CS 370 - S25
Numerical Computation
Full Course Notes

With Prof Leili Rafiee Sevyeri

These are my in-class lecture notes. They cover all course content, besides example problems.

Josiah Plett

CS 370 Notes $\boxed{1}$

Syllabus

- Floating Point Numbers
- Interpolation & Splines
- Ordinary Differential Equations
- Fourier Analysis
- Numerical Linear Algebra

(2) Floating Point

Normalized Form

$$\{\overset{\text{base}}{\beta}, \overset{\text{precision (mantissa)}}{t}, \overset{\text{range (extent)}}{L, U}\}$$

$$\pm 0.d_1 d_2 d_3 \dots d_t \times \beta^p$$

for $L \le p \le U$ and $d_1 \ne 0$

Standards of Precision

IEEE Single: (32 bits) $\{2, 24^{1}, -126, 127\}$

IEEE Double: (64 bits) $\{2, 53^{1}, -1022, 1023\}$

Machine Epsilon

"maximum relative error" is smallest E such that

$$\text{float}(1+E) > 1$$

Standards of Conversion

Round-to-nearest: Usually default. $\frac{1}{2}$ rounds up.

Truncation: Round towards zero.

(3) Error Analysis & Stability

eg: Truncation system: $E = \beta^{1-t}$

eg: Error Bounds of $(a \oplus b) \oplus c$ satisfies

$$E_{rel} \le \frac{|a|+|b|+|c|}{|a+b+c|}(2E+E^2)$$

Derive that!

Cancellation Error

173.00026 − 173.00196

Benign Cancellation

$$fl(w-z) = (w-z)(1+\delta)$$

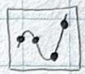
Ill-conditioned: small input Δ → large output Δ

Unstable: small error → large output error

- Conditioning: Problem itself's sensitivity
- Stability: Numerical algorithm's sensitivity

(4) Interpolation

"Predicting other values from limitted data"



It's helpful for: curve fitting, estimation, and numerical methods of integration etc.

Polynomial Interpolation $p(x) = c_1 + c_2 x + c_3 x^2 + \dots$

Ways to solve a Polynomial Interpolation

Vandermonde Matrix

$$\underbrace{\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix}}_{V} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}}_{\vec{c}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}_{\vec{y}}$$

Monomial Basis

$$p(x) = \sum_{i=1}^{n} c_i x^{i-1}$$

Lagrange Basis

$$p(x) = \sum_{i=1}^{n} y_i L_i(x) \qquad L_i(x) = \frac{(x-x_1)\cdots(x-x_n)}{(x_i-x_1)\cdots(x_i-x_n)}$$

coefficients AND data values!

but no $(x-x_i)$ or (x_i-x_i)

(5) Piecewise Interpolation

Hermite Interpolation: fit to a functions points and derivatives

eg: Fit a cubic where we have $p(0)=0$ $p(1)=3$ $p'(0)=1$ $p'(1)=0$

Piecewise Hermite Interpolation (cubics): Use 1 cubic per pair of points, sharing slope/deriv.

Knots: Point where interpolant transitions from one polynomial to the next.

Nodes: Point where data is actually specified.

Kinks: Point where derivitares are different on either side.

Boundary Conditions

Clamped: p' defined
↳ both: "complete"

Free: $p'' = 0$
↳ both: "natural"

Periodic: $p'_1 = p'_2$ & $p''_1 = p''_2$

Not-a-Knot: end segment 3rd derivatives match.

I^{th} Interval Polynomial (Hermite) (w/cubic splines)

$$p_i(x) = a_i + b_i(x-x_i) + c_i(x-x_i)^2 + d_i(x-x_i)^3$$

$$a_i = y_i \quad b_i = s_i \quad c_i = \frac{3y'_i - 2s_i - s_{i+1}}{\Delta x_i}$$

$$\Delta x_i = x_{i+1} - x_i \quad y'_i = \frac{y_{i+1} - y_i}{\Delta x_i} \quad d_i = \frac{s_{i+1} + s_i - 2y'_i}{\Delta x_i^2}$$

(6) Splines & Parametric Curves

Efficient Cubic Splines (Matrix Form)

Interior: $\Delta x_i s_{i-1} + 2(\Delta x_{i-1} + \Delta x_i) s_i + \Delta x_{i-1} s_{i+1} = 3(\Delta x_i y'_{i-1} + \Delta x_{i-1} y'_i)$

Clamped BC $(i=1, i=n)$: $s_1 = s_1^*, \quad s_n = s_n^*$

Free BC $(i=1, i=n)$: $s_1 + \frac{s_2}{2} = \frac{3}{2}y'_1, \quad \frac{s_{n-1}}{2} + s_n = \frac{3}{2}y'_{n-1}$

Tri-diagonal ü

$$\begin{bmatrix} x & x & & & & \\ x & x & x & & & \\ & x & \ddots & \ddots & & \\ & & \ddots & x & x & \\ & & & x & x & x \\ & & & & x & x \end{bmatrix} \begin{bmatrix} s_1 \\ \vdots \\ s_i \\ \vdots \\ s_n \end{bmatrix} = \begin{bmatrix} r_1 \\ \vdots \\ r_i \\ \vdots \\ r_n \end{bmatrix}$$

Concept: Parametric Curves

Instead of $P(x) = y$, consider $\vec{P(t)} = (x(t), y(t))$, allowing loops & overlaps.

approx arc-length: $t_{i+1} = t_i + \sqrt{(x_{i+1}-x_i)^2 + (y_{i+1}-y_i)^2}$

(7) ODEs - Ordinary Differential Equations

Very Simple ODE Example $y'(t) = ay(t) \Rightarrow y(t) = y_0 e^{a(t-t_0)}$

Closed-form solutions are rare, so we find approx. solutions via numerical methods. !

Timestepping

Forward Euler: $y_{n+1} = y_n + h \cdot \underbrace{f(t_n, y_n)}_{y'_n}$

real: $y(t_n)$
approx: y_n

(8) ODEs - Higher Order Timestepping

(LTE) Local Truncation Error: Error for one step of Forward Euler → $y_{n+1} - y(t_{n+1}) = O(h^2)$

We can also use the Taylor Expansion of $y(t_{n+1})$ to compute higher order LTE.

Trapezoidal Rule: $y(t_{n+1}) = y(t_n) + h \cdot y'(t_n) + \frac{h^2}{2}\left(\frac{y'(t_{n+1}) - y'(t_n)}{h}\right)$ (has error $O(h^3)$)

eg: Consider point $(x(t), y(t))$ satisfying:

$x'(t) = -y(t)$, $y'(t) = x(t)$, and $x(t_0) = 2$, $y(t_0) = 0$ $\quad t_0 = 2$

① Write down vector recurrence. (Forward Euler)

② Apply Forward Euler up to $t = 6$. (Bonus: improved Euler)

LTE: $LTE = y(t_{n+1}) - y_{n+1}$

Absolute Error: $E_{abs} = |X_{exact} - X_{approx}|$

Relative Error: $E_{rel} = \frac{|X_{exact} - X_{approx}|}{|X_{exact}|}$

Fourier Expansion: $f(t) = a_0 + a_1\cos(qt) + b_1\sin(qt) + a_2\cos(2qt) + b_2\sin(2qt) + \dots$

Taylor Series: $f(x) = f(a) + (x-a)f'(a) + \frac{(x-a)^2}{2!}f''(a) + \dots \Longleftrightarrow f(x+h) = f(x) + h\frac{\partial f}{\partial x} + \frac{h^2}{2!}\frac{\partial \partial f}{\partial \partial x} + O(h^3)$

CS 370 Notes $\boxed{2}$

(8) ODEs: More Schemes

LTE **Explicit:** only y_n or earlier define y_{n+1} on RHS.

$O(h^2)$ Forward Euler $y_{n+1} = y_n + hf(t_n, y_n)$

$O(h^3)$ Improved Euler $y_{n+1} = y_n + \frac{h}{2}(f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n)))$

Implicit: unknowns like y_{n+1} are used on RHS.

$O(h^3)$ Trapezoidal $y_{n+1} = y_n + \frac{h}{2}(f(t_n, y_n) + f(t_{n+1}, y_{n+1}))$

$O(h^3)$ BDF 2 (multistep) $y_{n+1} = \frac{4}{3}y_n - \frac{1}{3}y_{n-1} + \frac{2}{3}hf(t_{n+1}, y_{n+1})$

There are many more; implicit/explicit, single/multistep, LTE

(9) Higher Order ODEs + Stability

eg! Convert to 1st Order:

$x''(t) + y''(t)x(t) + 2t = 0$

$y''(t) + (y(t))^2 x(t) = 0$

Convert to First Order

① Introduce $y_i = y^{(i-1)}$ for each ≥ 2 derivative. $(i = 1, 2, \ldots)$

② Substitute into the original equation(s).

Stability Analysis

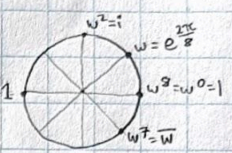
★ Practice this!!

① Apply timestepping to test equation,

② Find closed form of solution + error,

③ Find conditions on h that ensures stability.

8th roots of unity



$\omega^2 = i$
$\omega = e^{\frac{2\pi}{8}}$
$\omega^4 = -1$
$\omega^8 = \omega^0 = 1$
$\omega^7 = \overline{\omega}$

(10) Truncation Error + Adaptive Timestepping

LTE Process ★

Given timestepping Scheme $y_{n+1} = \text{RHS}\ldots$

① Replace approximations with exact (eg. $y_n \to y(t_n)$).

② Taylor expand all RHS values about t_n (like t_{n-1}).

③ Taylor expand exact solution $y(t_{n+1})$ for comparison.

④ Compute $y(t_{n+1}) - y_{n+1}$. Lowest degree non-canceling power of h gives the local truncation error.

Adaptive Timestepping

"Change timestep size h according to function."

① Compute approx solutions w/2 shemes of different orders.

② Estimate the error by taking their difference.

③ While (error > tolerance):

- Set $h := h/2$ and recompute solutions (1) and error (2).

④ Estimate the error coefficient to predict next step size h_{new}.

⑤ Repeat until end time is reached.

(11) Fourier Transforms

Continuous Fourier Series

$$f(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi kt}{T}\right) + \sum_{k=1}^{\infty} b_k \sin\left(\frac{2\pi kt}{T}\right)$$

$$a_0 = \frac{\int_0^{2\pi} f(t)\,dt}{2\pi} \qquad a_k = \frac{\int_0^{2\pi} f(t)\cos(kt)\,dt}{\int_0^{2\pi} \cos^2(kt)\,dt} \qquad (b_k \text{ is sin})$$

Handy Identities

Orthogonality

$\int_0^{2\pi} \cos(kt)\sin(jt)\,dt = 0, \forall\, k,j \in \mathbb{Z}$

↳ $\cos^2$ or $\sin^2$ require $k \neq j$

$\int_0^{2\pi} \sin(kt)\,dt = 0$

Euler's Formula

$e^{i\theta} = \cos(\theta) + i\sin(\theta) \qquad e^{-i\theta} = \cos(\theta) - i\sin(\theta)$

$$\cos(\theta) = \frac{e^{i\theta} + e^{-i\theta}}{2} \qquad \sin(\theta) = \frac{e^{i\theta} - e^{-i\theta}}{2i}$$

(12) Discrete Fourier Analysis

Given our sinusoidal expression of f(t) from before, we now have:

$$\left[\!\left[\, f(t) = \sum_{k=-\infty}^{+\infty} c_k e^{ikt} \qquad c_k = \frac{1}{2\pi}\int_0^{2\pi} e^{-ikt} f(t)\,dt \,\right]\!\right]$$

Converting between c_k and a_k, b_k

$(k > 0)$

$$a_0 = c_0, \quad c_k = \frac{a_k}{2} - \frac{ib_k}{2} \qquad c_{-k} = \frac{a_k}{2} + \frac{ib_k}{2}$$

Discrete Interpolation: N points → N coefficients

$$f(t) \approx \sum_{k=-\frac{N}{2}+1}^{N/2} c_k e^{\frac{(2\pi i)kt}{T}} = \sum_{k=0}^{N-1} F_k W^{nk} \qquad W = e^{\left(\frac{2\pi i}{N}\right)}$$

(13) More DFT

More Identities

$k, \ell \in [0, N-1]$

$$\sum_{j=0}^{N-1} W^{jk} W^{-j\ell} = \sum_{j=0}^{N-1} W^{j(k-\ell)} = N\delta_{k,\ell} \to \delta_{k,\ell} = \begin{cases} 0; & k \neq \ell \\ 1; & k = \ell \end{cases}$$

DFT ⇨ Inverse DFT $\quad F_k = \frac{1}{N}\sum_{n=0}^{N-1} f_n W^{-nk} \Rightarrow f_n = \sum_{k=0}^{N-1} F_k W^{nk}$

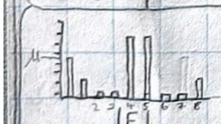
$$\sum_{j=0}^{N-1} x^j = \frac{x^N - 1}{x - 1} \leftarrow x \neq 1$$

Properties

- Doubly-infinite
- Periodic in N
- Conjugate symmetric at N/2 ($F_k = \overline{F_{N-k}}$)

(14) Even More DFT

Power Spectrum



$F_0 = \frac{1}{N}\sum_{n=0}^{N-1} f_n$

Real data → symmetric plots

(15) Fast Fourier Transform ★

① Apply butterfly-like algorithm:

$$\vec{f} \to g_n = \frac{1}{2}\left(f_n + f_{n+\frac{N}{2}}\right), \quad h_n = \frac{1}{2}\left(f_n - f_{n+\frac{N}{2}}\right)W^{-n} \qquad n \in \left[0, \frac{N}{2}-1\right]$$

② Unscramble bit-reversed coefficients:

$[F_0, F_2, F_1, F_3] \longrightarrow [F_0, F_1, F_2, F_3] \qquad 110 \leftrightarrow 011, \; 1000 \leftrightarrow 0001$

(16) Images & Aliases

Simple compression strategy: Discard $|F_k| < tol$

$$\text{2D} \quad F_{k,\ell} = \frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} f_{n,m} W_N^{-nk} W_M^{-m\ell}$$

$W_x = e^{\frac{2\pi i}{x}}$

Aliasing

if non-zero, gets aliased

$$F_k = c_k + \underbrace{c_{k+N} + c_{k-N} + c_{k+2N} + c_{k-2N} + \ldots}$$

So high-frequency $c_k \notin [-\frac{N}{2}+1, \frac{N}{2}]$ alias as low-frequency F_k for $k \in [-\frac{N}{2}+1, \frac{N}{2}]$

Prevent Aliasing

① ⇧ Sampling resolution

② Filter too-high freq.s before sampling

CS 370 Notes $\boxed{3}$

(17) PageRank

Structure the web as a directed graph.

$\deg(j)$ = # of edges out of node j.

Adjacency Matrix: $G_{ij} = \begin{cases} 1; & \text{link } j \to i \text{ exists} \\ 0; & \text{otherwise} \end{cases}$

Markov Chain Matrix: $P_{ij} = \begin{cases} 1/\deg(j); & \text{link } j \to i \text{ exists} \\ 0; & \text{otherwise} \end{cases}$

↳ Solve dead-ends by turning each column of all zeros into a column of $1/i$ s.

$d_i = \begin{cases} 1; & \deg(i)=0 \\ 0; & \text{otherwise} \end{cases}$ $\quad e = [1, 1, \dots, 1]^T \quad P' = P + \frac{1}{R} e d^T$

↳ Solve loops by escaping to a random page with probability $1-\alpha$ (α usually large, ≈ 0.85)

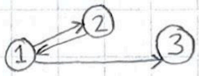
Google Matrix: $M = \alpha P' + (1-\alpha) \frac{1}{R} e e^T$ teleportation preferences

Linear Algebra Time!

Random Surfer Algorithm

Follow K random links R times...

Issues: • Way too many pages
• Dead ends & cycles



$$G_{33} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$ "column to row"

$$P_{33} = \begin{bmatrix} 0 & 1 & 0 \\ 1/2 & 0 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$$ divide by $\deg(j)$

$$P'_{33} = \begin{bmatrix} 0 & 1 & 1/3 \\ 1/2 & 0 & 1/3 \\ 1/2 & 0 & 1/3 \end{bmatrix}$$ fill zeros

$$M = \begin{bmatrix} 1/20 & 9/10 & 1/3 \\ 19/40 & 1/20 & 1/3 \\ 19/40 & 1/20 & 1/3 \end{bmatrix}$$ $\alpha = 0.85$

↳ sums to 1, always

(18) Numerical Linear Algebra

Probability Vector: vector q s.t: $0 \le q_i \le 1$ and $\sum_{i=1}^{R} q_i = 1$

With initial state $\vec{p_0}$ and Markov Matrix M, then $M^n p_0$ is the probabilities of being at each page after n steps.

Pagerank asks: $p^\infty = \lim_{k \to \infty} M^k p_0$

↳ more simply, $p^{n+1} = M p^n$

Optimization

Precomputation: p^∞ computed once & stored, then just filter w/keywords.

Sparsity: Most entries are zero... Sadly M is dense, so we use big-brain:

Page Rank Algorithm

```
p^0 = e/R
For k = 1, 2, ... until converged:
    p^k = M p^{k-1}
    If max_i |[p^k]_i - [p^{k-1}]_i| < tol, quit
End For
```

(19) Gaussian Elimination

FACT: Every Markov Matrix has 1 as an eigenvalue.

FACT 2: $\forall \lambda$ of Markov Matrix M, $|\lambda| \le 1$.

FACT 3: If Q is a Positive MM, there is only one linearly independent eigenvector with $|\lambda| = 1$.

FACT 4: PageRank will Converge!

↳ convergence rate of google matrix is α, eg. α^n = accuracy

But we want to compute it efficiently, so we use our classic RREF gaussian operations!

EigenValues/Vectors

Eigenvalue λ & eigenvector x:

$Qx = \lambda x \rightarrow (\lambda I - Q)x = 0$

Thus we solve $\det(\lambda I - Q) = 0$

eg: $Q = \begin{bmatrix} 2 & 2 \\ 5 & -1 \end{bmatrix}$ $\quad \det(\lambda I - Q) = \det \begin{bmatrix} \lambda-2 & -2 \\ -5 & \lambda+1 \end{bmatrix}$

det = ad − bc $\quad = \lambda^2 - \lambda - 12 = (\lambda - 4)(\lambda + 3) = 0$

For numerical solution, we take a different view:

① Factor A into $A = LU$, L & U are triangular.
② Solve $Lz = b$ for intermediate z.
③ Solve $Ux = z$ for x.

$Ax = b$

$\|A\|_2 = \max_i \sqrt{|\lambda_i|}$ ← eigenvalues of $A^T A$

(20) LU Factorization

Gaussian Elimination as Factorization

① RREF for U(pper): $\underbrace{\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 2 \\ 1 & 2 & -1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} 0 \\ 4 \\ 2 \end{bmatrix}}_{b} \rightarrow \left[\begin{array}{ccc|c} 1 & 1 & 1 & 0 \\ 0 & -3 & 1 & 4 \\ 0 & 0 & -\frac{5}{3} & \frac{10}{3} \end{array}\right] = U$

② Get L(ower): $\begin{bmatrix} 1 & 1 & 1 \\ 0 & -3 & 1 \\ 0 & 0 & -\frac{5}{3} \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & -1/3 & 1 \end{bmatrix} = L$ $\quad$ =1

③ Solve via Backward ($Ux = z$) or Forward ($Lz = Pb$) sub.

③+ $A \to PA = LU$, where $P =$ (eg) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

★ largest factor top

④ Solve! ⓐ $b' = Pb$ ⓑ $Lz = b'$ ⓒ $Ux = z$

Efficient PageRank

$$p^{(n+1)} = Mp^{(n)} = \alpha P p^{(n)} + \frac{\alpha}{R} e d^T p^{(n)} + \frac{1-\alpha}{R} e \overbrace{(e^T p^{(n)})}^{=1}$$

1 in zero-columns

(21) Norms & Conditioning

P-norms

$$\|\vec{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}$$

for $p = 1, 2, \dots, \infty$

$(\|x\|_\infty = \max_i |x_i|)$

Matrix Norms

$$\|A\| = \max_{\|x\| \ne 0} \frac{\|Ax\|}{\|x\|}$$

$$\|A\|_1 = \max_j \sum_{i=1}^{n} |A_{ij}|$$ max column

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |A_{ij}|$$ max row

Condition Number: $K_n = \|A^{-1}\|_n \|A\|_n$ $(n=2)$

Perturbing

$Ax = b$

↓

$A(x + \Delta x) = b + \Delta b$ added

↓

$\boxed{A \Delta x = \Delta b}$

$$\frac{\|\Delta x\|}{\|x\|} \le K(A) \frac{\|\Delta b\|}{\|b\|}$$

$$\frac{\|\Delta x\|}{\|x + \Delta x\|} \le K(A) \frac{\|\Delta A\|}{\|A\|}$$

(22) Conditioning

Residual: $r = b - Ax_{approx} = b - A(x + \Delta x) \Rightarrow \frac{\|\Delta x\|}{\|x\|} \le K(A) \frac{\|r\|}{\|b\|}$

Gaussian Elim: $\frac{\|x - \hat{x}\|}{\|\hat{x}\|} \le K(A) \frac{\|A\| \epsilon_{machine}}{\|A\|} \le K(A)\, \epsilon_{machine}$

< Leili looking after us before the midterm

Us refusing to face the reality of the final >

v   Me sharing my notes with y’all 🫂 We will survive!   v