Numerical Analysis Review Sheet

Michael Li

May 26, 2016

Polynomial approximation
Interpolation by polynomials. Divided differences of functions and relations to derivatives. Orthogonal polynomials and their recurrence relations. Least squares approximation by polynomials. Gaussian quadrature formulae. Peano kernel theorem and applications. [6]

Computation of ordinary differential equations
Euler's method and proof of convergence. Multistep methods, including order, the root condition and the concept of convergence. Runge-Kutta schemes. Stiff equations and A-stability. [5]

Systems of equations and least squares calculations
LU triangular factorization of matrices. Relation to Gaussian elimination. Column pivoting. Factorizations of symmetric and band matrices. The Newton-Raphson method for systems of non-linear algebraic equations. QR factorization of rectangular matrices by Gram-Schmidt, Givens and Householder techniques. Application to linear least squares calculations. [5]

Contents

1 Lagrange's Formula and Newton Divided Differences

1.1 Lagrange's Formula

Now from all the previous courses (Analysis, Complex Analysis, Differential Equations) we know that polynomials are good. They are great. So we want to interpolate some function with a polynomial so use that polynomial instead in our calculation. To do this, we first define some terms:

Definition (Lagrange cardinal polynomials)**.** The *Lagrange cardinal polynomials* with respect to the interpolation points $\{x_i\}_{i=0}^n$ are, for $k = 0, \cdots, n$,

$$\ell_k(x) = \prod_{i=0, i\neq k}^{n} \frac{x - x_i}{x_k - x_i}.$$

With these polynomials, if we have a function value at $n+1$ points with $f(x_k)$ denoted f_k, then we can immediately write down the solution to interpolating f_k at $n+1$ points:

$$p(x) = \sum_{k=0}^{n} f_k \ell_k(x).$$

Note this is the unique solution for an nth degree polynomial, as the difference of two solutions would share $n+1$ common points, so they are equal.

Newton built on this and developed an interpolation formula that can even be reused when we add more interpolation points.

1.2 Divided Differences

for $k = 0, \cdots, n$, we write $p_k \in P_k[x]$ for the polynomial that satisfies

$$p_k(x_i) = f_i \quad \text{for} \quad i = 0, \cdots, k.$$

This is the unique degree-k polynomial that satisfies the first $k+1$ conditions, whose existence (and uniqueness) is guaranteed by the previous section. Then we can write

$$p(x) = p_n(x) = p_0(x) + (p_1(x) - p_0(x)) + \cdots + (p_n(x) - p_{n-1}(x)).$$

We know that p_k and p_{k-1} agree on $x_0, \cdots, x_{k-1}$. So $p_k - p_{k-1}$ evaluates to 0 at those points, and we must have

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

for A_k the leading coefficient of $p_k(x)$. Then we can write

$$p(x) = p_n(x) = A_0 + \sum_{k=1}^{n} A_k \prod_{i=0}^{k-1} (x - x_i).$$

Now note stopping the sum at any k just gives the interpolation polynomial for k points. Now we define $A_k = f[x_j, \cdots, x_k]$, the leading coefficient of the unique $q \in P_{k-j}[x]$ such that $q(x_i) = f_i$ for $i = j, \cdots, k$. Now we have the following recurrence formula:

Theorem (Recurrence relation for Newton divided differences)**.** For $0 \leq j < k \leq n$, we have

$$f[x_j, \cdots, x_k] = \frac{f[x_{j+1}, \cdots, x_k] - f[x_j, \cdots, x_{k-1}]}{x_k - x_j}.$$

Proof. The key to proving this is to relate the interpolating polynomials. Let $q_0, q_1 \in P_{k-j-1}[x]$ and $q_2 \in P_{k-j}$ satisfy

$$\begin{aligned} q_0(x_i) &= f_i && i = j, \cdots, k-1 \\ q_1(x_i) &= f_i && i = j+1, \cdots, k \\ q_2(x_i) &= f_i && i = j, \cdots, k \end{aligned}$$

We now claim that

$$q_2(x) = \frac{x - x_j}{x_k - x_j} q_1(x) + \frac{x_k - x}{x_k - x_j} q_0(x).$$

We can check directly that the expression on the right correctly interpolates the points x_i for $i = j, \cdots, k$. By uniqueness, the two expressions agree. Since $f[x_j, \cdots, x_k]$, $f[x_{j+1}, \cdots, x_k]$ and $f[x_j, \cdots, x_{k-1}]$ are the leading coefficients of q_2, q_1, q_0 respectively, the result follows. □

Now we can construct Newton's famous divided difference tables:

x_i	f_i	$f[*,*]$	$f[*,*,*]$	$\cdots$	$f[*,\cdots,*]$
x_0	$f[x_0]$				
		$f[x_0,x_1]$			
x_1	$f[x_1]$		$f[x_0,x_1,x_2]$		
		$f[x_1,x_2]$		$\ddots$	
x_2	$f[x_2]$		$f[x_2,x_3,x_4]$	$\cdots$	$f[x_0,x_1,\cdots,x_n]$
		$f[x_2,x_3]$		$\iddots$	
x_3	$f[x_3]$		$\iddots$		
$\vdots$	$\vdots$	$\iddots$			
x_n	$f[x_n]$				

where each step is calculated by the recurrence formula.

2 Orthogonal Polynomials and Gaussian Quadrature Magic

2.1 Orthogonal Polynomials

On the space of polynomials, we define:

$$\langle f, g \rangle = \int_a^b w(x) f(x) d(x) \, \mathrm{d}x.$$

as the inner product with weight $w(x)$ such that $w(x) > 0$ on (a, b). Now we bring out the orthogonal polynomials:

Definition (Orthogonal polynomial)**.** Given a vector space V of polynomials and inner product $\langle \cdot, \cdot \rangle$, we say $p_n \in P_n[x]$ is the *nth orthogonal polynomial* if

$$\langle p_n, q \rangle = 0 \text{ for all } q \in P_{n-1}[x].$$

And it is *monic*, meaning its leading coefficient is 1. In particular, $\langle p_n, p_m \rangle = 0$ for $n \neq m$.

Now we have the following proof:

Theorem. Given a vector space V of functions and an inner product $\langle \cdot, \cdot \rangle$, there exists a unique monic orthogonal polynomial for each degree $n \geq 0$. In addition, $\{p_k\}_{k=0}^n$ form a basis for $P_n[x]$.

Proof. This is a big induction proof over both parts of the theorem. We induct over n. For the base case, we pick $p_0(x) = 1$, which is the only degree-zero monic polynomial.

Now suppose we already have $\{p_n\}_{k=0}^n$ satisfying the induction hypothesis.

Now pick any monic $q_{n+1} \in P_{n+1}[x]$, eg. x^{n+1}. We now construct p_{n+1} from q_{n+1} by the Gram-Schmidt process. We define

$$p_{n+1} = q_{n+1} - \sum_{k=0}^{n} \frac{\langle q_{n+1}, p_k \rangle}{\langle p_k, p_k \rangle} p_k.$$

This is again monic since q_{n+1} is, and we have

$$\langle p_{n+1}, p_m \rangle = 0$$

for all $m \leq n$, and hence $\langle p_{n+1}, p \rangle = 0$ for all $p \in P_n[x] = \langle p_0, \cdots, p_n \rangle$.

To obtain uniqueness, assume both $p_{n+1}, \hat{p}_{n+1} \in P_{n+1}[x]$ are both monic orthogonal polynomials. Then $r = p_{n+1} - \hat{p}_{n+1} \in P_n[x]$. So

$$\langle r, r \rangle = \langle r, p_{n+1} - \hat{p}_{n+1} \rangle = \langle r, p_{n+1} \rangle - \langle r, \hat{p}_{n+1} \rangle = 0 - 0 = 0.$$

So $r = 0$. So $p_{n+1} = \hat{p}_{n-1}$.

Finally, we have to show that $p_0, \cdots, p_{n+1}$ form a basis for $P_{n+1}[x]$. Now note that every $p \in P_{n+1}[x]$ can be written uniquely as

$$p = cp_{n+1} + q,$$

where $q \in P_n[x]$. But $\{p_k\}_{k=0}^n$ is a basis for $P_n[x]$. So q can be uniquely decomposed as a linear combination of $p_0, \cdots, p_n$. □

Now to get the linear polynomials, we don't use this proof. We do something a little easier,called the three-term recurrence relation:

Theorem. Monic orthogonal polynomials are generated by

$$p_{k+1}(x) = (x - \alpha_k)p_k(x) - \beta_k p_{k-1}(x)$$

with initial conditions

$$p_0 = 1, \quad p_1(x) = (x - \alpha_0)p_0,$$

where

$$\alpha_k = \frac{\langle xp_k, p_k \rangle}{\langle p_k, p_k \rangle}, \quad \beta_k = \frac{\langle p_k, p_k \rangle}{\langle p_{k-1}, p_{k-1} \rangle}.$$

Proof. By inspection, the p_1 given is monic and satisfies

$$\langle p_1, p_0 \rangle = 0.$$

Using $q_{n+1} = xp_n$ in the Gram-Schmidt process gives

$$p_{n+1} = xp_n - \sum_{k=0}^{n} \frac{\langle xp_n, p_k \rangle}{\langle p_k, p_k \rangle} p_k$$
$$p_{n+1} = xp_n - \sum_{k=0}^{n} \frac{\langle p_n, xp_k \rangle}{\langle p_k, p_k \rangle} p_k$$

We notice that $\langle p_n, xp_k \rangle$ and vanishes whenever xp_k has degree less than n. So we are left with

$$= xp_n - \frac{\langle xp_n, p_n \rangle}{\langle p_n, p_n \rangle} p_n - \frac{\langle p_n, xp_{n-1} \rangle}{\langle p_{n-1}, p_{n-1} \rangle} p_{n-1}$$
$$= (x - \alpha_n)p_n - \frac{\langle p_n, xp_{n-1} \rangle}{\langle p_{n-1}, p_{n-1} \rangle} p_{n-1}.$$

Now we notice that xp_{n-1} is a monic polynomial of degree n so we can write this as $xp_{n-1} = p_n + q$. Thus

$$\langle p_n, xp_{n-1} \rangle = \langle p_n, p_n + q \rangle = \langle p_n, p_n \rangle.$$

Hence the coefficient of p_{n-1} is indeed the β we defined. □

2.1.1 Least Squares Polynomial Fitting

Now in the real world, what we really want is to fit a polynomial that minimizes its error, or square of the error. So we have the following theorem:

Theorem. If $\{p_n\}_{k=0}^{n}$ are orthogonal polynomials with respect to $\langle \cdot, \cdot \rangle$, then the choice of c_k such that

$$p = \sum_{k=0}^{n} c_k p_k$$

minimizes $\|f - p\|^2$ is given by

$$c_k = \frac{\langle f, p_k \rangle}{\|p_k\|^2},$$

and the formula for the error is

$$\|f - p\|^2 = \|f\|^2 - \sum_{k=0}^{n} \frac{\langle f, p_k \rangle^2}{\|p_k\|^2}.$$

Proof. We consider a general polynomial

$$p = \sum_{k=0}^{n} c_k p_k.$$

We substitute this in to obtain

$$\langle f-p, f-p\rangle = \langle f, f\rangle - 2\sum_{k=0}^{n} c_k\langle f, p_k\rangle + \sum_{k=0}^{n} c_k^2\|p_k\|^2.$$

Note that there are no cross terms between the different coefficients due to orthogonality. We minimize this quadratic by setting the partial derivatives to zero:

$$0 = \frac{\partial}{\partial c_k}\langle f-p, f-p\rangle = -2\langle f, p_k\rangle + 2c_k\|p_k\|^2.$$

□

2.2 Ordinary and Gaussian Quadrature

We begin with some definitions:

Definition (Linear functional)**.** A *linear functional* is a linear mapping $L : V \to \mathbb{R}$, where V is a real vector space of functions.

Now the objective is to interpolate the following integral:

$$L(f) = \int_a^b w(x)f(x)\,\mathrm{d}x,$$

So using our good old Lagrange polynomials, we get:

Theorem (Ordinary quadrature)**.** For any distinct $\{c_k\}_{k=1}^{\nu} \subseteq [a, b]$, let $\{\ell_k\}_{k=1}^{\nu}$ be the Lagrange cardinal polynomials with respect to $\{c_k\}_{k=1}^{\nu}$. Then by choosing

$$b_k = \int_a^b w(x)\ell_k(x)\,\mathrm{d}x,$$

the approximation

$$L(f) = \int_a^b w(x)f(x)\,\mathrm{d}x \approx \sum_{k=1}^{\nu} b_k f(c_k)$$

is exact for $f \in P_{\nu-1}[x]$.

We call this method ordinary quadrature.

Now we prove an upper limit to how many degrees of polynomials we can interpolate:

Proposition. There is no choice of ν weights and nodes such that the approximation of $\int_a^b w(x)f(x)\,\mathrm{d}x$ is exact for all $f \in P_{2\nu}[x]$.

Proof. Define

$$q(x) = \prod_{k=1}^{\nu}(x - c_k) \in P_\nu[x].$$

Then we know

$$\int_a^b w(x)q^2(x)\,\mathrm{d}x > 0,$$

since q^2 is always non-negative and has finitely many zeros. However,

$$\sum_{k=1}^{\nu} b_k q^2(c_n) = 0.$$

So this cannot be exact for q^2. □

Now we prove that the zeros of the orthogonal polynomials on $[a, b]$ must lie in (a, b):

Theorem. For $\nu \geq 1$, the zeros of the orthogonal polynomial p_ν are real, distinct and lie in (a, b).

Proof. First we show there is at least one root. Notice that $p_0 = 1$. Thus for $\nu \geq 1$, by orthogonality, we know

$$\int_a^b w(x)p_\nu(x)p_1(x)\,\mathrm{d}x = \int_a^b w(x)p_\nu(x)\,\mathrm{d}x = 0.$$

So there is at least one sign change in (a, b). We have already got the result we need for $\nu = 1$, since we only need one zero in (a, b).

Now for $\nu > 1$, suppose $\{\xi_j\}_{j=1}^m$ are the places where the sign of p_ν changes in (a, b) (which is a subset of the roots of p_ν). We define

$$q(x) = \prod_{j=1}^{m}(x - \xi_j) \in P_m[x].$$

Since this changes sign at the same place as p_ν, we know qp_ν maintains the same sign in (a, b). Now if we had $m < \nu$, then orthogonality gives

$$\langle q, p_\nu\rangle = \int_a^b w(x)q(x)p_\nu(x)\,\mathrm{d}x = 0,$$

which is impossible, since qp_ν does not change sign. Hence we must have $m = \nu$. □

Now Gaussian quadrature is defined as the magic points and weights such that this approximation is exact for polynomials up to $2\nu - 1$ order. Yup. The theoretical maximum.

Theorem. In the ordinary quadrature, if we pick $\{c_k\}_{k=1}^\nu$ to be the roots of $p_\nu(x)$, then get we exactness for $f \in P_{2\nu-1}[x]$. In addition, $\{b_n\}_{k=1}^\nu$ are all positive.

Proof. Let $f \in P_{2\nu-1}[x]$. Then by polynomial division, we get

$$f = qp_\nu + r,$$

where q, r are polynomials of degree at most $\nu - 1$. We apply orthogonality to get

$$\int_a^b w(x)f(x)\,\mathrm{d}x = \int_a^b w(x)(q(x)p_\nu(x) + r(x))\,\mathrm{d}x = \int_a^b w(x)r(x)\,\mathrm{d}x.$$

Also, since each c_k is a root of p_ν, we get

$$\sum_{k=1}^{\nu} b_k f(c_k) = \sum_{k=1}^{\nu} b_k(q(c_k)p_\nu(c_k) + r(c_k)) = \sum_{k=1}^{\nu} b_k r(c_k).$$

But r has degree at most $\nu - 1$, and this formula is exact for polynomials in $P_{\nu-1}[x]$. Hence we know

$$\int_a^b w(x)f(x)\,\mathrm{d}x = \int_a^b w(x)r(x)\,\mathrm{d}x = \sum_{k=1}^{\nu} b_k r(c_k) = \sum_{k=1}^{\nu} b_k f(c_k).$$

To show the weights are positive, we simply pick a special f. Consider $f \in \{\ell_k^2\}_{k=1}^{\nu} \subseteq P_{2\nu-2}[x]$, for ℓ_k the Lagrange cardinal polynomials for $\{c_k\}_{k=1}^{\nu}$. Since the quadrature is exact for these, we get

$$0 < \int_a^b w(x)\ell_k^2(x)\,\mathrm{d}x = \sum_{j=1}^{\nu} b_j \ell_k^2(c_j) = \sum_{j=1}^{\nu} b_j \delta_{jk} = b_k.$$

Since this is true for all $k = 1, \cdots, \nu$, we get the desired result. □

3 Peano Kernel Theorem

Now we want to estimate its error. First up, definitions:

Definition (Sharp error bound)**.** The constant c_L is said to be *sharp* if for any $\varepsilon > 0$, there is some $f_\varepsilon \in C^{k+1}[a, b]$ such that

$$|e_L(f)| \geq (c_L - \varepsilon)\|f_\varepsilon^{(k+1)}\|_\infty.$$

Now we prove the Peano Kernel Theorem:

Theorem (Peano kernel theorem)**.** If λ annihilates polynomials of degree k or less, then

$$\lambda(f) = \frac{1}{k!}\int_a^b K(\theta)f^{(k+1)}(\theta)\,\mathrm{d}\theta$$

for all $f \in C^{k+1}[a, b]$, where the *Peano kernel* is

$$K(\theta) = \lambda((x-\theta)_+^k).$$

Proof. To proceed, we need Taylor's theorem with the integral remainder, ie.

$$f(x) = f(a) + (x-a)f'(a) + \cdots + \frac{(x-a)^k}{k!}f^{(k)}(a) + \frac{1}{k!}\int_a^x (x-\theta)^k f^{(k+1)}(\theta)\,\mathrm{d}\theta.$$

This is not really good, since there is an x in the upper limit of the integral. Instead, we write the integral as

$$\int_a^b (x-\theta)_+^k f^{(k+1)}(\theta)\,\mathrm{d}\theta,$$

where we define $(x-\theta)_+^k$ is a new function defined by

$$(x-\theta)_+^k = \begin{cases} (x-\theta)^k & x \geq \theta \\ 0 & x < \theta. \end{cases}$$

Then if λ is a linear functional that annihilates $P_k[x]$, then we have

$$\lambda(f) = \lambda\left(\frac{1}{k!}\int_a^b (x-\theta)_+^k f^{(k+1)}(\theta)\,\mathrm{d}\theta\right)$$

for all $f \in C^{k+1}[a,b]$.

For *our* linear functionals, we can simplify by taking the λ inside the integral sign (it works fine unless you are a pure mathematician) and obtain

$$\lambda(f) = \frac{1}{k!}\int_a^b \lambda((x-\theta)_+^k) f^{(k+1)}(\theta)\,\mathrm{d}\theta,$$

noting that λ acts on $(x-\theta)_+^k \in C^{k-1}[a,b]$ as a function of x, and think of θ as being held constant. □

4 Numerical Differential Equation Solvers

Now we come to solve ordinary differential equations numerically. As usual, definitions first:

Definition (Lipschitz function)**.** A function $\mathbf{f} : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$ is *Lipschitz with Lipschitz constant* $\lambda \geq 0$ if

$$\|\mathbf{f}(t,\mathbf{x}) - f(t,\hat{\mathbf{x}})\| \leq \lambda \|\mathbf{x} - \hat{\mathbf{x}}\|$$

for all $t \in [0,T]$ and $\mathbf{x}, \hat{\mathbf{x}} \in \mathbb{R}^N$.

A function is *Lipschitz* if it is Lipschitz with Lipschitz constant λ for some λ.

Definition ((Explicit) one-step method)**.** A numerical method is *(explicit) one-step* if $\mathbf{y}_{n+1}$ depends only on t_n and $\mathbf{y}_n$, ie.

$$\mathbf{y}_{n+1} = \boldsymbol{\phi}_h(t_n, \mathbf{y}_n)$$

for some function $\boldsymbol{\phi}_h : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N$.

Definition (Multi-step method)**.** A *s-step numerical method* is given by

$$\sum_{\ell=0}^{s} \rho_\ell \mathbf{y}_{n+\ell} = h \sum_{\ell=0}^{s} \sigma_\ell \mathbf{f}(t_{n+\ell}, \mathbf{y}_{n+\ell}).$$

This formula is used to find the value of $\mathbf{y}_{n+s}$ given the others.

Definition (Local truncation error)**.** For a general (multi-step) numerical method

$$\mathbf{y}_{n+1} = \boldsymbol{\phi}(t_n, \mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_n),$$

the *local truncation error* is

$$\boldsymbol{\eta}_{n+1} = \mathbf{y}(t_{n+1}) - \boldsymbol{\phi}_n(t_n, \mathbf{y}(t_0), \mathbf{y}(t_1), \cdots, \mathbf{y}(t_n)).$$

Where $\mathbf{y}(t_{n+1})$ is the true solution at t_{n+1}.

Definition (Order)**.** The order of a numerical method is the largest $p \geq 1$ such that $\boldsymbol{\eta}_{n+1} = O(h^{p+1})$.

4.1 Convergence

Definition (Convergence)**.** For each $h > 0$, we can produce a sequence of discrete values $\mathbf{y}_n$ for $n = 0, \cdots, [T/h]$, where $[T/h]$ is the integer part of T/h. A method *converges* if, as $h \to 0$ and $nh \to t$ (hence $n \to \infty$), we get

$$\mathbf{y}_n \to \mathbf{y}(t),$$

uniformly in t.

Note. The following example of convergence of the Euler's method serves as a prime example of how to answer these questions on the test. It doesn't really appear often but it has came up in the past few years and might as well reappear this year.

Theorem (Convergence of Euler's method)**.**

(i) For all $t \in [0, T]$, we have

$$\lim_{\substack{h\to 0\\ nh\to t}} \mathbf{y}_n - \mathbf{y}(t) = 0.$$

(ii) Let λ be the Lipschitz constant of f. Then there exists a $c \geq 0$ such that

$$\|\mathbf{e}_n\| \leq ch\frac{e^{\lambda T} - 1}{\lambda}$$

for all $0 \leq n \leq [T/h]$, where $\mathbf{e}_n = \mathbf{y}_n - \mathbf{y}(t_n)$.

Note that the bound in the second part is uniform. So this immediately gives the first part of the theorem.

Proof. There are two parts to proving this. We first look at the *local truncation error*. This is the error we would get at each step assuming we got the previous steps right. More precisely, we write

$$\mathbf{y}(t_{n+1}) = \mathbf{y}(t_n) + h(\mathbf{f}, t_n, \mathbf{y}(t_n)) + \mathbf{R}_n,$$

and $\mathbf{R}_n$ is the local truncation error. For the Euler's method, it is easy to get $\mathbf{R}_n$, since $\mathbf{f}(t_n, \mathbf{y}(t_n)) = \mathbf{y}'(t_n)$, by definition. So this is just the Taylor series expansion of $\mathbf{y}$. We can write $\mathbf{R}_n$ as the integral remainder of the Taylor series,

$$\mathbf{R}_n = \int_{t_n}^{t_{n+1}} (t_{n+1} - \theta) y''(\theta)\, \mathrm{d}\theta.$$

By some careful analysis, we get

$$\|\mathbf{R}_n\|_\infty \leq ch^2,$$

where

$$c = \frac{1}{2}\|y''\|_\infty.$$

This is the easy part, and tends to go rather smoothly even for more complicated methods.

Once we have bounded the local truncation error, we patch them together to get the actual error. We can write

$$\begin{aligned}\mathbf{e}_{n+1} &= \mathbf{y}_{n+1} - \mathbf{y}(t_{n+1}) \\ &= \mathbf{y}_n + h\mathbf{f}(t_n, \mathbf{y}_n) - \Big(\mathbf{y}(t_n) + h\mathbf{f}(t_n, \mathbf{y}(t_n)) + \mathbf{R}_n\Big) \\ &= (\mathbf{y}_n - \mathbf{y}(t_n)) + h\Big(\mathbf{f}(t_n, \mathbf{y}_n) - \mathbf{f}(t_n, \mathbf{y}(t_n))\Big) - \mathbf{R}_n\end{aligned}$$

Taking the infinity norm, we get

$$\begin{aligned}\|\mathbf{e}_{n+1}\|_\infty &\le \|\mathbf{y}_n - \mathbf{y}(t_n)\|_\infty + h\|\mathbf{f}(t_n, \mathbf{y}_n) - \mathbf{f}(t_n, \mathbf{y}(t_n))\|_\infty + \|\mathbf{R}_n\|_\infty \\ &\le \|\mathbf{e}_n\|_\infty + h\lambda\|\mathbf{e}_n\|_\infty + ch^2 \\ &= (1 + \lambda h)\|\mathbf{e}_n\|_\infty + ch^2.\end{aligned}$$

This is valid for all $n \ge 0$. We also know $\|\mathbf{e}_0\| = 0$. Doing some algebra, we get

$$\|\mathbf{e}_n\|_\infty \le ch^2 \sum_{j=0}^{n-1} (1 + h\lambda)^j \le \frac{ch}{\lambda}\left((1 + h\lambda)^n - 1\right).$$

Finally, we have

$$(1 + h\lambda) \le e^{\lambda h},$$

since $1 + \lambda h$ is the first two terms of the Taylor series, and the other terms are positive. So

$$(1 + h\lambda)^n \le e^{\lambda hn} \le e^{\lambda T}.$$

So we obtain the bound

$$\|\mathbf{e}_n\|_\infty \le ch\frac{e^{\lambda T} - 1}{\lambda}.$$

Then this tends to 0 as we take $h \to 0$. So the method converges. □

About the convergence of multi-step methods, we have the following powerful theorem:

Theorem (Dahlquist equivalence theorem)**.** A multi-step method is convergent if and only if

(i) The order p is at least 1; and

(ii) The root condition holds.

The proof is difficult and saved for Part III. The root condition is defined as:

Definition (Root condition)**.** We say $\rho(w)$ satisfies the *root condition* if all its zeros are bounded by 1 in size, ie. all roots w satisfy $|w| \le 1$. Moreover any zero with $|w| = 1$ must be simple.

Now we look at how to determine the order of a multistep method:

4.2 Order and Stability

4.2.1 Order

Theorem. An s-step method has order p ($p \geq 1$) if and only if

$$\sum_{\ell=0}^{s} \rho_\ell = 0$$

and

$$\sum_{\ell=0}^{s} \rho_\ell \ell^k = k \sum_{\ell=0}^{s} \sigma_\ell \ell^{k-1}$$

for $k = 1, \cdots, p$, where $0^0 = 1$.

Proof. The local truncation error is

$$\sum_{\ell=0}^{s} \rho_\ell \mathbf{y}(t_{n+\ell}) - h \sum_{\ell=0}^{s} \sigma_\ell \mathbf{y}'(t_{n+\ell}).$$

We now expand the $\mathbf{y}$ and $\mathbf{y}'$ about t_n, and obtain

$$\left(\sum_{\ell=0}^{s} \rho_\ell \right) \mathbf{y}(t_n) + \sum_{k=1}^{\infty} \frac{h^k}{k!} \left(\sum_{\ell=0}^{s} \rho_\ell \ell^k - k \sum_{\ell=0}^{s} \sigma_\ell \ell^{k-1} \right) \mathbf{y}^{(k)}(t_n).$$

This is $O(h^{p+1})$ under the given conditions. □

Note. This next result is just an easy extension of this result, but the sums are really horrible. Pay attention to the indices when summing!

Theorem. A multi-step method has order p (with $p \geq 1$) if and only if

$$\rho(e^x) - x\sigma(e^x) = O(x^{p+1})$$

as $x \to 0$.

Proof. We expand

$$\rho(e^x) - x\sigma(e^x) = \sum_{\ell=0}^{s} \rho_\ell e^{\ell x} - x \sum_{\ell=0}^{s} \sigma_\ell e^{\ell x}.$$

We now expand the $e^{\ell x}$ in Taylor series about $x = 0$. This comes out as

$$\sum_{\ell=0}^{s} \rho_\ell + \sum_{k=1}^{\infty} \frac{1}{k!} \left(\sum_{\ell=0}^{s} \rho_\ell \ell^k - k \sum_{\ell=0}^{s} \sigma_\ell \ell^{k-1} \right) x^k.$$

So the result follows. □

et's now come up with a sensible strategy for constructing convergent s-step methods:

(i) Choose a ρ so that $\rho(1) = 0$ and the root condition holds.

(ii) Choose σ to maximize the order, ie.

$$\sigma = \frac{\rho(w)}{\log w} + \begin{cases} O(|w-1|^{s+1}) & \text{if implicit} \\ O(|w-1|^s) & \text{if explicit} \end{cases}$$

We have the two different conditions since for implicit methods, we have one more coefficient to fiddle with, so we can get a higher order.

Where does the $\frac{1}{\log w}$ come from? We try to substitute $w = e^x$ (noting that $e^x - 1 \sim x$). Then the formula says

$$\sigma(e^x) = \frac{1}{x}\rho(e^x) + \begin{cases} O(x^{s+1}) & \text{if implicit} \\ O(x^s) & \text{if explicit} \end{cases}.$$

Rearranging gives

$$\rho(e^x) - x\sigma(e^x) = \begin{cases} O(x^{s+2}) & \text{if implicit} \\ O(x^{s+1}) & \text{if explicit} \end{cases},$$

which is our order condition. So given any ρ, there is only one sensible way to pick σ. So the key is in picking a good enough ρ.

The root conditions is "best" satisfied if $\rho(w) = w^{s-1}(w-1)$, ie. all but one roots are 0. Then we have

$$\mathbf{y}_{n+s} - \mathbf{y}_{n+s-1} = h\sum_{\ell=0}^{s} \sigma_\ell \mathbf{f}(t_{n+\ell}, \mathbf{y}_{n+\ell}),$$

where α is chosen to maximize order. These methods are called *Adams Methods*.

4.2.1.1 Runge-Kutta Methods

We introduce a class of useful but computationally complicated methods:

Definition (Runge-Kutta method)**.** General (implicit) ν-stage *Runge-Kutta (RK) methods* have the form

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\sum_{\ell=1}^{\nu} b_\ell \mathbf{k}_\ell,$$

where

$$\mathbf{k}_\ell = \mathbf{f}\left(t_n + c_n h, \mathbf{y}_n + h\sum_{j=1}^{\nu} a_{\ell j}\mathbf{k}_j\right)$$

for $\ell = 1, \cdots, \nu$.

We don't do much with it. Just notice the following few things:

(i) The best possible order of a ν-stage RK method is 2ν.

(ii) A standard notation is to put the coefficients in the Butcher table:

$$\begin{array}{c|ccc} c_1 & a_{11} & \cdots & a_{1\nu} \\ \vdots & \vdots & \ddots & \vdots \\ c_\nu & a_{\nu 1} & \cdots & a_{\nu\nu} \\ \hline & b_1 & \cdots & b_\nu \end{array}$$

Now we give a famous example:

Example. The most famous explicit Runge-Kutta method is the 4-stage 4th order one, often called the *classical* Runge-Kutta method. The formula can be given explicitly by

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4),$$

where

$$\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}(x_n, \mathbf{y}_n) \\
\mathbf{k}_2 &= \mathbf{f}\left(x_n + \frac{1}{2}h, \mathbf{y}_n + \frac{1}{2}h\mathbf{k}_1\right) \\
\mathbf{k}_3 &= \mathbf{f}\left(x_n + \frac{1}{2}h, \mathbf{y}_n + \frac{1}{2}h\mathbf{k}_2\right) \\
\mathbf{k}_4 &= \mathbf{f}\left(x_n + h, \mathbf{y}_n + h\mathbf{k}_3\right).
\end{aligned}$$

We see that this is an explicit method. We don't need to solve any equations.

4.2.2 Stability

Finally we look at stability.

Definition (Linear stability domain)**.** If we apply a numerical method to

$$y'(t) = \lambda y(t)$$

with $y(0) = 1$, $\lambda \in \mathbb{C}$, then its linear stability domain is

$$D = \left\{ z = h\lambda : \lim_{n\to\infty} y_n = 0 \right\}.$$

Definition (A-stability)**.** A numerical method is *A-stable* if

$$\mathbb{C}^- = \{z \in \mathbb{C} : \Re(z) < 0\} \subseteq D.$$

If a method is A-stable, that means we don't need to worry about picking too large an h so that the solution becomes unboundedly far away from the solution.

4.3 Implementation of ODE Methods

Note. This section has almost never been tested before. Partially it is (probably) because examiners don't know how to make this into a exam question, but it is more possible that it is just because they are lazy. Just remember the key formula listed below.

To keep things simple, we consider the two-step Adams-Bashforth method. Recall that this is given by

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{h}{2}(2\mathbf{f}(t_n, \mathbf{y}_n) - \mathbf{f}(t_{n-1}, \mathbf{y}_{n-1})).$$

This is an order 2 error with

$$\boldsymbol{\eta}_{n+1} = \frac{5}{12}h^3\mathbf{y}'''(t_n) + O(h^4).$$

The other multi-step method of order 2 is our old friend, the trapezoidal rule. This is an implicit method

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{h}{2}(\mathbf{f}(t_n, \mathbf{y}_n) + \mathbf{f}(t_{n+1}, \mathbf{y}_{n+1})).$$

This has a local truncation error of

$$\boldsymbol{\eta}_{n+1} = -\frac{1}{12}h^3\mathbf{y}'''(t_n) + O(h^4).$$

The key to Milne's device is the coefficients of $h^3\mathbf{y}'''(y_n)$, namely

$$c_{\mathrm{AB}} = \frac{5}{12}, \quad c_{\mathrm{TR}} = -\frac{1}{12}.$$

Since these are two different methods, we get different $\mathbf{y}_{n+1}$'s. We distinguish these by superscripts, and have

$$\begin{aligned}\mathbf{y}(t_{n+1}) - \mathbf{y}_{n+1}^{\mathrm{AB}} &\simeq c_{\mathrm{AB}}h^3\mathbf{y}'''(t_n)\\ \mathbf{y}(t_{n+1}) - \mathbf{y}_{n+1}^{\mathrm{TR}} &\simeq c_{\mathrm{TR}}h^3\mathbf{y}'''(t_n)\end{aligned}$$

We can now eliminate some terms to obtain

$$\mathbf{y}(t_{n+1}) - \mathbf{y}_{n+1}^{\mathrm{TR}} \simeq \frac{-c_{\mathrm{TR}}}{c_{\mathrm{AB}} - c_{\mathrm{TR}}}(\mathbf{y}_{n+1}^{\mathrm{AB}} - \mathbf{y}_{n+1}^{\mathrm{TR}}).$$

In this case, the constant we have is $\frac{1}{6}$. So we can estimate the local truncation error for the trapezoidal rule, without knowing the value of $\mathbf{y}'''$. We can then use this to adjust h accordingly.

5 LU Factorization

First up, definitions:

Definition (Triangular matrix)**.** A matrix A is *upper triangular* if $A_{ij} = 0$ whenever $i > j$. It is *lower triangular* if $A_{ij} = 0$ whenever $i < j$. We usually denote upper triangular matrices as U and lower triangular matrices as L.

Definition (LU factorization)**.** $A = LU$ is an *LU factorization* if U is upper triangular and L is *unit* lower triangular (ie. the diagonals of L are all 1).

it is easy to obtain the LU factorization by the following algorithm:

(i) Obtain $\mathbf{l}_1$ and $\mathbf{u}_1$ from the first row and column of A. Since the first entry of $\mathbf{l}_1$ is 1, $\mathbf{u}_1^T$ is exactly the first row of A. We can then obtain $\mathbf{l}_2$ by taking the first column of A and dividing by $U_{11} = A_{11}$.

(ii) Obtain $\mathbf{l}_2$ and $\mathbf{u}_2$ form the second row and column of $A - \mathbf{l}_1\mathbf{u}_1^T$ similarly.

(iii) $\cdots$

(iv) Obtain $\mathbf{l}_n$ and $\mathbf{u}_n^T$ from the nth row and column of $A - \sum_{i=1}^{n-1}\mathbf{l}_i\mathbf{u}_i^T$.

Now however, it doesn't always exist. Nope, even if it is non-singular it doesn't guarantee an LU factorization. This algorithm breaks down if $A_{kk}^{(k-1)} = (A - \sum_{i=1}^{k-1} \mathbf{l}_i \mathbf{u}_i^T)_{kk} = 0_{kk}$ for some k.

Now also from this that if we add a permutation matrix, then for non-singular matrices we can always have a factorization:

$$PA = LU,$$

where we interchange rows so that $A_{kk}^{(k-1)}$ isn't 0 (if all rows have 0 in this column, then the matrix is singular).

Definition (Leading principal submatrix)**.** The *leading principal submatrices* $A_k \in \mathbb{R}^{k\times k}$ for $k = 1, \cdots, n$ of $A \in \mathbb{R}^{n\times n}$ are

$$(A_k)_{ij} = A_{ij}, \quad i, j = 1, \cdots, k.$$

In other words, we take the first k rows and columns of A.

Theorem. A sufficient condition for the existence for both the existence and uniqueness of $A = LU$ is that $\det(A_k) \neq 0$ for $k = 1, \cdots, n-1$.

Note that we don't need A to be non-singular. This is equivalent to the restriction $A_{kk}^{(k-1)} \neq 0$ for $k = 1, \cdots, n-1$. Also, this is a *sufficient* condition, not a necessary one.

Proof. Straightforward induction. □

We can push the theorem a little further to get:

Theorem. If $\det(A_k) \neq 0$ for all $k = 1, \cdots, n$, then $A \in R^{n\times n}$ has a unique factorization of the form

$$A = LD\hat{U},$$

where D is non-singular diagonal matrix, and both L and $\hat{U}$ are unit triangular.

Proof. From the previous theorem, $A = LU$ exists. Since A is non-singular, U is non-singular. So we can write this as

$$U = D\hat{U},$$

where D consists of the diagonals of U and $\hat{U} = D^{-1}U$ is unit. □

This is not hard, but is rather convenient. In particular, it allows us to factorize symmetric matrices in a nice way.

Theorem. Let $A \in \mathbb{R}^{n\times n}$ be non-singular and $\det(A_k) \neq 0$ for all $k = 1, \cdots, n$. Then there is a unique "symmetric" factorization

$$A = LDL^T,$$

with L unit lower triangular and D diagonal and non-singular.

Proof. From the previous theorem, we can factorize A uniquely as

$$A = LD\hat{U}.$$

We take the transpose to obtain

$$A = A^T = \hat{U}^T D L^T.$$

This is a factorization of the form "unit lower"-"diagonal"-"unit upper". By uniqueness, we must have $\hat{U} = L^T$. So done. □

Theorem. A symmetric matrix $A \in \mathbb{R}^{n\times n}$ is *positive-definite* iff we can factor it as

$$A = LDL^T,$$

where L is unit lower triangular, D is diagonal and $D_{kk} > 0$.

Proof. First suppose such a factorization exists, then

$$\mathbf{x}^T A\mathbf{x} = \mathbf{x}^T LDL^T\mathbf{x} = (L^T\mathbf{x})^T D(L^T\mathbf{x}).$$

We let $\mathbf{y} = L^T\mathbf{x}$. Note that $\mathbf{y} = \mathbf{0}$ if and only if $\mathbf{x} = \mathbf{0}$, since L is invertible. So

$$\mathbf{x}^T A\mathbf{x} = \mathbf{y}^T D\mathbf{y} = \sum_{k=1}^{n} y_k^2 D_{kk} > 0$$

if $\mathbf{y} \neq 0$.

Now if A is positive definite, it has an LU factorization, and since A is symmetric, we can write it as

$$A = LDL^T,$$

where L is unit lower triangular and D is diagonal. Now we have to show $D_{kk} > 0$. We define $\mathbf{y}_k$ such that $L^T\mathbf{y}_k = \mathbf{e}_k$, which exist, since L is invertible. Then clearly $\mathbf{y}_k \neq \mathbf{0}$. Then we have

$$D_{kk} = \mathbf{e}_k^T D\mathbf{e}_k = \mathbf{y}_k^T LDL^T\mathbf{y}_k = \mathbf{y}_k^T A\mathbf{y}_k > 0.$$

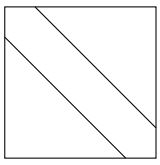
So done. □

This is a practical check for symmetric A being positive definite. We can perform this LU factorization, and then check whether the diagonal has positive entries.

Now we throw in this last bit:

Definition (Band matrix)**.** A *band matrix* of *band width* r is a matrix A such that $A_{ij} \neq 0$ implies $|i - j| \leq r$.

For example, a band matrix of band width 0 is a diagonal matrix; a band matrix of band width 1 is a tridiagonal matrix.



The result is

Proposition. If a band matrix A has band width r and an LU factorization $A = LU$, then L and U are both band matrices of width r.

Proof. Straightforward verification. □

6 QR Factorization

Definition (QR factorization)**.** A *QR factorization* of an $m \times n$ matrix A is a factorization of the form

$$A = QR,$$

where $Q \in \mathbb{R}^{m\times m}$ is an orthogonal matrix, and $R \in \mathbb{R}^{m\times n}$ is "upper triangular" matrix, where "upper triangular" means

$$R = \begin{pmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ 0 & R_{22} & \cdots & R_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_{mm} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

6.1 Gram-Schmidt

As we all know, the Gram-Schmidt process orthogonalizes vectors. What we are going to orthogonalize are the columns of A. We write

$$A = \begin{pmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{pmatrix}, \quad Q = \begin{pmatrix} \mathbf{q}_1 & \cdots & \mathbf{q}_n \end{pmatrix}.$$

By definition of the QR factorization, we need

$$\mathbf{a}_j = \sum_{i=1}^{j} R_{ij}\mathbf{q}_j.$$

This is just done in the usual Gram-Schmidt way.

(i) To construct column 1, if $\mathbf{a}_1 \neq \mathbf{0}$, then we set

$$\mathbf{q}_1 = \frac{\mathbf{a}_1}{\|\mathbf{a}_1\|}, \quad R_{11} = \|\mathbf{a}_1\|.$$

Note that the only non-unique possibility here is the sign $-$ we can let $R_{11} = -\|\mathbf{a}_1\|$ and $\mathbf{q}_1 = -\frac{\mathbf{a}_1}{\|\mathbf{a}_1\|}$ instead. But if we require $R_{11} > 0$, then this is fixed.

In the degenerate case, $\mathbf{a}_1 = \mathbf{0}$, then we can just set $R_{11} = 0$, and the pick any $\mathbf{q}_1 \in \mathbb{R}^n$ with $\|\mathbf{q}_1\| = 1$.

(ii) For columns $1 < k \leq n$, for $i = 1, \cdots, k-1$, we set

$$R_{ik} = \langle \mathbf{q}_i, \mathbf{a}_k \rangle,$$

and set

$$\mathbf{d}_k = \mathbf{a}_k - \sum_{i=1}^{k-1} \mathbf{q}_i \langle \mathbf{q}_i, \mathbf{a}_k \rangle.$$

If $\mathbf{d}_k \neq \mathbf{0}$, then we set

$$\mathbf{q}_k = \frac{\mathbf{d}_k}{\|\mathbf{d}_k\|},$$

and

$$R_{kk} = \|\mathbf{d}_k\|.$$

In the case where $\mathbf{d}_k = \mathbf{0}$, we again set $R_{kk} = 0$, and pick $\mathbf{q}_k$ to be anything orthonormal to $\mathbf{q}_1, \cdots, \mathbf{q}_{k-1}$.

6.2 Givens Rotations

Definition (Givens rotation)**.** In $\mathbb{R}^m$, where $m > 2$, we define the *Givens rotation* on 3 parameters $1 \le p < q \le m, \theta \in [-\pi, \pi]$ by

$$\Omega_\theta^{[p,q]} = \begin{pmatrix} 1 & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & 1 & & & & & & & \\ & & & \cos\theta & & & & \sin\theta & & \\ & & & & 1 & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 1 & & & \\ & & & -\sin\theta & & & & \cos\theta & & \\ & & & & & & & & 1 & \\ & & & & & & & & & \ddots \\ & & & & & & & & & & 1 \end{pmatrix} \in \mathbb{R}^{m\times m},$$

where the $\sin$ and $\cos$ appear at the p, qth rows and columns.

How exactly do we do this? Instead of writing down a general algorithm, we illustrate it with a matrix $A \in R^{4\times 3}$. The resultant R would look like

$$R = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & R_{33} \\ 0 & 0 & 0 \end{pmatrix}.$$

We will apply the Givens rotations in the following order:

$$\Omega_{\theta_6}^{[3,4]}\Omega_{\theta_5}^{[3,4]}\Omega_{\theta_4}^{[2,3]}\Omega_{\theta_3}^{[1,4]}\Omega_{\theta_2}^{[1,3]}\Omega_{\theta_1}^{[1,2]}A = R.$$

The matrix A transforms as follows:

$$\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{\Omega_{\theta_1}^{[1,2]}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{\Omega_{\theta_2}^{[1,3]}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{\Omega_{\theta_3}^{[1,3]}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}$$

$$\xrightarrow{\Omega_{\theta_4}^{[2,3]}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{\Omega_{\theta_5}^{[3,4]}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix} \xrightarrow{\Omega_{\theta_6}^{[3,4]}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix}$$

Note that when applying, say, $\Omega_{\theta_4}^{[2,3]}$, the zeros of the first column get preserved, since $\Omega_{\theta_4}^{[2,3]}$ only mixes together things in row 2 and 3, both of which are zero in the first column. So we are safe.

6.3 Householder Reflections

Definition (Householder reflection)**.** For $\mathbf{u} \neq \mathbf{0} \in \mathbb{R}^m$, we define the *Householder reflection* by

$$H_{\mathbf{u}} = I - 2\frac{\mathbf{u}\mathbf{u}^T}{\mathbf{u}^T\mathbf{u}} \in \mathbb{R}^{m\times m}.$$

Lemma. Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$, with $\mathbf{a} \neq \mathbf{b}$, but $\|\mathbf{a}\| = \|\mathbf{b}\|$. Then if we pick $\mathbf{u} = \mathbf{a} - \mathbf{b}$, then

$$H_{\mathbf{u}}\mathbf{a} = \mathbf{b}.$$

This is obvious if we draw some pictures in low dimensions.

Proof. We just do it:

$$H_{\mathbf{u}}\mathbf{a} = \mathbf{a} - \frac{2(\|\mathbf{a}\| - \mathbf{a}^T\mathbf{b})}{\|\mathbf{a}\|^2 - 2\mathbf{a}^T\mathbf{b} + \|\mathbf{b}\|^2}(\mathbf{a} - \mathbf{b}) = \mathbf{a} - (\mathbf{a} - \mathbf{b}) = \mathbf{b},$$

where we used the fact that $\|\mathbf{a}\| = \|\mathbf{b}\|$. □

We will use this to create out algorithm: So we begin our algorithm: we want to clear the first column of A. We let $\mathbf{a}$ be the first column of A, and assume $\mathbf{a} \in \mathbb{R}^m$ is not already in the correct form, ie. $\mathbf{a}$ is not a multiple of $\mathbf{e}_1$. Then we define

$$\mathbf{u} = \mathbf{a} \mp \|\mathbf{a}\|\mathbf{e}_1,$$

where either choice of the sign is pure-mathematically valid. However, we will later see that there is one choice that is better when we have to deal with inexact arithmetic. Then we end up with

$$H_1\mathbf{a} = H_{\mathbf{u}}\mathbf{a} = \pm\|\mathbf{a}\|\mathbf{e}_1.$$

Now we have

$$H_1 A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}.$$

To do the next step, we need to be careful, since we don't want to destroy the previously created zeroes.

Lemma. If the first $k-1$ components of $\mathbf{u}$ are zero, then

(i) For every $\mathbf{x} \in \mathbb{R}^m$, $H_{\mathbf{u}}\mathbf{x}$ does not alter the first $k-1$ components of $\mathbf{x}$.

(ii) If the last $(m-k+1)$ components of $\mathbf{y} \in \mathbb{R}^m$ are zero, then $H_{\mathbf{u}}\mathbf{y} = \mathbf{y}$.

These are all obvious from definition. All these say is that reflections don't affect components perpendicular to $\mathbf{u}$, and in particular fixes all vectors perpendicular to $\mathbf{u}$.

Lemma. Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$, with

$$\begin{pmatrix} a_k \\ \vdots \\ a_m \end{pmatrix} \neq \begin{pmatrix} b_k \\ \vdots \\ b_m \end{pmatrix},$$

but

$$\sum_{j=k}^{m} a_j^2 = \sum_{j=k}^{m} b_j^2.$$

Suppose we pick

$$\mathbf{u} = (0, 0, \cdots, 0, a_k - b_k, \cdots, a_m - b_m)^T.$$

Then we have

$$H_{\mathbf{u}}\mathbf{a} = (a_1, \cdots, a_{k-1}b_k, \cdots, b_m).$$

This is a generalization of what we've had before for $k = 1$. Again, the proof is just straightforward verification.

Now we can mess with the second column of H_1A. We let $\mathbf{a}$ be the second column of H_1A, and assume $a_3, \cdots, a_m$ are not all zero, ie. $(0, a_2, \cdots, a_m)^T$ is not a multiple of $\mathbf{e}_2$. We choose

$$\mathbf{u} = (0, a_2 \mp \gamma, a_3, \cdots, a_m)^T,$$

where

$$\gamma = \sqrt{\sum_{j=2}^{m} a_j}.$$

Then

$$H_{\mathbf{u}}\mathbf{a} = (a_1, \pm\gamma, 0, \cdots).$$

Also, by the previous lemma, this does not affect anything in the first column and the first row. Now we have

$$H_2H_1A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix}$$

Suppose we have reached $H_{k-1}\cdots H_1A$, where the first $k-1$ rows are of the correct form, ie.

$$H_{k-1}\cdots H_1A = \underbrace{\begin{bmatrix} \diagdown & \\ 0 & \end{bmatrix}}_{k-1}$$

We consider the kth column of $H_{k-1}\cdots H_1A$, say $\mathbf{a}$. We assume $(0, \cdots, 0, a_k, \cdots, a_m)^T$ is not a multiple of $\mathbf{e}_k$. Choosing

$$\mathbf{u} = \gamma(0, \cdots, 0, a_k \mp \gamma, a_{k+1}, \cdots, a_m)^t, \quad \gamma = \sqrt{\sum_{j=k}^{m} a_j^2},$$

we find that

$$H_{\mathbf{u}}\mathbf{a} = (a_1, \cdots, a_{k-1}, \pm\gamma, 0, \cdots, 0)^T.$$

Now we have

$$H_k\cdots H_1A = \underbrace{\begin{bmatrix} \diagdown & \\ 0 & \end{bmatrix}}_{k},$$

and H_k did *not* alter the first $k-1$ rows and columns of $H_{k-1}\cdots H_1A$.

6.4 Applications to Linear Least Squares

Finally, we consider a slightly different kind of problem, whose solution involves a different kind of factorization.

The question we are interested in is how we can "solve"

$$A\mathbf{x} = \mathbf{b},$$

where $A \in \mathbb{R}^{m\times n}$, with $m > n$, and $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{x} \in \mathbb{R}^n$.

Of course, this is an over-determined system. In general, this has no solution. So we want to find the "best" solution to this system of equations. More precisely, we want to find an $\mathbf{x}^* \in \mathbb{R}^n$ that minimizes $\|A\mathbf{x} - \mathbf{b}\|^2$. Note that we use the Euclidean norm. This is the *least squares problem*.

Why would one be interested in this? Often in, say, statistics, we have some linear model that says the outcome B is related to the input variables $A_1, \cdots, A_n$ by

$$B = A_1X_1 + A_2X_2 + \cdots + A_nX_n + \varepsilon,$$

where the X_i are some unknown parameters, and ε is some random fluctuation. The idea is to do lots of experiments, say m of them, collect the data in $A \in R^{m\times n}$ and $\mathbf{b} \in R^m$, and then find the X_i's that predict these results the most accurately, ie. find the $\mathbf{x}$ such that $A\mathbf{x}$ is as close to $\mathbf{b}$ as possible.

So how can we find this $\mathbf{x}^*$? First of all, we find a necessary and sufficient condition for $\mathbf{x}^*$ to be a solution to this.

Theorem. A vector $\mathbf{x}^* \in \mathbb{R}^n$ minimizes $\|A\mathbf{x} - \mathbf{b}\|^2$ if and only if

$$A^T(A\mathbf{x}^* - \mathbf{b}) = 0.$$

Proof. Checking the vanishing derivative condition gives the formula above as a necessary condition.

Now suppose our $\mathbf{x}^*$ satisfies $A^T(A\mathbf{x}^* - \mathbf{b}) = 0$. Then for all $\mathbf{x} \in \mathbb{R}^n$, we write $\mathbf{x} = \mathbf{x}^* + \mathbf{y}$, and then we have

$$\begin{aligned}\|A\mathbf{x} - \mathbf{b}\|^2 &= \|A(\mathbf{x}^* + \mathbf{y}) - \mathbf{b}\|\\ &= \|A\mathbf{x}^* - \mathbf{b}\| + 2\mathbf{y}^TA^T(A\mathbf{x} - \mathbf{b}) + \|A\mathbf{y}\|^2\\ &= \|A\mathbf{x}^* - \mathbf{b}\| + \|A\mathbf{y}\|^2\\ &\geq \|A\mathbf{x}^* - \mathbf{b}\|.\end{aligned}$$

So $\mathbf{x}^*$ must minimize the Euclidean norm. □

Corollary. If $A \in \mathbb{R}^{m\times n}$ is a full-rank matrix, then there is a unique solution to the least squares problem.

Proof. We know all minimizers are solutions to

$$(A^TA)\mathbf{x} = A^T\mathbf{b}.$$

The matrix A being full rank means $\mathbf{y} \neq \mathbf{0} \in \mathbb{R}^n$ implies $A\mathbf{y} \neq \mathbf{0} \in \mathbb{R}^m$. Hence $A^TA \in \mathbb{R}^{n\times n}$ is positive definite (and in particular non-singular), since

$$\mathbf{x}^TA^TA\mathbf{x} = (A\mathbf{x})^T(A\mathbf{x}) = \|A\mathbf{x}\|^2 > 0$$

for $\mathbf{x} \neq \mathbf{0}$. So we can invert A^TA and find a unique solution $\mathbf{x}$. □

Now to find the minimizing $\mathbf{x}^*$, we need to solve the *normal equations*

$$A^T A\mathbf{x} = A^T\mathbf{b}.$$

Now given $A = QR$, we can rearrange to get this new set of equations:

$$R^T(Rx - Q^T b) = 0$$

So we first solve $R^T y = 0$ and then solve $Rx = Q^T b + y$, which is much easier.