MATH 578 Numerical Analysis, Fall 2020
Student Notes

Kai Yang
kai.yang2@mail.mcgill.ca
McGill University
Montréal, Quebec Canada H3A 1A2

December 30, 2020

Abstract

This is just my re-organized notes going through the highlights from the course material for MATH 578 Numerical Analysis, Fall 2020 (Tsogtgerel, 2020) and the course textbook *Numerical linear algebra*. As a computational statistician muggle taking this course for optimization and machine learning, this notes might not always be a good summary. As an overview, the first chapter covers up some basics of analytic functions – specifically, Lagrange interpolation (Taylor series expansion is a special case) and minimax polynomials; the second chapter focuses on solving linear systems, which includes Gaussian elimination, LU factorization, QR factorization, with some content on error analysis; the third and fourth chapters are on eigenvalue problems and iterative methods (all eigenvalue problems must be iterative!), they mostly follow from L. Trefethen (1997), and consisting of QR algorithms, Arnoldi iteration, GMRES and CG. The last section is on quadrature and ODE solvers, where I summarized Chapter 11 of the book *Numerical Mathematics* very briefly.

I Function Evaluation

1 Basic Computer Arithmetic $\forall a \in \mathbb{Z}$, a *base−β representation* exists for some $\beta \in \mathbb{N} \setminus \{1\}$:

$$a = \pm \sum_{k=0}^{\infty} a_k \beta^k$$

where $0 \leq a_k \leq \beta - 1$ is defined as the k−th digit of a in base β. And grade-school column sum/difference first carries out *Cauchy sum* or *difference*, which takes sum/difference for

each digits; then it recursively perform carrying for addition for borrowing for subtraction. Let

$$n := \max\{k | a_k \neq 0\},\ m := \max\{k | b_k \neq 0\}$$

So a, b will be $n+1$ and $m+1$ digit number. The bit complexity for addition/subtraction will then be $O(n+m+1)$. Column multiplication carries out similarly. However, multiplication can also be done row-wisely: the *Cauchy product*

$$ab = \left(\sum_{i=0}^{\infty} a_i \beta^i\right) \cdot b = \sum_{i=0}^{n} a_i \cdot \beta^i b$$

where $\beta^j b$ is simply shifting digits, and multiplication by a_i can be carried out as column addition. The bit complexity for column multiplication would then be $O(nm+1)$.

As for division algorithm, assume that the quotient is expressed as:

$$q = q_0 + q_{-1}\beta^{-1} + q_{-2}\beta^{-2} + \cdots$$

And let a, b here be positive and normalized. The *partial reminder* refers to the *normalized* reminder obtained in the division process. Two division algorithms for a/b were introduced here: i). *restoring division*: keeping performing subtraction see if the partial reminder goes below 0, and if it goes below 0, "restore" by adding the divisor back to it to prevent negative digits; ii). *non-restoring division*: the idea of non-restoring division is to use generalized digit, e.g. $\{-1, 1\}$ for binary computing, to allow negative sign in a digit, and a conversion back to standard digit will be indeed required *in the end*. To generalize non-restoring division to any radix β, note that the partial reminders are given by:

$$r_{j+1} = \beta r_j - q_{-j} b$$

the above two division processes determine q_{-j} both by subtracting b from βr_j, the difference is for restoring division, $0 \leq q_{-j} < \beta$ gives partial reminder $0 \leq r_{j+1} < b$; for non-restoring division, $-\beta < q_j < \beta$ gives partial reminder $-b \leq r_{j+1} < b$.

However, both of above division algorithms are not efficient – especially not for bignums. WLOG, let a, b be integers here, the idea of *long division* is to determine the quotient by observing the first digit of the divisor and perform restoring division. In comparison, *SRT division* is non-restoring division with normalized divisor and reminder. *Error propagation* describes the idea of computation will alternate (mostly increase) the error of approximation numbers, such as floating point numbers. Usually error propagation is captured upper-boundedly by *conditional number*, e.g. conditional number of summation is

$$\kappa_+(x) = \frac{|x_1| + |x_2| + \cdots + |x_n|}{|x_1 + x_2 + \cdots + x_n|}$$

Furthermore, the following axiom is used for a wide-range of numerical error analysis for floating point numbers: For each $\star \in \{+,-,\times,/\}$, there exists a binary operation $\circledast : \tilde{\mathbb{R}} \times \tilde{\mathbb{R}} \mapsto \tilde{\mathbb{R}}$ s.t.

$$\left| x \star y - x \circledast y \right| \leq \varepsilon \left| x \star y \right|, \ x, y \in \tilde{\mathbb{R}}$$

dividing by zero is excluded. Normally, ε is referred as "machine precision."

2 Evaluation of Power Series A function $f : (a,b) \mapsto \mathbb{R}$ is called *analytic* at $c \in (a,b)$ if it can be developable into a power series around c; and called analytic at (a,b) if analytic at c, $\forall c \in (a,b)$. For such class of analytic functions, a way to evaluate them is through Taylor series, backed by a generalized version of mean value theorem proposed by Lagrange: Let f be a $n+1$ times differentiable function in (c,x), with $f^{(n)}$ continuous in $[c,x)$. Then $\exists \xi \in (c,x)$ s.t.

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(c)}{k!} (x-c)^k + \frac{f^{(n+1)}(\xi)}{(n+1)!} (x-c)^{n+1}$$

See Lagrange interpolation section coming later for proof. This theorem gives an expression of the error as a result of approximating using n–th order Taylor series. Moreover, the following series are listed with their relative condition numbers:

$$\begin{aligned}
\frac{1}{1-x} &= \sum_{k=0}^{\infty} x^k, \ \forall |x| < 1 \\
\kappa(x) &= \left| \frac{(1-x)^{-2}}{(1-x)^{-1}/x} \right| = \left| \frac{x}{1-x} \right| \\
e^x &= 1 + x + \frac{x^2}{2} + \cdots + \frac{x^n}{n!} + \cdots, \ \forall x \in \mathbb{R} \\
\kappa(x) &= \left| \frac{(e^x)'}{e^x/x} \right| = |x| \\
\log(1+x) &= \sum_{k=1}^{\infty} \frac{(-1)^{n-1} x^n}{k}, \ -1 < x \leq 1 \\
\kappa(x) &= \left| \frac{(1+x)^{-1}}{\log(1+x)/x} \right| = \frac{x}{(1+x)} \cdot \left| \frac{1}{\log(1+x)} \right| \\
\sin x &= \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}, \ \forall x \in \mathbb{R} \\
\kappa(x) &= \left| \frac{\cos x}{\sin x/x} \right| = |x \cot x|
\end{aligned}$$

$$\cos x = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!},\ \forall x \in \mathbb{R}$$

$$\kappa(x) = \left| \frac{-\sin x}{\cos x / x} \right| = |x \tan x|$$

$$\arctan x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{2n+1},\ |x| \le 1$$

$$\arcsin x = x + \frac{1}{2} \cdot \frac{x^3}{3} + \frac{1 \cdot 3}{2 \cdot 4} \cdot \frac{x^5}{5} + \frac{1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6} \cdot \frac{x^7}{7} + \cdots,\ -1 \le x < 1$$

And recall that the relative condition numbers is defined by:

$$\kappa := \lim_{\varepsilon \downarrow 0} \sup_{\|\delta x\| \le \varepsilon} \frac{\|\delta f\| / \|f(x)\|}{\|\delta x\| / \|x\|}$$

3 Acceleration of Convergence Two methods of acceleration of convergence are discussed here:

i). *Euler transform*: *Hausdorff moment characterization* says that

$$m_k = \int_0^1 x^k d\mu \text{ for some } \sigma - \text{additive Borel probability measure } \mu$$

$$\Leftrightarrow m_0 = 1,\ m \text{ is completely monotone; i.e. } (-1)^n \Delta^n m_k \ge 0,\ \forall n, k$$

The formula

$$\frac{\pi}{4} = \sum_{k=0}^{n} (-1)^k \frac{1}{2k+1}$$

can be accelerated by repeatedly taking average of two consecutive terms, called *Euler transform*. Applying Hausdorff moment characterization, error analysis for this can be done by noticing that

$$a_k = \frac{1}{k} = \int_0^1 t^k d\mu$$

and the rest follows from power series.

ii). *Aitken's Δ^2–process*: used to evaluate a noisy geometric series. For a series defined by

$$a_k = Cq^k + O\left(\delta^k\right), \text{ for some } 0 < \delta < q < 1$$

$$S_n = \sum_{k=1}^{n} a_k$$

Observe that

$$\begin{aligned} S =& S_n + \sum_{k=n+1}^{\infty} a_k \\ =& S_n + \sum_{k=n+1}^{\infty} Cq^k + O(\delta^n) \\ =& S_n + \frac{Cq^{n+1}}{1-q} + O(\delta^n) \\ =& S_n + \frac{a_n^2}{a_{n-1} - a_n} + O(\delta^n) \end{aligned}$$

The last inequality above used the fact that

$$\begin{aligned} q =& \frac{a_n}{a_{n-1}} + O\left(\left(\frac{\delta}{q}\right)^n\right) \\ a_n =& Cq^n + O(\delta^n) \\ \Rightarrow & Cq^{n+1} = \left(\frac{a_n}{a_{n-1}} + O\left(\left(\frac{\delta}{q}\right)^n\right)\right)(a_n - O(\delta^n)) = \frac{a_n^2}{a_{n-1}} + O(\delta^n), \text{ and} \\ & \frac{1}{1-q} = \frac{a_{n-1}}{a_{n-1} - a_n} + O\left(\left(\frac{\delta}{q}\right)^n\right) \end{aligned}$$

Let

$$\begin{aligned} \Delta S_{n-1} :=& S_n - S_{n-1} = a_n \\ \Delta^2 S_{n-2} :=& a_n - a_{n-1} = \Delta a_{n-1} \end{aligned}$$

We then have

$$S_n + \frac{a_n^2}{a_{n-1} - a_n} = S_n - \frac{(\Delta S_{n-1})^2}{\Delta^2 S_{n-2}}$$

which gives the name "Δ^2"

4 Root Finding Fixed point iterations are based on a theorem: Let $\phi : (a,b) \mapsto (a,b)$ be continuous. Further, let $x_{k+1} = \phi(x_k)$, $x_0 \in (a,b)$, and

$$\forall x, y \in (a,b),\ \exists \rho < 1 \text{ s.t. } \left|\phi(x) - \phi(y)\right| \leq \rho \left|x - y\right|$$

moreover, assume that $\exists \alpha \in (a,b)$ s.t. $\phi(\alpha) = \alpha$. Then $\forall x_0 \in (a,b)$, $x_n \to \alpha$ as $n \to \infty$ (linear convergence). Note that possible underlying connection to Lipschitz continuity here. And

recall that optimization can be more or less considered as a root finding procedure of the first-order optimality condition. The examples given here are chord method (corresponding to gradient descent), and Newton-Raphson method (local quadratic convergence).

5 Lagrange Interpolation The problem *Lagrange Interpolation* aims to solve is, given $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, find coefficients $a_0, \ldots, a_n$ for $p \in \mathbb{P}_n$ s.t.

$$p(x) \coloneqq \begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & & & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

One way to get the coefficients for the polynomial is to use *Lagrange coefficients*:

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

and

$$p(x) = \sum_{k=0}^{n} y_k \phi_k(x)$$

as we can observe that

$$\phi_j(x_i) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

Now define *Lagrange interpolation* as a map $\mathcal{L}_n : \mathcal{C}(a,b) \mapsto \mathbb{P}_n$, where $\{x_0, \ldots, x_n\} \subset (a,b)$ are distinct and fixed; i.e., to take $n+1$ points on f and construct the Lagrange polynomial passing through these $n+1$ points. Note that $\mathcal{L}_n$ is a projection, i.e. $\mathcal{L}_n \mathcal{L}_n = \mathcal{L}_n$. Recall we have seen how Lagrange generalized mean value theorem to higher-orders for Taylor series before, and here is the origin of Lagrange Theorem:

Let f be $n+1$th order differentiable in (a,b), and $x \in (a,b)$. Then $\exists \xi = \xi(x)$ s.t.

$$\min\{x_0, \ldots, x_n, x\} < \xi < \max\{x_0, \ldots, x_n, x\}, \text{ and} \tag{1}$$

$$f(x) - (\mathcal{L}_n f)(x) = \frac{(x - x_0) \cdots (x - x_n)}{(n+1)!} f^{(n+1)}(\xi)$$

The idea of proof is to construct the *Lagrange reminder*:

$$R(x) \coloneqq f(x) - (\mathcal{L}_n f)(x);\ A \coloneqq \frac{R(x)}{\prod_{i=0}^{n}(x - x_i)}$$

then the function

$$F(z) \coloneqq f(z) - (\mathcal{L}_n f)(x) - A\prod_{i=0}^{n}(z - x_i)$$

has $n+2$ distinct zeros $\{x_0, \ldots, x_n, x\}$; $F'(z)$ has $n+1$ distinct zeros; ...; $F^{(n+1)}(\xi) = f^{(n+1)}(\xi) - A(n+1)! = 0$ for some ξ in the convex hull as described in (1). This implies

$$f(x) - (\mathcal{L}_n f)(x) = R(x) = A\prod_{i=0}^{n}(x - x_i) = \frac{(x - x_0)\cdots(x - x_n)}{(n+1)!} f^{(n+1)}(\xi)$$

Note here how Rolle's theorem can be used to bridge the gap between higher-order in the last of the proof. Another interesting thing is that, Taylor's series expansion can be considered as Lagrange interpolation with repeated x_j.

6 Runge's Phenomenon *Runge's phenomenon* refers to the phenomenon that for a typical analytic function, equispaced Lagrange interpolation tends to oscillate more towards the boundary – that is, it tends to interpolate better in the middle. A typical analytic function will have $f^{(n)}(x) \sim \frac{n!}{\delta^n}$, and will have error$\sim \frac{\pi(x)}{\delta^n}$ for $\pi(x) = (x - x_0)\cdots(x - x_n)$. This suggests that *high-order polynomials on equispaced grid is not a good idea*, rather, it's a better idea to pick more points around the edge. Alternatively, it might be a better idea to approximate a function not by interpolating at certain points, but rather to minimize the upper bound of the approximation error norm – which leads to the discussion of the following three sections.

7 Weierstrass Approximation Theorem The *Weierstrass Approximation Theorem* states that a polynomial is dense in the space of continuous function in uniform norm: Let $f \in \mathcal{C}[a, b]$ and $\varepsilon > 0$; then $\exists n \in \mathbb{N}$, $\exists q \in \mathbb{P}_n(x)$ s.t.

$$\max_{x \in [a,b]} |f(x) - q(x)| \leq \varepsilon$$

Bernstein proposed a constructive proof back in 1904. WLOG, $[a, b] = [0, 1]$. Define *Bernstein polynomials* to have coefficients

$$\beta_{n,j}(x) = \binom{n}{j} x^j (1 - x)^{n-j}, \ j = 0, \ldots, n$$

i.e., binomial polynomial if you study stats... It has a few simple properties:

1. $\beta_{n,j}(x) > 0, \ \forall x \in (0, 1)$
2. $\sum_{j=0}^{n} \beta_{n,j}(x) = 1$

3. $\sum_{j=0}^{n} \frac{j}{n} \beta_{n,j}(x) = x$
4. $\sum_{j=0}^{n} \frac{j^2}{n^2} \beta_{n,j}(x) = \left(1 - \frac{1}{n}\right) x^2 + \frac{1}{n} x$

And the interpolation proceeds as: let $x_j = \frac{j}{n}$, $j = 0, 1, \ldots, n$, and let $B_n f(x) = \sum_{j=0}^{n} f\left(x_j\right) \beta_{n,j}(x)$. Observe that

$$\begin{aligned} f(x) - B_n f(x) =& f(x) \sum_{j=0}^{n} \beta_{n,j}(x) - \sum_{j=0}^{n} f\left(x_j\right) \beta_{n,j}(x) \\ =& \sum_{j=0}^{n} \left[f(x) - f\left(x_j\right) \right] \beta_{n,j}(x) \end{aligned}$$

Now split the function into two components:

$$\begin{aligned} R_\delta(x) :=& \left| \sum_{|x - x_j| \le \delta} \left[f(x) - f\left(x_j\right) \right] \beta_{n,j}(x) \right| \\ \le& \underbrace{\left| \sum_{j=0}^{n} \beta_{n,j}(x) \right|}_{=1} \cdot \underbrace{\max_{y \in [0,1],\ |x-y| \le \delta} \left| f(x) - f(y) \right|}_{=: \omega(\delta)} \\ S_\delta(x) :=& \left| \sum_{|x - x_j| > \delta} \left[f(x) - f\left(x_j\right) \right] \beta_{n,j}(x) \right| \end{aligned}$$

and construct interpolation sequence of $\xi_1, \xi_2, \ldots, \xi_p$ between x and x_j s.t. the distance (in Euclidean norm) between two neighbor points $\le \delta$, then

$$\begin{aligned} \left| f(x) - f\left(x_j\right) \right| \le& |f(x) - f(\xi_1)| + |f(\xi_1) - f(\xi_2)| + \cdots + \left| f\left(\xi_p\right) - f\left(x_j\right) \right| \\ \le& (p+1)\, \omega(\delta) \\ \le& \left(1 + \frac{\left| x - x_j \right|}{\delta} \right) \omega(\delta) \end{aligned}$$

This further implies that

$$|S_\delta(x)| \leq \underbrace{\sum_{|x-x_j|>\delta} \omega(\delta)\beta_{n,j}(x)}_{\leq \omega(\delta)} + \frac{\omega(\delta)}{\delta} \underbrace{\sum_{|x-x_j|>\delta} \left|x-x_j\right| \beta_{n,j}(x)}_{=:A} \leq \left(1+\frac{1}{4\delta^2 n}\right)\omega(\delta)$$

where above inequality uses the fact that

$$\begin{aligned}
\delta A \leq & \sum_{|x-x_j|>\delta} \left(x-x_j\right)^2 \beta_{n,j}(x) \\
\leq & \sum_{j=0}^{n} \left(x-x_j\right)^2 \beta_{n,j}(x) \\
= & x^2 \sum_{j=0}^{n} \beta_{n,j}(x) - 2x \sum_{j=0}^{n} \frac{j}{n} \beta_{n,j}(x) + \sum_{j=0}^{n} \frac{j^2}{n^2} \beta_{n,j}(x) \\
= & x^2 - 2x^2 + \left(1-\frac{1}{n}\right)x^2 + \frac{1}{n}x \\
= & \frac{x(1-x)}{n} \\
\leq & \frac{1}{4n}
\end{aligned}$$

Hence,

$$|f(x) - B_n f(x)| \leq \left(2+\frac{1}{4n\delta^2}\right)\omega(\delta), \ \forall \delta > 0 \text{ and } x \in [0,1]$$

Pick $\delta = \frac{1}{\sqrt{n}}$ completes the proof.

8 Minimax polynomials The *minimax polynomial* refers to the polynomial of a given degree that minimizes the uniform norm of the error for a continuous function on a closed interval, and its existence is ensured by the following theorem: Let $f \in \mathcal{C}[0,1]$ and $n \in \mathbb{N}_0$. Then $\exists p \in \mathbb{P}_n$ s.t.

$$\|f-p\|_\infty = \inf_{q \in \mathbb{P}_n} \|f-q\|_\infty$$

such q is called a *minimax polynomial* of degree n for f (on $[0,1]$).

The proof follows from continuous function achieves minimizer over a compact set (Weierstrass Theorem): For the sake of simplicity, let $a \in \mathbb{R}^{n+1}$ denote the coefficient vector for a

nth order polynomial q, and

$$E(a) := \|f - q\|_\infty = \max_{x\in[0,1]} |f(x) - q(x)|$$

First we are to prove the continuity of E:

$$\begin{aligned} |E(a+\delta a)| &\le \left| \|f - q - \delta q\|_\infty - \|f - q\|_\infty \right| \\ &\le \|\delta q\|_\infty \\ &\le |\delta a_0| + \cdots + |\delta a_n| \end{aligned}$$

Now let $K := \left\{ a \in \mathbb{R}^{n+1} | E(a) \le \|f\|_\infty + 1 \right\}$. Then:

1. K is closed, because $K = E^{-1}([0, \|f\|_\infty + 1])$ (pre-image of a closed set under continuous mapping is closed)
2. K is bounded, because $\|q\|_\infty \le \underbrace{\|f - q\|_\infty}_{=:E(a)} + \|f\|_\infty$ and

$$\|a\| \le \text{constant} \cdot \|q\|_\infty \Rightarrow E(a) \to \infty \text{ as } \|a\| \to \infty$$

3. Nonempty, because $0 \in K$

Thus, by Weierstrass Theorem, $\exists a^* \in K$ s.t. $E(a^*) = \inf_{a\in K} E(a)$ – but we still have to prove that $E(a^*) = \inf_{a\in\mathbb{R}^{n+1}} E(a)$:

$$E(a^*) \le E(0) = \|f\|_\infty \le \|f\|_\infty + 1 < E(a),\ \forall a \in \mathbb{R}^{n+1} \setminus K$$

9 Equioscillation Theorems Two important theorems are given to characterize minimax polynomials.

The first one is *De la Vallee Poussin Theorem*: $\forall f \in C[a,b],\ n \in \mathbb{N}_0,\ p \in \mathbb{P}_n$, if

$$f(x_j) - p(x_j) = (-1)^j e_j,\ \forall j = 0, 1, \ldots, n+1$$

where $a_0 \le x_0 < x_1 < \cdots < x_{n+1} \le b$, and sgn e_j = constant for $j = 0, 1, \ldots n+1$; then[1]

$$E_n(f) := \min_{q\in\mathbb{P}_n} \|f - q\|_\infty \ge \min_j |e_j|$$

[1] existence of minimax polynomial was proved in Section 8

The proof is by contradiction: assume that the conclusion is false, then

$$\begin{aligned} p\left(x_j\right) - q\left(x_j\right) &= (-1)^j e_j + \underbrace{f\left(x_j\right) - q\left(x_j\right)}_{<|e_j|,\ \forall j=0,1,\ldots,n+1} \\ \Rightarrow & p - q \text{ has } n+1 \text{ (distinct) zeros} \\ \Rightarrow & p \equiv q \end{aligned}$$

but it contradicts our assumption on p and q

The second one is *Chebyshev's Oscillation Theorem*, which characterizes the minimax polynomials: $p \in \mathbb{P}_n$ is a minimax polynomial for $f \in \mathcal{C}[0,1]$ iff $f-p$ takes the value $\pm\|f-p\|_\infty$, with alternating changes of sign, at least $n+2$ times in $[0,1]$. Moreover, this minimax polynomial is unique.

For statement besides uniqueness: Proof for "$\Leftarrow$" is done by DLVP, $\|f-p\|_\infty \le E_n(f) \Rightarrow \|f-p\|_\infty = E_n(f)$ by minimality of $E_n(f)$; proof for "$\Rightarrow$" is done by contradiction: assume the conclusion is false, i.e., $f-p$ takes the value $\pm\|f-p\|_\infty$ of k times for some $2 \le k \le n+1$[2], and let $\delta := \pm\|f-p\|_\infty$; then $f(x_i) - p(x_i) = (-1)^j \delta$ for $j = 1,\ldots,k$. And WLOG this allows us to (quasi-)partition $[0,1]$ into k intervals split by $\xi_1, \xi_2, \ldots, \xi_{k-1}$ s.t. on

$$\begin{aligned} (0,\xi_1),(\xi_2,\xi_3),\cdots &: -\delta \le f-p \le \delta-\varepsilon \\ (\xi_1,\xi_2),(\xi_3,\xi_4),\cdots &: -\delta+\varepsilon \le f-p \le \delta \end{aligned}$$

for some $\varepsilon > 0$. Now let $r(x) = \pm(x-\xi_1)\cdots(x-\xi_{k-1})$ – we'll discuss choice of sign shortly after, and let $q(x) := p(x) - \alpha \cdot r(x)$ for some small $\alpha > 0$ s.t. $\|\alpha r\|_\infty \le \frac{\varepsilon}{2}$, then $f-q = f-p+\alpha r$. Thus on

$$\begin{aligned} (0,\xi_1),(\xi_2,\xi_3),\cdots &: -\delta < -\delta+\alpha r \le f-q \le \delta - \frac{\varepsilon}{2} \\ (\xi_1,\xi_2),(\xi_3,\xi_4),\cdots &: -\delta+\frac{\varepsilon}{2} \le f-q \le \delta+\alpha r < \delta \end{aligned}$$

and we choose the sign of $r(x)$ s.t. $r > 0$ on the first line above and $r < 0$ on the second line above. Then q actually takes strictly less error than p, which contradicts that p is the minimax polynomial.

For uniqueness statement: let p, q both be minimax polynomials, and let $r := \frac{p+q}{2}$. Then

$$\begin{aligned} & |f-r| \le \frac{1}{2}|f-p| + \frac{1}{2}|f-q| \le E_n(f) \\ \Rightarrow & |f-r| = E_n(f) \text{ at } n+2 \text{ distinct points} \end{aligned}$$

[2] $k \ge 2$ because it's a minimax polynomial

$\Rightarrow f - p = f - q = \pm E_n(f)$ at those points – because $f - p = -(f - q) \Rightarrow f - r = 0$
$\Rightarrow p = q$ at $n+2$ distinct points
$\Rightarrow p \equiv q$

10 Chebyshev Polynomials Recall that the Runge's phenomenon suggests that the equispaced interpolation of polynomials does not approximate the function well, then we aim to position the interpolation points over a non-equal grid to approximate the function better. For example, we are to find the minimax polynomial in $\mathbb{P}_n$ for $f(x) = x^{n+1}$. Recall that $\sin, \cos$ usually brings oscillations, but they are not polynomials, then Chebyshev introduced a polynomial variant from it:

$$t_n(x) := \cos(n \arccos x)$$

which is based on the idea that *the projection of equispaced semi-circle on horizontal axis (see Fig. 1)*. The above polynomial formula gives us $t_n(x) = 1$, $t_1(x) = x$. Recall that

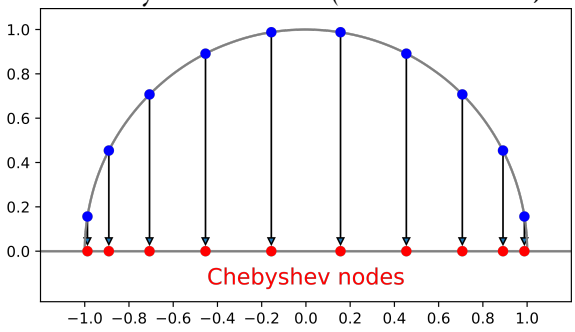
$$\cos((n+1)\theta) + \cos((n-1)\theta) = 2\cos(n\theta)\cos\theta$$

translate this into $t_n(x)$, it is

$$t_{n+1}(x) = 2t_n(x)x - t_{n-1}(x)$$

these are called *Chebyshev polynomials*, the zeros of $t_{n+1}(x)$ satisfy $(n+1)\arccos x = \frac{\pi}{2} + k\pi$ for $k = 0, 1, \ldots, n$.

Figure 1: Chebyshev's Nodes (L. N. Trefethen, 2013)



II Equation Solving

11 Gaussian Elimination The idea of *Gaussian Elimination*, is based on use upper rows to eliminate front-end matrix terms – one term at a time; and the resulting matrix will be an upper-triangular matrix. e.g.:

$$A = \underbrace{\begin{bmatrix} 2 & 1 & 1 \\ 4 & 3 & 3 \\ 8 & 7 & 9 \end{bmatrix}}_{A_1} \to \underbrace{\begin{bmatrix} 2 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 3 & 5 \end{bmatrix}}_{A_2} \to \underbrace{\begin{bmatrix} 2 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix}}_{A_3}$$

written in matrix form of above example, it will be

$$A_2 = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -4 & 0 & 1 \end{bmatrix}}_{\Lambda_1} A_1,\ A_3 = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -3 & 1 \end{bmatrix}}_{\Lambda_2} A_2$$

and note such Λ_k are always lower-triangular – in fact it only has nonzero entries at the kth column and all diagonal entries being 1, as we use upper rows to eliminate elements from lower rows.

12 LU-decomposition As a summary, in the example, $A_3 = \underbrace{\Lambda_2\Lambda_1}_{\Lambda} A$; as a product of lower-triangular matrices, Λ is also lower-triangular, hence Λ^{-1} is also lower-triangular. And the decomposition for full rank matrix $A = \Lambda^{-1}A_3$ is called *LU-decomposition*, in practice:

1. LU-decomposition has arithmetic complexity of roughly $\frac{1}{3}n^3$ multiplications;
2. LU-decomposition breaks down if $(A_k)_{k,k} = 0$ for some k
3. L and U can be stored in a single $n \times n$ array (because Λ^{-1} always has diagonal elements all being 1)

LU decomposition of A exists iff all principal minors of A are nonzero. If exists, LU decomposition is unique. Prove by noticing that Gaussian elimination always preserves principal minors. For uniqueness, let

$$LU = \hat{L}\hat{U} \Rightarrow \underbrace{\hat{L}^{-1}L}_{\text{lower-trig}} = \underbrace{\hat{U}U^{-1}}_{\text{upper-trig}} = I \Rightarrow \hat{U} = U,\ \hat{L} = L$$

Now the issue still remains if we encounter $(A_k)_{k,k} = 0$ for some k. To solve this issue, and also to make most prominent values (measured by Euclidean norm) up to the top to ensure numerical stability, *pivoting* is introduced. *Partial pivoting* means row interchanges (arithmetic complexity n^2); and *complete pivoting* refers to row and column interchanges (arithmetic complexity $\frac{1}{3}n^3$). An example for partial pivoting row interchange:

$$\underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{P} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} a_2 \\ a_4 \\ a_1 \\ a_3 \end{bmatrix}$$

and the *permutation matrix* P has properties that $PP^T = P^TP = I$, and the product of permutation matrix is still a permutation matrix (recall it just interchanges rows). Now partial pivoting LU decomposition performs pivoting after each elimination step, specifically,

$$\begin{aligned} U =& \Lambda_n P_n \Lambda_{n-1} P_{n-1} \cdots \Lambda_1 P_1 A \\ =& \Lambda_n \underbrace{\left(P_n \Lambda_{n-1} P_n^{-1}\right)}_{\Lambda'_{n-1}} \underbrace{\left(P_n P_{n-1} \Lambda_{n-2} P_{n-1}^{-1} P_n^{-1}\right)}_{\Lambda'_{n-2}} \cdots \underbrace{\left(P_n \cdots P_2 \Lambda_1 P_2^{-1} \cdots P_n^{-1}\right)}_{\Lambda'_1} P_n \cdots P_2 P_1 A \end{aligned}$$

where $\Lambda'_\cdot$ are *unit lower triangular matrices* – note that they are not lower triangular. And let $\Lambda' = \Lambda_n \Lambda'_{n-1} \Lambda'_{n-2} \cdots \Lambda'_1$, then $U = \Lambda' PA$, this gives

$$PA = LU$$

which is called *PLU-decomposition*. From the above pivoting process, it can be concluded that every square matrix has a PLU-decomposition.

13 Orthogonalization and QR-decomposition A matrix $Q \in \mathbb{R}^{n\times n}$ is called *orthogonal* if $Q^TQ = I$, i.e., if its column vectors form a orthonormal basis of $\mathbb{R}^n$. The idea of QR-decomposition comes from

$$Ax = QRx = b \Rightarrow QRx = b \Rightarrow Rx = Q^Tb$$

then $Rx = Q^Tb$ can be solved by back-substitution. If A, B are orthogonal, then AB and BA are both orthogonal. This allows us to perform QR-decomposition by a series of steps and times an orthogonal matrix at each step.

Recall that the projection of a on b is defined to be

$$\text{proj}_b a := \frac{\langle a,b\rangle}{\|a\|\cdot\|b\|}\cdot\|a\|\cdot\frac{b}{\|b\|} = \frac{\langle a,b\rangle}{\langle b,b\rangle}b$$

First we'll have a look at Gram-Schmidt method: let $a_1, a_2, \ldots, a_m \in \mathbb{R}^n$ be column vectors of $A \in \mathbb{R}^{n\times m}$, we can then form a orthonormal basis $q_1, q_2, \ldots, q_m$ by letting

$$\begin{aligned}
q_1 =& \frac{a_1}{\|a_1\|};\\
q_2' =& a_2 - \langle a_2, q_1\rangle q_1,\ q_2 = \frac{q_2'}{\|q_2'\|};\\
&\vdots\\
q_m' =& a_m - \sum_{k=1}^{m-1}\langle a_m, q_k\rangle q_k,\ q_m = \frac{q_m'}{\|q_m'\|}.
\end{aligned}$$

where $\|\cdot\|$ denote Euclidean norm. i.e., in each Gram-Schmidt step, first take off the projection of a_k onto the existing orthonormal basis s.t. the remaining vector will be orthogonal to the existing basis, then normalize a_k. Applying Gram-Schmidt to perform QR-decomposition, each Gram-Schmidt step can be considered as multiplication with a triangular matrix (i.e., step k will normalize $a_k^{(k)}$, and subtract the projections on q_k from $a_{k+1}^{(k)}, a_{k+2}^{(k)}, \ldots, a_m^{(k)}$):

$$\begin{bmatrix} a_1 & a_2 & \ldots & a_m \end{bmatrix}\begin{bmatrix} \frac{1}{\langle q_1,a_1\rangle} & -\frac{\langle q_1,a_2\rangle}{\langle q_1,a_1\rangle} & -\frac{\langle q_1,a_3\rangle}{\langle q_1,a_1\rangle} & \cdots & -\frac{\langle q_1,a_m\rangle}{\langle q_1,a_1\rangle} \\ 0 & 1 & 0 & & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} q_1 & a_2^{(2)} & a_3^{(2)} & \ldots & a_m^{(2)} \end{bmatrix}$$

Or view a_k as the sum of its projections on $q_1, q_2, \ldots, q_k$, from which we formulate

$$A = \begin{bmatrix} q_1 & q_2 & \ldots & q_m \end{bmatrix}\begin{bmatrix} \langle q_1,a_1\rangle & \langle q_1,a_2\rangle & \cdots & \langle q_1,a_m\rangle \\ 0 & \langle q_2,a_2\rangle & & \langle q_2,a_m\rangle \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \langle q_m,a_m\rangle \end{bmatrix} = QR$$

where q_k is obtained using Gram-Schmidt – note that this ensures all the 0s below diagonal.

14 QR-decomposition by Triangularization *Triangularization* refers to the idea of triangularizing a matrix by zeroing its below-diagonal entries. Here two methods are discussed.

The first one is *triangularization by givens rotation*: a *(clockwise) givens rotation matrix*[3] is defined by

$$G = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

And for $a = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$, to zero the second entry, i.e., to ensure

$$Ga = \begin{bmatrix} a_1\cos\theta + a_2\sin\theta \\ -a_1\sin\theta + a_2\cos\theta \end{bmatrix} = \begin{bmatrix} \sqrt{a_1^2 + a_2^2} \\ 0 \end{bmatrix}$$

we only need to let

$$\sin\theta = \frac{a_2}{\sqrt{a_1^2 + a_2^2}}$$
$$\cos\theta = \frac{a_1}{\sqrt{a_1^2 + a_2^2}}$$

Note that givens rotation matrix is orthogonal; and generalization to zeroing the $n-$dimensional vector entry a_{k+1} can be simply done by taking an identity matrix and mutate $\begin{bmatrix} I_{kk} & I_{k(k+1)} \\ I_{(k+1)k} & I_{(k+1)(k+1)} \end{bmatrix}$ to be the givens rotation matrix. Then for $A \in \mathbb{R}^{n\times m}$, we can zero the entries of a_i, $\forall i = 1, 2, \ldots, m$ in an order of $n, n-1, \ldots, i+1$ – we have to stop at $i+1$ for a_i as further zeroing will mutate the sparse patterns for zeroed $a_1, a_2, \ldots, a_{i-1}$. This way, we can obtain an upper-triangular matrix.

The second QR-decomposition method is *Householder's reflector*. Different from how givens rotations method rotates the vector to zeroing an entry, Householder's method will reflect the vector by a hyperplane H s.t. the reflection can point to the desired direction – one column vector at a time. Specifically, for $a_1 \in \mathbb{R}^n$, we try to multiply by an orthogonal matrix Q_1 s.t. $Q_1 a_1 = \|a_1\| e_1$ where e_1 having first entry being 1 and rest of entries being 0, the hyperplane H will be orthogonal to $v \coloneqq \|a_1\| e_1 - a_1$, therefore

$$Q_1 = I - 2\frac{vv^T}{v^T v}$$

[3] recall we have seen them in complex analysis

where Q_1 is orthogonal. In general,

$$Q_k = \begin{bmatrix} I & 0 \\ 0 & F \end{bmatrix}$$

where $I \in \mathbb{R}^{(k-1)\times(k-1)}$, and $F \in \mathbb{R}^{(n-k+1)\times(n-k+1)}$ s.t. $F\tilde{a}_k^{(k)} = \left\|\tilde{a}_k^{(k)}\right\| e_1$, where $\tilde{a}_k^{(k)} \in \mathbb{R}^{n-k+1}$ is $\left(\left(a_k^{(k)}\right)_k, \left(a_k^{(k)}\right)_{k+1}, \ldots, \left(a_k^{(k)}\right)_n\right)$; i.e., the upper-left sub-matrix I together with the two zero sub-matrices are to preserve obtained $a_1^{(k)}, a_2^{(k)}, \ldots, a_{k-1}^{(k)}$ from the first $k-1$ steps, and F is reflecting $\tilde{a}_k^{(k)}$ to obtain $a_k^{(k+1)}$ – which is to be preserved later; i.e., $a_k^{(k+1)} = a_k^{(k+2)} = \cdots = a_k^{(\min(m,n-1))}$.

15 Singular Value Decomposition (SVD) Click here for the slides I've made for SVD. The most important thing to remember is, *SVD represents a change of basis; i.e., giving adequate basis for domain space and range space, any matrix can be represented as a diagonal matrix.*

16 Cholesky Decomposition *Hermitian* is an analogue for complex matrices to symmetric for real matrices, defined to be $A = A^*$ if $a_{ij} = \bar{a}_{ji}$. Note that $x^*Ay = \overline{y^*Ax}$ for all $x, y \in \mathbb{C}^m$, this implies $\forall x \in \mathbb{C}^m,\ x^*Ax \in \mathbb{R}$; and it follows that eigenvalues for hermitian matrices are real. If in addition $x^*Ax > 0$ for all $x \neq 0$, then A is called *hermitian positive definite* (note that all positive definite matrices must be hermitian). The eigenvalues for such matrices are all positive – the converse is also true. Eigenvectors corresponding to distinct eigenvalues of a hermitian matrix are orthogonal:

$$\lambda_2 x_1^* x_2 = x_1^* A x_2 = \overline{x_2^* A x_1} = \lambda_1 \overline{x_2^* x_1} = \lambda_1 x_2^* x_2 \Rightarrow x_1^* x_2 = 0$$

Cholesky decomposition on positive definite matrices can be viewed as a *symmetric Gaussian elimination*; i.e., each Gaussian elimination step introduced zeros below the kth entry in column k by left multiplying a matrix, doing this symmetrically on the right will be

$$A = \begin{bmatrix} a_{11} & w^* \\ w & K \end{bmatrix} = \begin{bmatrix} \sqrt{a_{11}} & 0 \\ \frac{w}{\sqrt{a_{11}}} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & K - \frac{ww^*}{a_{11}} \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & \frac{w^*}{\sqrt{a_{11}}} \\ 0 & I \end{bmatrix}$$

and from positive definiteness of A we have $a_{11} > 0$ and $\begin{bmatrix} \sqrt{a_{11}} & 0 \\ \frac{w}{\sqrt{a_{11}}} & I \end{bmatrix}$ is non-singular, which implies $\begin{bmatrix} 1 & 0 \\ 0 & K - \frac{ww^*}{a_{11}} \end{bmatrix} \succ 0$, hence $K - \frac{ww^*}{a_{11}} \succ 0 \Rightarrow$ the upper left corner entry of $K - \frac{ww^*}{a_{11}}$ must be positive. This construction procedure also gives uniqueness of Cholesky

factorization.

17 Conditioning of $Ax = b$ Consider a linear system with numerical error:

$$\begin{aligned} &(A + \delta A)(x + \delta x) = b + \delta b \\ \Leftrightarrow &(A + \delta A)\delta x = \delta b - \delta A x \end{aligned}$$

From where, we would expect for all invertible A, $A + \delta A$ will also be invertible if δA is small. To solve this issue, first we'll look at *induced*[4] *matrix norm*, defined for any $A \in \mathbb{R}^{n\times m}$ by

$$\|A\| \coloneqq \sup_{x\in\mathbb{R}^m} \frac{\|Ax\|}{\|x\|} = \sup_{\|x\|\leq 1} \frac{\|Ax\|}{\|x\|} = \sup_{\|x\|=1} \|Ax\|$$

Immediately following from the definition, the matrix norm satisfies

- $\|\alpha A\| = |\alpha|\|A\|$ (*absolutely homogeneous*)
- $\|A + B\| \leq \|A\| + \|B\|$ (*triangle-inequality*)
- $\|A\| \geq 0$ and $\|A\| = 0 \Leftrightarrow A = 0$ (*positive-definiteness*)

and the well-known *Frobenius norm*:

$$\|A\|_F \coloneqq \sqrt{\sum_{j=1}^{m}\sum_{i=1}^{n}\left|a_{ij}\right|^2} = \operatorname{tr}\left(A^T A\right) = \operatorname{tr}\left(AA^T\right)$$

which leads to the use fact that: $\exists \alpha, \beta > 0$ s.t. $\alpha\|A\|_F \leq \|A\|_* \leq \beta\|A\|_F$, where $\|\cdot\|_*$ denotes any induced norm.

And for matrix geometric series, we are thinking of

$$(I - K)^{-1} = I + K + K^2 + \cdots \tag{2}$$

and (2) converges iff the $\ell - 2$ norm of all eigenvalues of A are strictly less than 1 – recall that $I - K$ is invertible iff 1 is not an eigenvalue of K. And specific for convergence proof, let

$$B_l \coloneqq I + K + \cdots + K^l$$

then we have

$$\|B_{l+m} - B_l\| = \left\|K^{l+1} + \cdots + K^{l+m}\right\|$$

[4] *induced* means matrix norm induced by vector norms

$$\leq \|K\|^{l+1} + \cdots + \|K\|^{l+m}$$
$$\leq \frac{\|K\|^{l+1}}{1-\|K\|}$$

if $\|K\| < 1$. Then $\{B_l\}$ is Cauchy, which implies that $\exists B \in \mathbb{R}^{n\times m}$ s.t. $B_l \to B$ as $l \to \infty$.

Now go back to our problem, we need $A + \delta A = A\left(I + A^{-1}\delta A\right)$ to be invertible, then we'll have $(A+\delta A)^{-1} = \left(I + A^{-1}\delta A\right)^{-1} A^{-1}$; and $\left(I + A^{-1}\delta A\right)^{-1}$ exists if $\left\|A^{-1}\delta A\right\| < 1$, note that $\left\|A^{-1}\delta A\right\| \leq \left\|A^{-1}\right\| \|\delta A\|$, then we only need $\|\delta A\| < \frac{1}{\left\|A^{-1}\right\|}$. The rest of error analysis follows from matrix norm properties and matrix geometric series properties[5]. Eventually, it can be derived that

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\|A\|\left\|A^{-1}\right\|}{1-\left\|A^{-1}\delta A\right\|}\left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|}\right)$$

where we define the *condition number for matrix A* as

$$\kappa(A) = \|A\|\left\|A^{-1}\right\|$$

18 Backward Error Analysis For floating point addition, we can treat them *as if* the input were perturbed; e.g.:

$$x_1 \oplus x_2 = (x_1 + x_2)(1+\delta) = (1+\delta)x_1 + (1+\delta)x_2 =: \tilde{x}_1 + \tilde{x}_2$$

Applying this treatment to the entire algorithm, we perform *backward error analysis (BEA)*. Let $\tilde{f} : \mathbb{R}^n \mapsto \mathbb{R}^m$ be some algorithmic realization of $f : \mathbb{R}^n \mapsto \mathbb{R}^m$, then BEA refers to the idea of model the errors committed within $\tilde{f}$ by error in the input data. The algorithm $\tilde{f}$ is called *stable* if *it gives nearly the right answer to nearly the right question*; i.e., $\forall x \in \mathbb{R}^n$, $\frac{\left\|\tilde{f}(x) - f(\tilde{x})\right\|}{\|f(\tilde{x})\|} = O(\varepsilon)$ for some $\tilde{x}$ with $\frac{\|\tilde{x}-x\|}{\|x\|} = O(\varepsilon)$. And a stronger condition defined as *backward stable* is used if *the algorithm gives exactly the right answer to nearly the right question*, i.e., $\forall x \in \mathbb{R}^n$, $\tilde{f}(x) = f(\tilde{x})$ for some $\tilde{x}$ with $\frac{\|\tilde{x}-x\|}{\|x\|} = O(\varepsilon)$. The error analysis for an algorithm can be written as

$$\tilde{f}(x) - f(x) = \tilde{f}(x) - f(\tilde{x}) + f(\tilde{x}) - f(x)$$

[5] which is frequently used when we deal with matrix inverse

from which we can deduce that

$$\frac{\left\|\tilde{f}(x)-f(x)\right\|}{\|f(x)\|} \leq \frac{\left\|\tilde{f}(x)-f(\tilde{x})\right\|}{\|f(x)\|} + \underbrace{\frac{\|f(\tilde{x})-f(x)\|}{\|f(x)\|}\cdot\frac{\|x\|}{\|\tilde{x}-x\|}}_{=:\kappa_f(x,\tilde{x})}\cdot\frac{\|\tilde{x}-x\|}{\|x\|}$$

In case of backward stability, we'll have

$$\frac{\left\|\tilde{f}(x)-f(x)\right\|}{\|f(x)\|} \leq \kappa_f(x,\tilde{x})\cdot\frac{\|\tilde{x}-x\|}{\|x\|}$$

Moreover, let $[x,y] \coloneqq \frac{\|x-y\|}{\|x\|}$ denote some error measure, and we define *backward stability constant of $\tilde{f}$ at x* to be

$$\beta(x) \coloneqq \inf_{\tilde{x}\in\mathbb{R}^n}\left\{[x,\tilde{x}]\,|\,\tilde{f}(x)=f(\tilde{x})\right\}$$

and backward stability was defined to be $\beta(x)=O(\varepsilon)$. Backward stability can be characterized by the following fact: if $\tilde{f}$ is backward stable at x, then

$$\begin{aligned}\left[f(x),\tilde{f}(x)\right] &= [f(x),f(\tilde{x})]\\ &= \frac{[f(x),f(\tilde{x})]}{\beta(x)}\cdot\beta(x)\\ &\leq \kappa_f(x)\cdot\beta(x)\\ &= O\left(\kappa_f(x)\cdot\varepsilon\right)\end{aligned}$$

Finally, as an example to describe backward non-stability, consider $f:\mathbb{R}\mapsto\mathbb{R}^2$ defined by $f(x)\coloneqq\begin{bmatrix}\cos x & \sin x\end{bmatrix}^T$, and let $\tilde{f}$ be an algorithm realization of f; but $\tilde{f}(x)\neq f(\tilde{x})$ because the range of f is the unit circle – so unless $\tilde{f}(x)$ maps to a unit circle $\forall x\in\mathbb{R}$, the algorithm $\tilde{f}$ can never be backward stable. It can be shown that naive outer product is not backward stable, and Householder's QR is backward stable.

III Eigenvalue Problems

19 Eigenvalue Problems Eigenvalue problems are most encountered in iteratively compound form of a matrix, such as power series A^k or e^{tA}. Let X be a matrix with

column vectors being eigenvectors of A, then we have

$$AX = X\underbrace{\begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix}}_{=:\Lambda} \Rightarrow AX = X\Lambda \Rightarrow A = X\Lambda X^{-1}$$

called the *eigenvalue decomposition* of $A \in \mathbb{C}^{m\times m}$ – it might well not exist. The eigenvalue decomposition can be expressed as a change of basis:

$$Ax = b \Rightarrow \left(X^{-1}b\right) = \Lambda\left(X^{-1}x\right)$$

The set of eigenvectors corresponding to a single eigenvalue forms a subspace of $\mathbb{C}^m$, known as the *eigenspace*, denoted by E_λ; and we have $AE_\lambda \subseteq E_\lambda$. The dimension of E_λ can be interpreted as the maximum number of linearly independent eigenvectors that can be found, defined as the *geometric multiplicity* of λ; recall $\text{null}(A-\lambda I) = E_\lambda$, the geometric multiplicity is then $\dim E_\lambda = \dim(\text{null}(A-\lambda I))$.

The *characteristic polynomial* of $A \in \mathbb{C}^{m\times m}$ is defined as

$$p_A(z) = \det(zI - A)$$

The main usage reflects in the fact that λ is an eigenvalue of $A \Leftrightarrow p_A(\lambda) = 0$. Recall that fundamental theorem of algebra allows us to write $p_A(z)$ as a product of root, which raises the definition of *algebraic multiplicity* of an eigenvalue λ as its multiplicity as a root of p_A.

If $X \in \mathbb{C}^{m\times m}$ is nonsingular, then the map $A \mapsto X^{-1}AX$ is called a *similarity transformation* of A. To characterize similarity transformation, note that if X is nonsingular, then A and $X^{-1}AX$ have the same characteristic polynomial, eigenvalues, as well as algebraic and geometric multiplicities – the rest all follow from characteristic polynomial proof:

$$p_{X^{-1}AX}(z) = \det\left(zI - X^{-1}AX\right) = \det\left(X^{-1}(zI-A)X\right) = \det\left(X^{-1}\right)\det(zI-A)\det(X) = \det(zI-A)$$

and similarity transformation allows us to prove the following proposition: the algebraic multiplicity of an eigenvalue λ is greater or equal to its geometric multiplicity. To see this, for matrix $A \in \mathbb{C}^{m\times m}$, let n be the geometric multiplicity of λ for A, apparently $n \le m$, now form a matrix $V = \begin{bmatrix} \hat{V} & \tilde{V} \end{bmatrix}$, where the column vectors of $\hat{V}$ form a orthonormal basis for E_λ, and $\tilde{V} \in \mathbb{C}^{m\times(m-n)}$ is chosen s.t. V is unitary; then

$$V^*AV = \begin{bmatrix} \lambda I & C \\ 0 & D \end{bmatrix} \tag{3}$$

will have characteristic polynomial $\det(zI - \lambda I)\det(zI - D) = (z-\lambda)^n \det(zI - D)$, which completes the proof.

Note that the geometric and algebraic multiplicity doesn't have to be the same: e.g., the matrix $\begin{bmatrix} 2 & & \\ & 2 & \\ & & 2 \end{bmatrix}$ and $\begin{bmatrix} 2 & 1 & \\ & 2 & 1 \\ & & 2 \end{bmatrix}$ both have same algebraic multiplicity 3 for eigenvalue $\lambda = 2$, but the former matrix has $\dim E_\lambda = 3$, spanned by e_1, e_2, e_3; the later one, however, has $\dim E_\lambda = 1$, spanned only by e_1. An eigenvalue whose algebraic multiplicity exceeds geometric multiplicity is called a *defective eigenvalue*; and a matrix that has a defective eigenvalue is a *defective matrix*. The following diagonalization proposition characterizes non-defectiveness: a matrix A is non-defective if it has an eigenvalue decomposition $A = X\Lambda X^{-1}$; "$\Leftarrow$" can be shown by using the similarity transformation proposition we've shown above, and "$\Rightarrow$" can be shown by constructing X using linearly independency of eigenvectors implied by non-defectiveness. As a sidenote, *for both defective and non-defective matrices*, the determinant and trace equal to the product and sum of eigenvalues counted with algebraic multiplicity (proved by using characteristic polynomial).

Note that not all diagonalizable matrices have orthogonal eigenvectors – those who do are called unitarily diagonalizable; i.e., $\exists Q$ unitary s.t. $A = Q\Lambda Q^*$. And matrix A is defined to be *normal* if $A^*A = AA^*$. To characterize normality, A is unitarily diagonalizable $\Leftrightarrow$ A is normal; proof for "$\Rightarrow$" is trivial by substitution, proof for "$\Leftarrow$" can be done by Schur factorization.

A *Schur factorization* of a matrix A is a factorization $A = QTQ^*$ where Q is unitary and T is upper-triangular. Every square matrix A has a Schur factorization; the proof follows induction on the dimension of A: similar to (3), for arbitrary square matrix A, $\exists U$ unitary s.t.

$$U^*AU = \begin{bmatrix} \lambda & B \\ 0 & C \end{bmatrix}$$

assume a Schur factorization VTV^* exist for C, then $Q \coloneqq U\begin{bmatrix} 1 & 0 \\ 0 & V \end{bmatrix}$ is unitary and $Q^*AQ = \begin{bmatrix} \lambda & BV \\ 0 & T \end{bmatrix}$.

20 Eigenvalue Algorithms An explicit solving method for eigenvalue problems will be reducing to a root finding problem of its characteristic polynomials – but root finding is in fact an ill-conditioned problem *per se*, so it is not really an ideal approach. Moreover, note that for a monic polynomial

$$p(z) = z^m + a_{m-1}z^{m-1} + \cdots + a_1 z + a_0$$

the roots are in fact the eigenvalues of the matrix

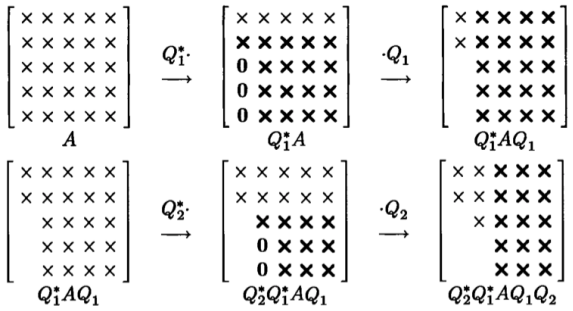
$$A := \begin{bmatrix} 0 & 1 & & & & & \\ & 0 & 1 & & & & \\ & & 0 & 1 & & & \\ & & & \ddots & \ddots & & \\ & & & & 0 & 1 & \\ -a_0 & -a_1 & -a_2 & \cdots & \cdots & -a_{m-2} & -a_{m-1} \end{bmatrix}$$

with corresponding eigenvectors $\left(1, z, z^2, \ldots, z^{m-1}\right)^T$. Therefore, A is called a *companion matrix* corresponding to p. Moreover, Abel proved that no analogue of the quadratic formula can exist for polynomial of degree 5 or more; in view of this, *any eigenvalue solver must be iterative*.

Recall form last section that Schur factorization exists for all matrices, it then follows that most general eigenvalues solver aims to reduce a matrix to by Schur factorization to reveal its eigenvalues. For the sake of computational efficiency, today's algorithms usually splits into two phrases: the first phrase reduce the matrix to a upper–Hessenberg form, and the second phrase upper-triangularizes this upper-Hessenberg matrix.

Now for the first phrase, we apply Householder's reflection to reduce A to a upper-Hessenberg form. At step k, Q_k^* perform Householder's reflection *starting the second row of the (sub-)matrix*, so that right multiplication will not mutate the zeros just created (see Fig. 2).

Figure 2: reduce to upper-Hessenberg form using Householder's reflection (L. Trefethen, 1997)



21 Rayleigh Quotient, Inverse Iteration For this and next section, we consider $A = A^T \in \mathbb{R}^{m\times m}$ and $x \in \mathbb{R}^m$. The *Rayleigh quotient* of a vector $x \in \mathbb{R}^m$ is defined as

$$r(x) := \frac{x^T Ax}{x^T x}$$

Apparently, if x is an eigenvector, then $r(x) = \lambda$. The motivation comes from: given x, what scalar α *acts most like an eigenvalue for* x *in the sense of minimizing* $\|Ax - \alpha x\|_2$? Furthermore, we have

$$\nabla r(x) = \frac{2}{x^T x}(Ax - r(x)x)$$

Therefore, at eigenvector x of A, the gradient is a zero vector; moreover, if the gradient is zero at x with $x \neq 0$, then x is an eigenvector and $r(x)$ is the corresponding eigenvalue. Furthermore, note that $\forall \lambda \in \mathbb{R} \setminus \{0\}$, $r(\lambda x) = r(x)$; we can then view the Rayleigh quotient as a continuous function on the unit sphere, and its stationary points are normalized eigenvectors of A.

The *power iteration* process can be used to find the eigenvector corresponding to the largest eigenvalue of A: initialized with unit vector $v^{(0)}$, then the update process proceeds as $v^{(k)} = \text{normalized } Av^{(k-1)}$, and $\lambda^{(k)} = \left(v^{(k)}\right)^T Av^{(k)}$.

Recall that the eigenvectors of $(A-\mu I)^{-1}$ are the same as the eigenvectors of A, with corresponding eigenvalues $\left\{\left(\lambda_j - \mu\right)^{-1}\right\}$. Therefore, if choosing μ closer to λ_j than to all other eigenvalues of A, $\left(q_j - \mu\right)^{-1}$ is greater than all other $\left(q_{-j} - \mu\right)^{-1}$; which suggests applying power iteration on $(A-\mu I)^{-1}$ can be used to find eigenvector corresponding to the eigenvalue closest to μ; this process is called *inverse iteration*, which takes updating formula $(A-\mu I)w = v^{(k-1)}$ and normalize w for $v^{(k)}$.

With a small modification to inverse iteration above: at each step, instead of using μ, we use $r\left(v^{(k-1)}\right)$ – this is called *Rayleigh quotient iteration*.

22 QR Algorithm The second phrase of eigenvalue problems is based on an algorithm called *QR algorithm*: initialize $A^{(0)} = A$; for each step, perform QR factorization $Q^{(k)}R^{(k)} = A^{(k-1)}$, then recombine factors $A^{(k)} = R^{(k)}Q^{(k)}$. Under suitable assumptions, it will converge to an upper-triangular form for matrix A – diagonal if A is hermitian. QR algorithm itself is motivated by power iteration; just like inverse iteration can use Rayleigh quotient to for acceleration, same goes for QR algorithm: for step k, we pick a shift $\mu^{(k)}$, then perform QR factorization $Q^{(k)}R^{(k)} = A^{(k-1)} - \mu^{(k)}I$, and recombine factors by $A^{(k)} = R^{(k)}Q^{(k)} + \mu^{(k)}I$. For the choice of shifts, we can choose Rayleigh quotient shift $\mu^{(k)} = R^{(k-1)}_{m,m}$, or Wilkinson's shift, etc.

There are quite a few other algorithms as well; e.g., one might crop the matrix $A^{(k)}$ when $A^{(k)}_{m,m-1}$ becomes close enough to zero, and it turns out to accelerate convergence.

IV Iterative Methods

23 Iterative Methods Basics Let X be a Banach space and $U \subset X$ be a complete subspace, then $\phi : U \mapsto U$ is called *Lipschitz continuous* if $\exists \rho \geq 0$ s.t. $\left\|\phi(x) - \phi(y)\right\| \leq \rho \left\|x - y\right\|$, $\forall x, y \in U$; furthermore, ϕ is called *non-expansive* if $\rho \leq 1$, and called $\rho-$contractive if $\rho < 1$. As a theorem to characterize $\rho-$contraction: Let $U \subset X$ be closed, $\phi : U \mapsto U$ be a $\rho-$contractive mapping, then

- $\exists x \in U$ s.t. $x = \phi(x)$, and x is unique
- $\forall x_0 \in U$, $x_{k+1} = \phi(x_k)$ for $k = 0, 1, \ldots$ converges to x, and

$$\begin{aligned} \|x_{k+1} - x\| &\leq \rho \|x_k - x\| \\ \|x_k - x\| &\leq \frac{\rho^k}{1-\rho} \|x_0 - x\| \\ \|x_k - x\| &\leq \frac{\rho}{1-\rho} \|x_k - x_{k-1}\| \end{aligned}$$

we just show the existence and uniqueness of such x in the first point, the rest will be trivial. Note that $\forall n \geq m \geq 1$ and $\forall x_0 \in U$, $\exists \rho < 1$ s.t.

$$\begin{aligned} \|x_n - x_m\| &= \left\|\phi^n(x_0) - \phi^m(x_0)\right\| \\ &\leq \rho^m \left\|\phi^{n-m}(x_0) - x_0\right\| \\ &\leq \rho^m \left(\left\|\phi^{n-m}(x_0) - \phi^{n-m-1}(x_0)\right\| + \left\|\phi^{n-m-1}(x_0) - \phi^{n-m-2}(x_0)\right\| + \cdots + \left\|\phi(x_0) - x_0\right\| \right) \\ &\leq \rho^m \sum_{i=0}^{n-m-1} \rho^k \left\|\phi(x_0) - x_0\right\| \\ &\leq \rho^m \sum_{i=0}^{\infty} \rho^k \left\|\phi(x_0) - x_0\right\| \\ &= \frac{\rho^m}{1-\rho} \left\|\phi(x_0) - x_0\right\| \end{aligned}$$

which implies that $\{x_k\}$ is Cauchy. Now let $x \in U$ denote the limit of $\{x_k\}$ as $n \to \infty$, then x is a fixed point because

$$\phi(x) = \phi\left(\lim_{k\to\infty} x_k\right) = \lim_{k\to\infty} \phi(x_k) = \lim_{k\to\infty} x_{k+1} = x$$

and uniqueness of x can be proved by assuming x, y are both fixed points, then

$$0 \le \|x - y\| = \|\phi(x) - \phi(y)\| \le \rho \|x - y\| < \|x - y\|$$

which implies $x = y$.

Here are a few examples of fixed point methods: (i). $\phi(x) = Tx + c$, where T is called *iteration matrix*, contraction of T is equivalent to $\|T\| = \sigma_{\max}(T) \le 1$. (ii). *Richardson method*: $\phi(x) = x + \omega(b - Ax)$, where $b - Ax$ is called *residual*. View Richardson's methods in an iteration manner, we have $\phi(x) = (I - \omega A)x + \omega b$; further suppose $A > 0$, with eigenvalues $0 < \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$, then $I - \omega A$ has eigenvalues $1 - \omega\lambda_1, 1 - \omega\lambda_2, \ldots, 1 - \omega\lambda_n$, then $\|I - \omega A\| = \max_k |1 - \omega\lambda_k| = \max\{|1 - \omega\lambda_1|, |1 - \omega\lambda_n|\}$, which yields the optimal $\omega^* = \frac{2}{\lambda_1 + \lambda_n}$. As a remark to Richardson's method, people usually left-multiply *pre-conditioner* P^{-1} to the linear system, resulting in $P^{-1}Ax = P^{-1}b$ – the purpose here is to make $\mathrm{cond}(A)$ small.

24 Arnoldi Iteration Most of the iterative methods discussed here will be based on the idea of projecting into Krylov subspaces. Given matrix $A \in \mathbb{C}^{m\times m}$ and vector b, *Krylov sequence* is the set of vectors $b, Ab, A^2b, \ldots$ (they are *not* necessarily linearly independent), and the corresponding *Krylov subspaces* of order r is then the space spanned by the first r terms of Krylov sequence. *Arnoldi iteration* can be interpreted as performing (modified) Gram-Schmidt on the Krylov matrix

$$K_n := \begin{bmatrix} b & Ab & A^2b & \cdots & A^{n-1}b \end{bmatrix} \in \mathbb{C}^{m\times n}$$

to construct its orthonormal basis. In view of A itself, Arnoldi iteration can be considered as an Hessenbergized method analogous to Gram-Schmidt, see Table 1 – one similarity is, it can stop at any step, therefore serves as a better iterative method.

Table 1: Householder's reflection vs Gram-Schmidt/Arnoldi process

	QR factorization $A = QR$	Hessenberg formation $A = QHQ^*$
Householder's reflection	orthogonal triangularization	orthogonal Hessenberg-ition
Gram-Schmidt/Arnoldi process	triangular orthogonalization	Hessenbergized orthogonalization

For iterative methods we consider m to be large or infinite, so only consider the first n columns of $AQ = QH$. Let $Q_n \in \mathbb{C}^{m\times n}$ denote the first n columns of Q; and let $\tilde{H}_n \in \mathbb{C}^{(n+1)\times n}$ be the submatrix located at the upper-left corner of H, which will also be a Hessenberg matrix itself. Then we'll have $AQ_n = Q_{n+1}\tilde{H}_n$ as the first n columns of $AQ = QH$. *And equating the nth column of both sides gives us* $Aq_n = h_{1n}q_1 + \cdots + h_{nn}q_n + h_{n+1,n}q_{n+1}$, *a recurrence relation for* q_{n+1} *– Arnoldi iteration follows directly on this recurrence relation: let* $q_1 = \frac{b}{\|b\|}$ *be the initializer, and choose* h_{kn} *s.t.* $h_{kn}q_k$ *is a projection of* q_k *on* Aq_n *for* $k = 1, 2, \ldots, n$; *as an interpretation, the updating step first subtracts the projections of built orthogonal bases from* Aq_n, *then normalizing* q_{n+1} *with* $h_{n+1,n}$. Because the recurrence formula states that each q_{n+1} is formed by a linear combination of Aq_n and $q_1, q_2, \ldots, q_{n-1}$, each q_n is therefore a degree-$(n-1)$ polynomial of A times b; hence $q_1, q_2, \ldots, q_n$ form an orthonormal basis for the Krylov subspace

$$\mathcal{K}_n := \left\langle b, Ab, \ldots, A^{n-1}b \right\rangle$$

(i). In this view, Arnoldi process can be considered as systematic construction of orthonormal bases for successive Krylov subspaces. Because Arnoldi iteration constructs orthonormal basis in a Gram-Schmidt manner, the Q_n here will be exactly same as the Q_n present in the Gram-Schmidt QR factorization of K_n, while K_n and R are never implicitly constructed. And it's called *modified* Gram-Schmidt because at iteration k, we subtract projections of constructed bases $q_1, q_2, \ldots, q_k$ from the vector Aq_k instead of the "original" vector $A^k b$.

(ii). Another view of Arnoldi process is a computation of projections onto successive Krylov subspaces. Note that $Q_n^* Q_{n+1}$ is a $n \times (n+1)$ matrix with 1 on diagonal and 0 elsewhere; then from $AQ_n = Q_{n+1}\tilde{H}_n$ we have

$$\underbrace{Q_n^* Q_{n+1}\tilde{H}_n}_{=:H_n} = Q_n^* A Q_n$$

apparently, H_n here will be the $n \times n$ submatrix located at the upper-left corner of H. This is an analogue to a change of basis, with Q_n not orthogonal but of shape $m \times n$ – and the resulting interpretation is: given some $v \in \mathcal{K}_n$, applying A to it, then orthogonally project Av back to $\mathcal{K}_n$.

Note that H_n and A are *pseudo*-similar. Intuitively, one might then consider the eigenvalues of H_n as estimates for the eigenvalues of A – for this reason, they are called *Arnoldi eigenvalue estimates* (at step n) or *Ritz values* (wrt. $\mathcal{K}_n$).

Consider a vector $x \in \mathcal{K}_n$, such a vector can then be written as a linear combination of Krylov's vectors $b, Ab, \ldots, A^{n-1}b$, put in polynomial form, it will be

$$x = q(A)\,b$$

Now consider $P^n \coloneqq \{\text{monic polynomials of degree } n\}$, the famous Arnoldi-Lanczos approximation problem is proposed as

$$\min_{p^n \in P^n} \|p^n(A)\,b\|$$

and the Arnoldi iteration solves this problem exactly (if it doesn't break down ofc...) – *the minimizer $\bar{p}^n$ is uniquely given by the characteristic polynomial of H_n.* As a proof, write $p^n(A)\,b = A^n b - y$ for some $y \in \mathcal{K}_n$; i.e., the problem is to minimize the distance from $A^n b$ to $\mathcal{K}_n$ – thus minimization can be characterized by $p^n(A)\,b \perp \mathcal{K}_n \Leftrightarrow Q_n^* p^n(A)\,b = 0$ as $q_1, q_2, \ldots, q_n$ are a basis of $\mathcal{K}_n$. Now consider $A = QHQ^*$; where $Q \coloneqq \begin{bmatrix} Q_n & U \end{bmatrix}$ s.t. Q is a orthogonal matrix extended from Q_n, and $H \coloneqq \begin{bmatrix} H_n & X_2 \\ X_1 & X_3 \end{bmatrix}$, where the entries of X_1 is all 0 besides its upper-right entry and X_3 is Hessenberg due to the Hessenberg structure of H. Then we have

$$\begin{aligned} & Q_n^* p^n(A)\,b = 0 \\ \Leftrightarrow & Q_n^* Q p^n(H)\,Q^* b = 0 \\ \Leftrightarrow & \begin{bmatrix} I_n & 0 \end{bmatrix} p^n(H)\,e_1 \|b\| = 0 \end{aligned} \tag{4}$$

and (4) follows from $q_1 = \frac{b}{\|b\|}$. The interpretation of last equation is, the minimization characterization now becomes that the first n entries in the first column of $p^n(H)$ are 0. Due to the Hessenberg structure of H, the first n entries in the first column of $p^n(H)$ are exactly the first column of $p^n(H_n)$ – in view of this, it is *sufficient* to make $p^n(H_n) = 0$: by Cayley-Hamilton theorem, if p^n is the characteristic polynomial of H_n, $p^n(H_n) = 0$. Proof of uniqueness uses contradiction: if uniqueness is voided, taking difference of two distinct degree-n monic polynomials that both minimize $\|p^n(A)\,b\|$ will then result in a non-zero polynomial $q(A)$ of degree $\leq n-1$ s.t. $q(A)\,b = 0$ – this contradicts the assumption that K_n is of full-rank.

Based on this finding, *(iii). the Ritz values generated by Arnoldi iteration are the roots of the optimal polynomial to the Arnoldi-Lanczos approximation problem.* And this gives the Ritz values some invariant properties:

- (Translation-invariance) If A is changed to $A+\sigma I$ for some $\sigma \in \mathbb{C}$, and b is left unchanged, the Ritz values $\left\{\theta_j\right\}$ at each step will be changed to $\left\{\theta_j+\sigma\right\}$
- (Scale-invariance) If A is changed to σA for some $\sigma \in \mathbb{C}$, and b is left unchanged, the Ritz values $\left\{\theta_j\right\}$ at each step will be changed to $\left\{\sigma\theta_j\right\}$
- (Unitary-similarity-transformation invariance) If A is changed to UAU^* for some unitary U, and b is changed to Ub, the Ritz values do not change

25 GMRES and CG *Generalized minimal residuals (GMRES)* is a method using Arnoldi iteration to solve a linear system $Ax=b$, the resulting mechanic is to use $x_n \in \mathcal{K}_n$ at step n to approximate the root by formulating the problem

$$\begin{aligned}
&\min_{x_n\in\mathcal{K}_n} \|Ax_n-b\| \\
\Leftrightarrow &\min_{c\in\mathbb{C}^n} \|AK_nc-b\| \\
\Leftrightarrow &\min_{y\in\mathbb{C}^n} \left\|AQ_ny-b\right\| \\
\Leftrightarrow &\min_{y\in\mathbb{C}^n} \left\|Q_{n+1}\tilde{H}_ny-b\right\| \\
\Leftrightarrow &\min_{y\in\mathbb{C}^n} \left\|\tilde{H}_ny-Q_{n+1}^*b\right\| &(5)\\
\Leftrightarrow &\min_{y\in\mathbb{C}^n} \left\|\tilde{H}_ny-\|b\|e_1\right\| &(6)
\end{aligned}$$

where (5) is because that b is in the column space of Q_{n+1} (because $q_1 \coloneqq \frac{b}{\|b\|}$), therefore left multiplication of Q_{n+1}^* does not change the norm. Furthermore, note that $Q_{n+1}^*b=\|b\|e_1$, which gives us (6).

On another note, the initial assumption for GMRES of $x_n \in \mathcal{K}_n$ is equivalent to $x_n=q_n(A)b$ for some degree-$(n-1)$ polynomial q_n, with coefficients being c mentioned in above equations. Then the residual $b-Ax_n=(I-Aq_n(A))b$; let $p_n(z)\coloneqq 1-zq(z)$, we therefore aim to solve problem $\min_{p_n\in P_n}\|p_n(A)b\|$ for GMRES, but with $P_n \coloneqq \{\text{degree } n \text{ polynomials } p \text{ with } p(0)=1\}$ here.

If A is hermitian, the Arnoldi iteration will be redundant to find eigenvalues of A – a method called Lanczos iteration was introduced as a simplification of Arnoldi iteration (*mainly simplified by noting that H_n becomes tri-diagonal now*). With same simplification idea, if A is positive definite, solving $\min_x\|Ax-b\|$ using GMRES will in fact not be efficient – *conjugate gradient (CG)* was introduced based on *minimizing the A-norm of the error*; where the *A–norm of x* defined as x^TAx.

V Quadrature and ODE Solvers

26 Numerical Integration Let $f : \mathbb{R} \mapsto \mathbb{R}$ be a second-order smooth function, i.e., $f \in \mathcal{C}^2$, numerical integration aims to approximate the integral $I(f) \coloneqq \int_a^b f(x)\,dx$. *Midpoint method* uses the midpoint $c \coloneqq \frac{a+b}{2}$: based on Taylor theorem $f(x) = f(c) + f'(c)(x-c) + O\left(h^2\right)$, where $h \coloneqq b-a$; then the midpoint approximation will be $\tilde{I}(f) = (b-a) f\left(\frac{a+b}{2}\right) + O\left(h^3\right)$. *Composite midpoint formula* improved by dividing $[a,b]$ into n intervals, each of length $h \coloneqq \frac{b-a}{n}$; i.e., consider $n-1$ "midpoint" interpolations $a < x_1 < x_2 < \cdots < x_n < b$; as a result, the error is now $n \cdot O\left(h^3\right) = O\left(n^{-2}\right) = O\left(h^2\right)$.

Now consider when $f \in \mathcal{C}^1$, we are to perform analysis of convergence for the midpoint rule, called *Lebesgue-type analysis*. Now consider the integral and approximate-integral operators I and $\tilde{I} : C([a,b]) \mapsto \mathbb{R}$; it's trivial that they are both linear operators. The *degree of exactness* measures to which degree of polynomial p, $\tilde{I}(p)$ approximates $I(p)$ exactly; e.g., for midpoint method, we have $\tilde{I}(p) = I(p)$ for $p \in P_1$, then the degree of exactness is 1. As a normal strategy in numerical analysis, we analyze error by using polynomials:

$$\begin{aligned} \tilde{I}(f) - I(f) &= \tilde{I}(f) - \tilde{I}(p) + I(p) - I(f) \\ &= \tilde{I}(f-p) + I(p-f) \\ &= \left(\tilde{I} - I\right)(f-p) \end{aligned} \tag{7}$$

Now consider the norm of I, defined by $\|I\| = \sup_f \frac{|I(f)|}{\|f\|_\infty}$:

$$|I(f)| = \left|\int_a^b f(x)\,dx\right| \le (b-a)\|f\|_\infty \Rightarrow \frac{|I(f)|}{\|f\|_\infty} \le b-a$$

$$\forall f \text{ constant function}, \left|\int_a^b f(x)\,dx\right| = (b-a)\|f\|_\infty \Rightarrow \sup_f \frac{|I(f)|}{\|f\|_\infty} \ge b-a$$

therefore, $\|I\| = b-a$; same goes for $\tilde{I}$, we'll have $\left\|\tilde{I}\right\| = b-a$. And from (7),

$$\begin{aligned} &\left|\tilde{I}(f) - I(f)\right| \le \left|\tilde{I}(f-p)\right| + |I(p-f)| \le 2(b-a)\|f-p\|_\infty,\ \forall p \in P_1 \\ \Rightarrow &\left|\tilde{I}(f) - I(f)\right| \le 2(b-a) \inf_{p \in P_1} \|f-p\|_\infty \end{aligned}$$

this analysis essentially says without assumption $f \in \mathcal{C}_2$, the quadrature error can be deduced from p – making p to be the minimax polynomial of f can give most meaningful results (though not always easy). Note that p here is merely a tool introduced to perform error analysis.

In general, quadrature proceeds as

$$\tilde{I}(f) = \sum_{i=0}^{n} w_i f(x_i),\ a \le x_i \le b$$

and the sequence $x_1, x_2, \ldots, x_n$ are called *quadrature nodes*, while $w_1, w_2, \ldots, w_n$ are called *quadrature weights*. An example could be *interpolatory quadrature*, which proceeds by computing the Lagrangian interpolation polynomial first, then taking integral of the polynomial:

$$\begin{aligned}\tilde{I}(f) = I(\mathcal{L}_n f) &= \int_a^b \mathcal{L}_n f(x)\,dx \\ &= \int_a^b \sum_{i=0}^{n} f(x_i)\,\phi_i(x)\,dx \\ &= \sum_{i=0}^{n} f(x_i) \int_a^b \phi_i(x)\,dx\end{aligned}$$

and the degree of exactness here will be at least n – as Lagrangian interpolation can represent a degree-n polynomial exactly. The rest of error analysis will proceed similar as we've seen before for midpoint quadrature.

27 (first-order[6]) IVP The *initial value problem (IVP)*, otherwisely called the *Cauchy problem*, basically refers to a DE with initial values given; specifically, it is defined as finding a real-valued function $y \in C^1(I)$ s.t.

$$\begin{cases} y'(t) = f(t, y(t)), & t \in I \\ y(t_0) = y_0 \end{cases} \tag{8}$$

where $f(t, y)$ is a given real-valued function in the strip $S = I \times \mathbb{R}$. Immediately from this definition, we have the following equivalent of above formula:

$$y(t) - y_0 = \int_{t_0}^{t} f(\tau, y(\tau))\,d\tau$$

And recall from ODE that the existence and uniqueness results are:

1. *Local*: if $f(t, y)$ is locally L–Lipschitz continuous at (t_0, y_0) wrt. y, with the neighbor-

[6]Note that in numerical analysis, *first-order IVP* is of special interest because higher-order IVP can always be rewritten as a first-order IVP; e.g. see this example of five-body motion problem (Problem 4) solved by Runge-Kutta.

hood J, Σ width to be r_J for t_0 and r_Σ for y_0; then the IVP admits a unique solution in a neighborhood of t_0 with radius $0 < r_0 < \min\left\{r_J, \frac{r_\Sigma}{\max_{t,y\in J\times\Sigma}|f(t,y)|}, \frac{1}{L}\right\}$;

2. *Global*: if f is uniformly Lipschitz continuous wrt. y over the entire domain $I \times \mathbb{R}$.

And in view of stability analysis of IVP, the definition follows from the idea that *small perturbations results in small error to solutions*. Formally, the IVP is *Liapunov stable on I* if $\forall\,(\delta_0, \delta(t))$ satisfying

$$|\delta_0| < \varepsilon,\ |\delta(t)| < \varepsilon,\ \forall t \in I,$$

with $\varepsilon > 0$ sufficiently small to ensure existence of solution for the perturbed problem

$$\begin{cases} z'(t) = f(t, z(t)) + \delta(t), & t \in I \\ z(t_0) = y_0 + \delta_0, \end{cases}$$

we have $\exists C > 0$ s.t. $\left|y(t) - z(t)\right| < C\varepsilon,\ \forall t \in I$ – *the constant C here depends on* t_0, y_0, f, *but not on* ε.

28 Numerical Methods for ODE The approximation of the problem (8) is called *one-step* method if $\forall n \geq 0$, u_{n+1} depends only on u_n; otherwise, it will be called *multistep* method. Moreover, if u_{n+1} depends only on the past q steps (i.e., $u_n, \ldots, u_{n+2-q}, u_{n+1-q}$). A few one-step methods are:

- forward Euler: $u_{n+1} \leftarrow u_n + hf_n$
- backward Euler: $u_{n+1} \leftarrow u_n + hf_{n+1}$
- Crank-Nicolson method: $u_{n+1} \leftarrow h \cdot \frac{f_n + f_{n+1}}{2}$
- Heun method: $u_{n+1} \leftarrow u_n + h \cdot \frac{f_n + f(t_{n+1}, u_n + hf_n)}{2}$

A method is called *explicit* if u_{n+1} can be computed directly from u_k, for some $k \leq n$ (e.g., forward Euler and Heun); *implicit* if u_{n+1} depends implicitly on itself through f (e.g., backward Euler and Crank-Nicolson).

The *linear multistep method* refers to

$$u_n \leftarrow \sum_{j=1}^{r} a_j u_{n-j} + h\left(\sum_{j=1}^{r} b_j f_{n-j} + b_0 f_n\right),\ \forall n = r, r+1, \ldots$$

or write in a more interpretable manner: the updating formula above aims to build a connection between numerical values and slopes:

$$u_n - \sum_{j=1}^{r} a_j u_{n-j} = h\left(\sum_{j=1}^{r} b_j f_{n-j} + \underbrace{b_0 f_n}_{\text{implicit term}}\right)$$

Linear multistep method is, of course, linear in both u. and f., and it evaluates f. once per step; its accuracy can be increased by increasing the number of steps r. For the sake of clarity, we shall re-parameterize the equation above, denoted by:

$$\sum_{j=0}^{r} \alpha_j u_{k+j} = h\left(\sum_{j=0}^{r} \beta_j f_{k+j}\right)$$

where now $k := n - r$. Thanks to its simple linear structure, error analysis is feasible for linear multistep method – first let's define the local truncation error for linear multistep method τ_h:

$$h\tau_h := y_{k+r} - u_{k+r}$$

and the error analysis will just assume previous r steps are exact, i.e., $u_{k+j} = y_{k+j}$ for $j = 0, 1, \ldots, r-1$; then, the error analysis can be achieved by applying the updating formula equivalent to this assumption:

$$\begin{aligned}\alpha_r u_{k+r} + \sum_{j=0}^{r-1} \alpha_j y_{k+j} &= h\left(\beta_r f_{k+r} + \sum_{j=0}^{r-1} \beta_j f\left(t_{k+j}, y_{k+j}\right)\right) \\ &= h\left(\beta_r f_{k+r} + \sum_{j=0}^{r-1} \beta_j y'_{k+j}\right)\end{aligned}$$

and taking difference term-wisely, we have:

$$\begin{aligned} L_h &:= \sum_{j=0}^{r}\left[\alpha_j y_{k+j} - h\beta_j y'_{k+j}\right] \\ &= \alpha_r\left(y_{k+r} - u_{k+r}\right) - h\beta_r\left(f\left(t_{k+r}, y_{k+r}\right) - f_{k+r}\right) \\ &= \alpha_r\left(y_{k+r} - u_{k+r}\right) - h\beta_r \underbrace{\left[\frac{\partial f}{\partial u}(\xi)\right]}_{=:J\in\mathbb{R}^{m\times m}}\left(y_{k+r} - u_{k+r}\right) \end{aligned} \tag{9}$$

where m is just the dimension of $u_\cdot$ or $y_\cdot$, and the symbol "ξ" comes from mean value theorem. Moreover, by definition, now we have

$$h\tau_n(h) = (\alpha_0 I - h\beta_0 J)^{-1} L_h$$

The linear multistep method is *of order* p if $\tau_n(h) = O(h^p) \Leftrightarrow L_h = O\left(h^{p+1}\right)$. And from above steps, we are ready to prove that the following statements are equivalent:

1. linear multistep method is of order p
2. $\sum \alpha_j = 0, \sum j\alpha_j = \sum \beta_j, \sum j^2\alpha_j = 2\sum j\beta_j, \cdots, \sum j^p\alpha_j = p\sum j^{p-1}\beta_j$
3. $\rho\left(e^h\right) - h\cdot\sigma\left(e^h\right) = O\left(h^{p+1}\right)$ as $h \to 0$, where the characteristic and generating polynomials are defined by

$$\rho(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_r z^r$$
$$\sigma(z) = \beta_0 + \beta_1 z + \cdots + \beta_r z^r$$

4. $\frac{\rho(z)}{\log z} - \sigma(z) = O\left((z-1)^p\right)$ as $z \to 1$

The proof of equivalence between 2 and 3 follows by taking $u(t) = e^t$ we have

$$L_h = \sum_{j=0}^{r}\left[\alpha_j e^{t+jh} - h\beta_j e^{t+jh}\right] = e^t\left[\rho\left(e^h\right) - h\cdot\sigma\left(e^h\right)\right].$$

and other part of proof follows by taking second-order Taylor polynomial expansion on (9).

The *Runge-Kutta method* refers to

$$u_{n+1} \leftarrow u_n + h\cdot F(t_n, u_n, h; f),\ \forall n \geq 0$$

where F is the increment function defined as:

$$F(t_n, u_n, h; f) = b^T K,$$
$$[K]_i = f\left(t_n + [c]_i h, u_n + hA_{i,:}K\right),\ i = 1, 2, \ldots, s$$

where $A \coloneqq \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1s} \\ a_{21} & a_{22} & & a_{2s} \\ \vdots & & \ddots & \vdots \\ a_{s1} & a_{s2} & \cdots & a_{ss} \end{bmatrix} \in \mathbb{R}^{s\times s}$, $b, c \in \mathbb{R}^s$; and we assume $c_i = \sum_{j=1}^{s} a_{ij}$, $\forall i = 1, 2, \ldots, s$.

Obviously Runge-Kutta is explicit iff A is a strictly lower triangular matrix. *Runge-Kutta method is a one-step method, and its accuracy can be increased by increasing number of function evaluations s*. It's straightforward to verify that Euler's methods belong to the Runge-Kutta family.

References

Tsogtgerel, Gantumur (2020). *MATH 598 Lecture Notes, Fall 2020*.

Trefethen, Lloyd (1997). *Numerical linear algebra*. Philadelphia, Pa: Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104. ISBN: 0898719577.

Quarteroni, Alfio M., Riccardo Sacco, and Fausto Saleri (Oct. 19, 2006). *Numerical Mathematics*. Springer-Verlag GmbH. ISBN: 3540346589. URL: https://www.ebook.de/de/product/5810163/alfio_m_quarteroni_riccardo_sacco_fausto_saleri_numerical_mathematics.html.

Trefethen, Lloyd N. (Jan. 3, 2013). *Approximation Theory and Approximation Practice*. CAMBRIDGE. 295 pp. ISBN: 1611972396. URL: https://www.ebook.de/de/product/20339745/lloyd_n_trefethen_approximation_theory_and_approximation_practice.html.

Jiao, Xiangmin (2012). *Lecture 13: Householder Reflectors;Updating QR Factorization*. URL: http://www.ams.sunysb.edu/~jiao/teaching/ams526_fall12/lectures/lecture13.pdf.