MAT 226A: Numerical Methods Final Review

Adam Rose

Contents

1 Introduction

There are many examples of numerical computations that force you to rethink and reformulate problems, e.g. computing eigenvalues, or finding the roots of a high degree polynomial. The way we do these analytically are usually not the best way to do them numerically.

For instance, when computing eigenvalues of a matrix A, we compute the **Schur decomposition**

$$A = QTQ^*,$$

where T is triangular and Q is unitary.

2 Floating Point Numbers

Our computers store a number $x \in \mathbb{R}$ such that

$$x = \pm(1+q) \times 2^{c-1023},$$

with the **mantissa** $(1+q)$, and the **characteristic** $(c-1023)$. This allows us to store numbers with a range of 10^{-308} to 10^{308}.

Note, for $x \in \mathbb{R}$, take x_- and x_+ as the two closes floating point numbers. Then

$$|fl(x) - x| \leq |x_+ - x_-|,$$

if there is **no rounding**. With rounding, the bound is half of that. For the following, we introduce $\boxed{\varepsilon_{\text{mach}} \approx 10^{-16}}$.

Theorem 1. *For all $x \in \mathbb{R}$ (in some range), there exists an $x' \in \mathbb{F}$ and an ε with $|\varepsilon| < \varepsilon_{mach}$ such that*

$$\begin{aligned} &1)\ |x - x'| \leq \varepsilon_{mach}|x|, \\ &2)\ fl(x) = x(1+\varepsilon). \end{aligned}$$

Theorem 2. *Let $\bullet$ denote floating point addition, subtraction, multiplication, and division. Then for all $x, y \in \mathbb{F}$, there exists ε with $|\varepsilon| < \varepsilon_{mach}$ such that*

$$x \bullet y = (x \cdot y)(1+\varepsilon).$$

3 Conditioning and Stability

In this section, we explore the conditioning and stability of problems and algorithms to solve problems.

- **Conditioning** is related to how sensitive a *problem* is to perturbations.
- **Stability** is related to the error in an *algorithm* to solve a problem.

3.1 Conditioning

The condition number $\kappa(x)$ of a problem f , given a perturbation in x, must satisfy

$$\frac{||\delta f||}{||f||} \leq \kappa(x) \frac{||\delta x||}{||x||}.$$

We can further refine this.

Definition 1. *A **relative condition number** is*

$$\kappa(x) = \lim_{\delta \to 0} \sup_{||\delta x|| < \delta} \frac{\frac{||\delta f||}{||f||}}{\frac{||\delta x||}{||x||}}.$$

Furthermore, if f is differentiable,

$$\kappa(x) = \frac{||J(x)|| \cdot ||x||}{||f(x)||}.$$

Definition 2. *A **absolute condition number** is*

$$\hat{\kappa}(x) = \lim_{\delta \to 0} \sup_{||\delta x|| < \delta} \frac{||\delta f||}{||\delta x||}.$$

If f is differentiable, then

$$\hat{\kappa}(x) = ||J(x)||.$$

3.2 Stability

We begin with a definition of stability.

Definition 3. *An algorithm $\tilde{f} : X \to Y$ is* ***stable*** *if for each $x \in X$, there exist $\tilde{x} \in X$ such that*

$$1)\ \frac{||\tilde{f}(x) - f(\tilde{x})||}{||f(\tilde{x})||} = \mathcal{O}(\varepsilon_{mach})$$
$$2)\ \frac{||x - \tilde{x}||}{||x||} = \mathcal{O}(\varepsilon_{mach})$$

The drawback to this version of stability is it only requires *nearly* the right answer to *nearly* the right question. So we introduce a strongle version of stability.

Definition 4. *An algorithm $\tilde{f} : X \to Y$ is* ***backwards stable*** *if for each $x \in X$, there exists $\tilde{x} \in X$ such that*

$$1)\ \tilde{f}(x) = f(\tilde{x})$$
$$2)\ \frac{||x - \tilde{x}||}{||x||} = \mathcal{O}(\varepsilon_{mach})$$

Thus backwards stability provides *exactly* the right solution to *nearly* the right question. Here are some examples.

Example 1. *(a) Floating point arithmetic is backwards stable.*

(b) Adding a fixed number is not backwards stable, i.e. $\tilde{f}(x) = fl(x) \oplus 1$ is not backwards stable.

4 Linear Systems of Equations

We first learn to solve linear systems. We will then use the knowledge to linearize nonlinear systems and solve them.

4.1 Matrix Norms

We will introduce the most commonly used norms.

Definition 5. *Let $A \in \mathbb{C}^{m \times n}$. Then*

$$1)\ ||A||_1 = \max_{1 \le j \le n} ||\underline{a}_j||_1, \text{ where } \underline{a}_j \text{ is the } j\text{-th column of } A$$
$$2)\ ||A||_2 = \rho(A^* A)^{1/2} = \sup_{||x||_2 = 1} ||Ax||_2$$
$$3) ||A||_\infty = \max_{1 \le i \le m} ||\underline{a}_i^*||_1, \text{ where } \underline{a}_i^* \text{ is the } i\text{-th row of } A$$

Definition 6. *Note that in class, we reviewed that if $A \in \mathbb{C}^{m \times n}$ and $A^* = A$, or if A is **Orthogonal**, we have*

1)*A has real eigenvalues,*
2) *Eigenvectors of A are orthogonal.*

Furthermore, $A = Q^* \Lambda Q$, where Q has eigenvecotors of A as columns, and Λ is a diagonalmatrix of eigenvalues of$A$.

Definition 7. *The **condition number of a matrix** $A \in \mathbb{C}^{m \times n}$ is*

$$\kappa(A) = \frac{||A|| \cdot ||x||}{||Ax||}.$$

If the matrixis invertible, then

$$\kappa(A) = ||A|| \cdot ||A^{-1}||.$$

If $A^* = A$, then

$$\kappa(A) = \frac{\max_j |\lambda_j|}{\min_j |\lambda_j|},$$

or largest singular value of A over the smallest singluar value of$A$ (in case A is not invertible).

4.2 Gaussian Elimination

To solve, $A\underline{x} = \underline{b}$ with A invertible, we use Gaussian Elimination, or row-reduction. The idea is to factor $A = LU$, where L is lower triangular and U is upper triangular.

However, the most common way of producing these matrices is an unstable algorithm. From here, we introduce the idea of **pivots**.

Definition 8. ***Partial Pivoting** is the process of selecting the largest magnitude element in a column below the diagonal, and swapping rows so it is in the diagonal.*

There is also full pivoting, but it takes much more work so we don't generally use it. The idea of a pivot is swapping rows to get the"pivot" onto the diagonal.

The only issue with this concept is that when we swap rows, we arent actually working with A anymore. We end up finding the factorization for a permuation of A, i.e.

$$L_nP_nL_{n-1}P_{n-1}\cdots L_2P_2L_1P_1A = U,$$

Where the $P_i's$ are permuation matrices and the $L_i's$ are lower triangular matrices. It is shown in class that in fact we can simplify to

$$PA = LU,$$

with P being a single permutation matrix.

Theorem 3. ***NO PIVOTING:*** *Let A be a non-singular matrix and $A = LU$ be computed without pivoting. If the algorithm completes, then computed $\tilde{L}$, $\tilde{U}$ satisfy*

$$\tilde{L}\tilde{U} = A + \delta A,\ \frac{||\delta A||}{||L||\cdot||U||} = \mathcal{O}(\varepsilon_{mach}).$$

for some δA. NOTE: If $\frac{||\delta A||}{||A||} = \mathcal{O}(\varepsilon_{mach})$, then LU is backwards stable.

Theorem 4. ***WITH PIVOTING:*** *Let A be a non-singular matrix and $A = LU$ be computed with pivoting. If the algorithm completes, then computed $\tilde{L}$, $\tilde{U}$ satisfy*

$$\tilde{L}\tilde{U} = \tilde{P}A + \delta A,\ \frac{||\delta A||}{||A||} = \mathcal{O}(\rho\varepsilon_{mach}).$$

*for some δA. Here, $\rho = \frac{\max_{i,j}|u_{uj}|}{\max_{i,j}|a_{ij}|}$ is the **growth factor** of A. NOTE: If $\rho = \mathcal{O}(1)$, then GE with partial pivoting is backwards stable.*

5 Nonlinear Equations

The first method to solve nonliear equations we will introduce is the **Bisection Method**. This method can be used to solve problem of the form $f : D \subset \mathbb{R} \to \mathbb{R}$, for $x \in D$ such that $f(x) = 0$. The idea is to check the signs of our ouput and cut our interval in half to narrow down where we check next. This method is **globally convergent** with a linear rate of convergence, and each step cuts down error by a factor of 2. However, it doesn't generalize to higher dimensions.

5.1 Newton's Method

Newton's method uses a linear approximation to find the root of a problem. For a finding the real-valued solution x_* such that $f(x_*) = 0$, the idea is to define a recursive formula that will converge. Once you pick x_0, define

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

We know this converges if the initial guess x_0 with within a small enough neighborhood of x_*. Newton's method converges faster than Bisection method.

Generalizing Newton's method to **higher dimensions**, Set

$$x_{k+1} = x_k - J^{-1}(x_k)F(x_k).$$

Theorem 5. *Let $f : D \to \mathbb{R}$ for an open interval D. Suppose f' is Lipschitz on D with constant γ and $|f'(x)| \geq \rho > 0$. If $f(x_*) = 0$, for some $x_* \in D$, then there exists some ν such that if $|x_0 - x| < \nu$, then Newton's method converges to x_* and*

$$|x_{k+1} - x_*| \leq \frac{\gamma}{2\rho}|x_k - x_*|^2.$$

We can also add **Line Searching** to our routine, where we check to see if first $\lambda = 1$, $\alpha \in (0, 1)$, and

$$||F(x_k - \lambda J^{-1}(x_k)F(x_k))|| \geq (1 - \alpha\lambda)||F(x_k)||.$$

If it is, cut λ in half. We terminate when the norm of F gets small enough, or $x_k's$ starts changing below a certian tolerance.

5.2 Least Squares

An intoduction to least squares problem. The idea is to approximate thge solution to a problem $A\underline{x} = \underline{b}$ as best as possible. Set

$$r = \underline{b} - A\underline{x}$$

to be the **residual**. Our goal is to minimize $||r(\underline{x})||_2$, i.e. minimize the 2-norm. From here, we find the equations to satisfy are the **Normal Equations:**

$$A^* A\underline{x} = A^*\underline{b}.$$

For $f \in C([a, b])$, set $p_n(x) = \sum_{k=0}^{n} c_k \phi_k(x)^k$, where $\{\phi_k\}$ are a basis. Then

$$c_k = 2\frac{\langle f, \phi_k\rangle}{\langle \phi_k, \phi_k\rangle}.$$

6 QR Factorization

Often used to solve the linear least squares method introduced above, QR factorization is the factoring of a matrix $A = QR$ where Q is orthogonal (orthogonal columns) and R is upper triangular. There are several different methods.

1) Gram-Schmidt Process,
We're given a basis $\{x_1, x_2, ...\}$ we can make an ONB by setting $u_k = x_k - \sum_{j=1}^{k-1} \frac{\langle u_j, v_k \rangle}{\langle u_j, u_j \rangle}$. I.e. we find $R_1, R_2, ...$ such that

$$AR_1R_2, \cdots R_n = \hat{Q}$$

We are essentially taking each new basis vector and making it orthogonal to the previous ones.

2) Modified Gram-Schmidt Process,
Given the same basis, we find new ONB by creating the new basis vectors to be orthogonal to the range of the vectors that haven't been orthogonalized yet.

3) **Householder** In this method, we generate a sequence of orthogonal matrices $Q_1, Q_2, ..., Q_n$ such that

$$Q_n, \cdots Q_2 Q_1 A = R.$$

This is stable and much more accurate becuase the condtion number of orthonal matrices is 1.

7 Approximation Theory and Orthogonal Polynomials

Consider a orthogonal basis $\{\phi_j(x)\}$ on $[-1, 1]$. If we want to extend this basis to be orthogonal on any interval $[a, b]$, set

$$s = \frac{2(x-a) - (b-a)}{b-a}.$$

Then $\{\phi_j(s)\}$ are an orthogonal basis on $[a, b]$, through linear extenstion of the variables.

7.1 Chebyshev Polynomials

There are many types, especially depending on the interval we wish to work with and the inner product we want.

The **Chebyshev Polynomials** are orthogonal on $[-1, 1]$ with the inner product

$$\langle f, g\rangle = \int_{-1}^{1} f(x)g(x)\frac{1}{\sqrt{1-x^2}}dx.$$

They are given by the formula $T_n(x) = \cos(n\cos^{-1}(x))$, and may also be defined recursively by

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \text{ and } T_0 = 1.$$

Let us consider the extrema and the zeros of the Chebyshev polynomials.

Definition 9. *The **max/min** of T_n occur at*

$$x_j = \cos\left(\frac{j\pi}{n}\right), \ j = 0, ..., n.$$

*The **zeros** of T_n occur at*

$$x_j = \cos\left(\frac{j-1/2}{n}\pi\right), \ j = 1, ..., n.$$

Note: that $\frac{T_n(x)}{2^{n-1}}$ is a monic polynomial of degree n.

Theorem 6. *Let Π_n be the space of all monic degree n polynomials. Then*

$$\min_{p\in\Pi_n}\left(\max_{x\in[-1,1]}|p(x)|\right) = \frac{1}{2^{n-1}}.$$

7.2 Polynomial Interpolation

Definition 10. ***Degree of Precision** is the largest positive integer m such that the quadrture is exact for all polynomials of deg $\geq m$, but not for some polynomials of deg $m+1$.*

E.G. trap. rule has d.o.p 1, simpsons rule has d.o.p. 3.

Definition 11. *The **order of accuracy** of an approximation is the power in which the error goes down by when the intervals shrink. I.e. when h gets halved, $\mathcal{E}$ gets decreased by a factor of 2^n where n is the order of accuracy.*

Thus an approximation whose error term is $\mathcal{O}(h^n)$ has order of accuracy n.

Theorem 7. *Given $x_0, ..., x_n$ distinct points and $f(x_0), ..., f(x_n)$, there is a unique polynomial of degree less than or equal to n, $p(x)$, such that*

$$p(x_j) = f(x_j),\ j = 0, ..., n.$$

Note this is not a well-conditioned problem for high degree polynomials.
For $n = 1$,

$$p(x) = \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1).$$

and for $n + 1$ points, the **Lagrange Form**

$$p(x) = \sum_{k=0}^{n} l_k(x) f(x_k),\ \text{where } l_k(x) = \prod_{j \neq k} \frac{x - xj}{x_k - x_j}.$$

the **Newton Form**

$$p(x) = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_n \prod_{j=1}^{n-1} (x - x_j),$$

, with $c_0 = f[x_0]$, $c_1 = f[x_0, x_2]$, $c_3 = f[x_0, x_1, x_2]$, etc... and the **barycentric form** given in homework.

7.2.1 Performance of Interpolation

The performance depends on the point space. Equal spacing versus weighted spacing? For smooth functions, the **interpolation error** is given by

$$\mathcal{E} = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{k=0}^{n} (x - x_k).$$

It so happens that **the zeros of the chebyshev polynomials minimize the error**. If we define λ_n to be the condition number for an nth degree polynomial interpolation,

$$\text{Equi-spaced points} \rightarrow \Lambda_n \sim \frac{2^{n+1}}{2n \log(n)}$$
$$\text{Chebyshev Zeros} \rightarrow \Lambda_n \sim \frac{2}{\pi} \log(n)$$

7.2.2 Cubic Splines

With splines, we interpolate with a different polynomial on each sub-interval. We define $S(x) = S_j(x)$ for $x_j \leq x \leq x_{j+1}$, where

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3, \ j = 0, ..., n-1 \tag{1}$$

$$S_j'(x_{j+1}) = S_{j+1}'(x_{j+1}) \tag{2}$$

$$S_j''(x_{J+1}) = S_{j+1}''(x_{j+1}). \tag{3}$$

We need boundary conditions. We have 4 different types:

Natural Spline: $S''(a) = S''(b) = 0$

Clamped Spline: $S'(a) = f'(a), \quad S'(b) = f'(b),$

Periodic function: $S'(a) = S'(b), \quad S''(a) = S''(b),$

Not-a-Knot: $S_0'''(x_1) = S_1'''(x_1), \quad S_{n-2}'''(x_{n-1}) = S_{n-1}'''(x_{n-1}).$

Theorem 8. *A **Clamped Spline** $s(x)$ for a $C^4[a,b]$ function $f(x)$ has error*

$$|f(x) - s(x)| \leq \frac{5}{384} \max_{x \in [a,b]} |f^{(4)}(x)| \max_j (x_{j+1} - x_j)^4.$$

8 Numerical Integration

We want to approximate the integral $\int_a^b f(x)dx$ numerically. There are several ways of doing this.

1. **Newton-Cotes Quadrature**
   - Equally spaced points,
   - Approximate f with interpolating polynomial
   - Integrate.

2. **Guassian Quadrature**
   -Pick points to minimize error,
   - Related to orthogonal polynomials - Approximate f using interpolating polynomial
   -Integrate.

3. **Adaptive Quadrature**
   - Low-order Newton-Cotes
   - Estimate error after interpolating and integrating
   - refine intervals with more points as needed, and repeat until error is sufficient.

8.1 Newton-Cotes Quadrature

1) Trapezoidal Rule:

$$\int_a^b f(x) = \frac{1}{2}(b-a)(f(b)+f(a)) + \mathcal{E}$$

2) Composite Trap. Rule:

$$\int_a^b f(x)dx = h\left(\frac{f(x_0)}{2} + \sum_{j=1}^{n-1} f(x_j) + \frac{f(x_n)}{2}\right) + \mathcal{E},$$

where $\mathcal{E} = \frac{b-a}{12}h^2 f''(\xi)$.

3) Composite Simpson's Rule:

$$\int_a^b f(x)dx = \frac{h}{3}\sum_{j=1}^{n/2}[f(x_{2j-2}) + 4f(x_{2j-1}) + f(x_2 j)] + \mathcal{E}$$

where $\mathcal{E} = -\frac{1}{120}\left(\frac{h}{2}\right)^4 (b-a)f^{(4)}(\xi)$.

If we approximate $f(x) = \frac{x-b}{a-b}f(a) + \frac{x-a}{b-a}f(b) + \frac{f''(\xi)}{2}(x-a)(x-b)$, then

$$\int_a^b f(x)dx = f(x_0)w_0 + f(x_1)w_1 + \mathcal{E}$$

For **higher-degree polynomial approximations**, $f(x) = P_n(x) + R(x)$, where

$$P_n(x) = \sum_{k=0}^{n} L_k(x)f(x_k), \text{ with } L_k(x) = \frac{\prod_{j\neq k}(x-x_j)}{\prod_{j\neq k}(x_k - x_j)}$$

and

$$R(x) = \prod_{k=0}^{n} \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_k).$$

8.2 Gaussian Qaudrature

• (An n-point Gaussian quad. rule is used to achieve an exact result for polynomials of degree $n-1$ or less.)
• With $\underline{n=2}$, we use $w_1 = w_2 = 1$, and $x_1 = -1/\sqrt{3}$, $x_2 = 1/\sqrt{3}$.

$$\int_{-1}^{1} f(x)dx \sim \sum_{k=1}^{n} w_k f(x_k).$$

Also,

$$\int_a^b f(x)dx = \frac{b-a}{2} \sum_{k=1}^{n} w_k f\left(\frac{b-a}{2}x_k + \frac{a+b}{2}\right) + \mathcal{E},$$

where

$$\mathcal{E}_n = \frac{(b-a)^{2n+1}(n!)^4}{(2n+1)(2n!)^3} f^{(2n)}(\xi).$$

8.3 Adaptive Quadrature

Idea: estimate error using 2 different approximations, then use more points as needed to reduce error, e.g.,

$$\int_a^b f(x)dx = S(a,b) + \mathcal{E}_1$$

and

$$\int_a^b f(x)dx = \int_a^c f(x)dx + \int_c^b f(x)dx = S(a,c) + S(c,b) + \mathcal{E}_2.$$

Let $Q_1 = S(a,b)$ and $Q_2 = S(a,c) + S(c,b)$. Then $\mathcal{E}_2 = \frac{Q_2 - Q_1}{15}$. If $|\mathcal{E}_2| <$ tolerance, set $Q = Q_1 + \mathcal{E}_2$.