$

we set $\gamma = \pm\sqrt{a_3^2 + a_4^2}$ (to equalize the lengths), and take

$$\boldsymbol{u} = \boldsymbol{a} - \boldsymbol{b} = \begin{bmatrix} 0 \\ 0 \\ a_3 - \gamma \\ a_4 \end{bmatrix}.$$

Theorem 15.10 *For any matrix A, there exist Housholder matrices H_k such that*

$$R := H_{n-1} \cdots H_2 H_1 A$$

is an upper triangular matrix.

Proof. We take $H_1 = H_{\boldsymbol{u}}$ with the vector $\boldsymbol{u} = \boldsymbol{a} - \gamma \boldsymbol{e}_1$, and a being the first column of A. Then $H_1 A$ has $\gamma \boldsymbol{e}_1$ as its first column. Explicitly,

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{u} = \begin{bmatrix} a_1 - \gamma \\ a_2 \\ \vdots \\ a_m \end{bmatrix}, \quad \gamma = \pm\|\boldsymbol{a}\| \quad \Rightarrow \quad H_1 \boldsymbol{a} = \begin{bmatrix} \gamma \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Suppose that the first $k-1$ columns of $C := H_{k-1} \cdots H_1 A$ have an upper triangular form, and let $\boldsymbol{c}$ be the k-th column of C. So, we define the next $H_k = H_{\boldsymbol{u}}$ taking $\boldsymbol{u}$ by the following rule

$$\boldsymbol{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_{k-1} \\ c_k \\ c_{k+1} \\ \vdots \\ c_m \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{u} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ c_k - \gamma \\ c_{k+1} \\ \vdots \\ c_m \end{bmatrix}, \quad \gamma^2 = \textstyle\sum_{i=k}^{m} c_i^2 \quad \Rightarrow \quad H_k \boldsymbol{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_{k-1} \\ \gamma \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

The conclusion (that the bottom $m-k$ component of $H_k c$ will vanish) is due to Lemma 15.7. Also, because of its zero components at the first $k-1$ positions, such a u is orthogonal to the previous $k-1$ columns of C, hence they remain invariant under reflection $H_k C$ (as well as the first $k-1$ rows of C). Therefore, the first k columns of $H_k C$ have an upper triangular form, and we can proceed further by induction. □

Method 15.11

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \xrightarrow{H_1} \begin{bmatrix} \bullet & \bullet & \bullet \\ 0 & \bullet & \bullet \\ 0 & \bullet & \bullet \\ 0 & \bullet & \bullet \end{bmatrix} \xrightarrow{H_2} \begin{bmatrix} * & * & * \\ 0 & \bullet & \bullet \\ 0 & 0 & \bullet \\ 0 & 0 & \bullet \end{bmatrix} \xrightarrow{H_3} \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & \bullet \\ 0 & 0 & 0 \end{bmatrix}$$

The $\bullet$-elements have changed through a single reflection while the $*$-elements remain the same.

Example 15.12

$$A = \left[\begin{array}{ccc} 2 & 4 & 7 \\ 0 & 3 & -1 \\ 0 & 0 & 3 \\ 0 & 0 & 4 \end{array}\right] \quad \Rightarrow \quad \gamma = 5, \quad u = \left[\begin{array}{c} 0 \\ 0 \\ -2 \\ 4 \end{array}\right] \quad \Rightarrow \quad \left(I - 2\frac{uu^{\mathrm{T}}}{\|u\|^2}\right) A = \left[\begin{array}{ccc} 2 & 4 & 7 \\ 0 & 3 & -1 \\ 0 & 0 & 5 \\ 0 & 0 & 0 \end{array}\right].$$

Calculation of Q and $Q^T b$. If the matrix Q is required in an explicit form, set $\Omega = I$ initially and, for each successive reflection, replace Ω by $H_u \Omega$. As in the case of Givens rotations, by the end of the computation, $Q = \Omega^T$. The same algorithm is being used to calculate the vector $c = Q^T b$.

Remark 15.13 For practical computations notice the following. For a matrix B, respectively a vector x, one should compute the products $H_u B$ and $H_u x$ as

$$H_u B = B - \frac{2}{\|u\|^2} u(u^T B), \qquad H_u x = x - 2\frac{u^T x}{\|u\|^2} u.$$

Deciding between Givens and Householder transformations. If A is dense, it is in general more convenient to use Householder reflections. Givens rotations come into their own, however, when A has many leading zeros in its rows. In an extreme case, if an $n \times n$ matrix A consists of zeros underneath the first subdiagonal, they can be 'rotated away' in just $n-1$ Givens rotations, at the cost of $\mathcal{O}(n^2)$ operations!

Dr A. Shadrin

Mathematical Tripos Part IB: Lent Term 2022

Numerical Analysis – Lecture 16

16 Linear least squares

16.1 Statement of the problem

Consider the problem of finding a vector $c \in \mathbb{R}^n$ such that $Ac = y$ where a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $y \in \mathbb{R}^m$ are given and $m > n$. When there are more equations than unknowns, the system $Ac = y$ is called *overdetermined*. In general, an overdetermined system has no solution, but we would like to have Ac and y close in a sense. Choosing the Euclidean distance $\|z\| = (\sum_{i=1}^m z_i^2)^{1/2}$ as a measure of closeness, we obtain the following problem.

Problem 16.1 (Least squares in $\mathbb{R}^m$) Given $A \in \mathbb{R}^{m\times n}$ and $y \in \mathbb{R}^m$, find

$$c^* = \arg\min_{c\in\mathbb{R}^n} \|Ac - y\|,$$

i.e., find (the argument) $c^* \in \mathbb{R}^n$ which minimizes (the functional) $\|Ac - y\|$.

Discussion 16.2 Problem of this form occur frequently when we collect m observations (x_i, y_i) (which, typically, are prone to measurement error) and wish to exploit them to form an n-variable linear model (given a-priori by some reasons)

$$f(x_i) \approx y_i, \quad i = \overline{1,m}, \qquad f(x) = c_1\phi_1(x) + c_2\phi_2(x) + \cdots + c_n\phi_n(x),$$

(e.g., trying to put some planet observations on, or near, an ellipse). That is, with f being a linear combination of ϕ_j's (as above), we want to determine particular values $c_j = c_j^*$ such that $f(x_i)$ would fit (y_i) in a certain way:

$$Ac = \begin{bmatrix} \phi_1(x_1) & \cdots & \phi_n(x_1) \\ \vdots & & \vdots \\ \phi_1(x_n) & \cdots & \phi_n(x_n) \\ \vdots & & \vdots \\ \phi_1(x_m) & \cdots & \phi_n(x_m) \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \\ \vdots \\ f(x_m) \end{bmatrix} \approx \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ \vdots \\ y_m \end{bmatrix} = y.$$

Of course, there will be many ways of doing this, but it is customary to determine c_i^*'s so as to minimize the sum of squares of the deviation, i.e., to minimize

$$\sum_{i=1}^m [f(x_i) - y_i]^2 = \|Ac - y\|^2,$$

for this leads to a *linear* system of equations for determination of the unknowns (c_j^*).

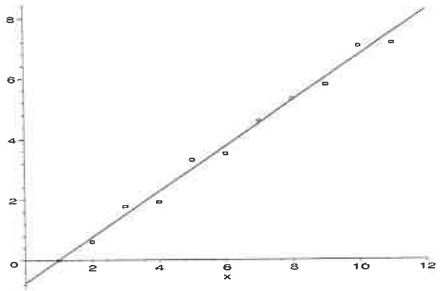


Fig 16.1 Least squares straight line data fitting

x_i	y_i
1	0.00
2	0.60
3	1.77
4	1.92
5	3.31
6	3.52
7	4.59
8	5.31
9	5.79
10	7.06
11	7.17

16.2 Normal equations

Theorem 16.3 *The vector $c^* \in \mathbb{R}^n$ is a solution to the least squares* $\Leftrightarrow$ $A^T(Ac^* - y) = 0$.

Proof. If c^* is a solution then it minimizes the quadratic form

$$F(c) := \|Ac - y\|^2 = (Ac - y, Ac - y) = c^T A^T Ac - 2c^T A^T y + y^T y\,,$$

hence the gradient $\nabla F = [\frac{\partial F}{\partial c_1}, \cdots, \frac{\partial F}{\partial c_n}]^T$ must vanish at $c = c^*$. So,

$$\tfrac{1}{2}\nabla F(c) = A^T Ac - A^T y, \quad \text{hence} \quad A^T(Ac^* - y) = 0.$$

Conversely, if $A^T(Ac^* - y) = 0$, then, for any $c' \in \mathbb{R}^n$, with $c := c' - c^*$, we find that

$$\begin{aligned} \|Ac' - y\|^2 &= (Ac + [Ac^* - y], Ac + [Ac^* - y]) \\ &= (Ac^* - y, Ac^* - y) + 2(A^T[Ac^* - y], c) + (Ac, Ac) \\ &= \|Ac^* - y\|^2 + \|Ac\|^2 \geq \|Ac^* - y\|^2, \end{aligned}$$

i.e., c^* is the least squares solution. Moreover, if A is non-singular and $c = c' - c^* \not\equiv 0$, then the last inequaliy is strict, hence c^* is unique. □

Corollary 16.4 *Optimality of c^** $\Leftrightarrow$ *the vector $Ac^* - y$ is orthogonal to all columns of A.*

Remark 16.5 This corollary has a clear geometrical visualization. If we denote the columns of A as $(a_j)_{j=1}^n$, then the least squares problem is the problem of finding the value $\min_c \| \sum_{j=1}^n c_j a_j - y\|$, which is the minimum of the Euclidean distance between a given vector y and vectors in the plane $\mathcal{A} := \text{span}\,(a_j)$. Geometrically it is clear that this minimum is attained when $a^* = \sum_{j=1} c_j^* a_j = Ac^*$ is the foot of the perpendicular from y onto the plane $\mathcal{A}$, i.e. when $y - \sum c_j a_j = y - Ac^*$ is orthogonal to all vectors a_j (columns of A).

Definition 16.6 Thus, we may find optimal c^* by solving the $n \times n$ system

$$A^T Ac^* = A^T y.$$

This system is called the *normal* equations, $A^T A$ is the *Gram* matrix, c^* is the *normal* solution.

Example 16.7 The least squares approximation to the data plotted in Fig. 16.1 by a straight line

$$f(x) = c_1 + c_2 x \qquad (\phi_1(x) = 1, \quad \phi_2(x) = x)$$

results in the problem $\|Ac - y\|^2 \to \min$ which normal solution is given by

$$A = (\phi_j(x_i)) = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ \vdots & \vdots \\ 1 & 11 \end{bmatrix}, \quad \underbrace{\begin{bmatrix} 11 & 66 \\ 66 & 506 \end{bmatrix}}_{A^T A} \begin{bmatrix} c_1^* \\ c_2^* \end{bmatrix} = \underbrace{\begin{bmatrix} 41.04 \\ 328.05 \end{bmatrix}}_{A^T y} \Rightarrow \begin{bmatrix} c_1^* \\ c_2^* \end{bmatrix} = \begin{bmatrix} -0.7314 \\ 0.7437 \end{bmatrix}.$$

Discussion 16.8 However, the approach based on the normal solution has three disadvantages. Firstly, $A^T A$ might be singular, secondly sparse A might be replaced by a dense $A^T A$ and, finally, forming $A^T A$ might lead to loss of accuracy.

16.3 QR and least squares

An alternative is provided by the QR factorization. Suppose that $A = QR$, with an orthogonal $Q \in \mathbb{R}^{m\times m}$ and an upper triangular $R \in \mathbb{R}^{m\times n}$. Then, since orthogonal mapping is length-preserving, i.e., $\|Q^T\boldsymbol{x}\|^2 = \|\boldsymbol{x}\|^2$ for any $x \in \mathbb{R}^m$, we have

$$\|A\boldsymbol{c} - \boldsymbol{y}\| = \|Q^T(A\boldsymbol{c} - \boldsymbol{y})\| = \|R\boldsymbol{c} - Q^T\boldsymbol{y}\|$$

therefore we may seek $\boldsymbol{c} \in \mathbb{R}^n$ that minimises $\|R\boldsymbol{c} - Q^T\boldsymbol{y}\|$.

Suppose for simplicity that $\operatorname{rank} R = \operatorname{rank} A = n$. Then the bottom $m - n$ rows of R are zero, thus we cannot approach corresponding components of $Q^T\boldsymbol{y}$ by $R\boldsymbol{c}$ no matter what the c_is are. So, we should try to match all others as best as we can, therefore we find $\boldsymbol{c}$ by solving the (nonsingular) linear system given by the first n equations of

$$R\boldsymbol{c} = Q^T\boldsymbol{y}.$$

Similar (although more complicated) algorithm applies when $\operatorname{rank} R \leq n - 1$. Note that we don't require Q explicitly and need to evaluate only $Q^T\boldsymbol{y}$.

16.4 Examples

Example 16.9 We wish to find $\boldsymbol{c} \in \mathbb{R}^3$ that minimizes $\|A\boldsymbol{c} - \boldsymbol{y}\|$, where

$$A = \frac{1}{2}\begin{bmatrix} 1 & 3 & 6 \\ 1 & 1 & 2 \\ 1 & 3 & 4 \\ 1 & 1 & 0 \end{bmatrix}, \qquad \boldsymbol{y} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

We QR-factorize

$$A = \frac{1}{2}\begin{bmatrix} 1 & 3 & 6 \\ 1 & 1 & 2 \\ 1 & 3 & 4 \\ 1 & 1 & 0 \end{bmatrix} = \underbrace{\frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}}_{Q} \times \underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{R}.$$

We thus need to solve the least-squares problem

$$\underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{R}\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} \approx \underbrace{\frac{1}{2}\begin{bmatrix} 3 \\ 1 \\ 1 \\ -1 \end{bmatrix}}_{Q^T\boldsymbol{y}}.$$

No matter what the c_i's are, we cannot approach the fourth component on the right-hand side, but all others can be matched. So, we solve the 'top' square system

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{c}^* = \frac{1}{2}\begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}, \quad \|A\boldsymbol{c}^* - \boldsymbol{y}\| = \|R\boldsymbol{c}^* - Q^T\boldsymbol{y}\| = \frac{1}{2}.$$

Example 16.10 Using the normal solution, find the least squares approximation to the data

x_i	y_i
-1	2
0	1
1	0

by a function $f = c_1\phi_1 + c_2\phi_2$, where $\phi_1(x) = x$ and $\phi_2(x) = 1 + x - x^2$.

We solve the system of normal equations:

$$A = (\phi_j(x_i)) = \begin{bmatrix} -1 & -1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, \qquad \underbrace{\begin{bmatrix} 2 & 2 \\ 2 & 3 \end{bmatrix}}_{A^TA} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \underbrace{\begin{bmatrix} -2 \\ -1 \end{bmatrix}}_{A^Ty} \Rightarrow \quad c^* = \begin{bmatrix} -2 \\ 1 \end{bmatrix}.$$

The error is

$$Ac^* - y = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} - \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ -1 \end{bmatrix} \quad \Rightarrow \quad \|Ac^* - y\| = \sqrt{2}.$$