©2025 ELEANORWAISS.GITHUB.IO

subject: Numerical Analysis
date: 11/13/2025
keywords:
topic: Welcome

addition.cpp

float x = 0
x += 0.1
exit when x == 1

Whomp, whomp ...

We're not using ℝ ...

Loss of precision
Loss of associativity

Numerical Analysis

Study of how computations are done — on computers

How efficient are algorithms?
↳ Can we bound error?
↳ What kind of error?

Computations

Integration, differentiation, invert matrices
↓
Matrix not invertible? Perturbate it such that it is!
Chaotic dynamical systems

2 objectives

Practical "cookbook" of algorithms
Theoretical study of error / analysis of efficiency

Ex 2

exp.cpp

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$
$$= \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

$$\Delta = \frac{x}{n}$$

i) eto x = X
@ eto x += $\frac{x}{n++}$

Can use Taylor's Remainder Theorem for poly-approx

It went wrong ...
↳ evaluate on negative #s

forward_diff.cpp

©2025 ELEANORWAISS.GITHUB.IO

subject: Numerical Analysis

date: 1/15/2024

topic:

keywords:

Graph example

$x^2 + 0.0001\sin(100x)$

Looks like x^2

↳ $f'' > 0$ breaks

Intuition of fxns

A collection of evaluations

↳ comprehensive ability to evaluate a graph "everywhere"

vs computer graphics ~ few evals / pixel

Assumes fxns are cheap — Calculus? Yes. Real life? No!

"Just graph it" breaks spirit of class

Our typical problem

Given a fxn f, find x s.t. $f(x)=0$.

↳ Want $f(x)=10$? find x s.t. $f(x)-10=0$

Root finding algorithms

Depending on problem, we may know some interval $[a,b]$ where $\exists x \in [a,b]$ s.t. $f(x)=0$.

If not, exhaustive search: $a, a+h, a+2h, \dots,$ until a sign change.

(assumes continuity)

Even this could be troublesome...

So failure to detect sign change $\not\Rightarrow f(x)$ never 0

Could even be

©2025 ELEANORWAISS.GITHUB.IO

subject | date | keywords

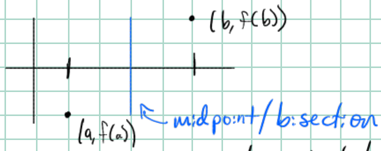
topic

Brackets

An interval containing a root of f is called a bracket

Eg. bracket [a, b]

so $\operatorname{sgn} f(a) \neq \operatorname{sgn} f(b)$



Bisection

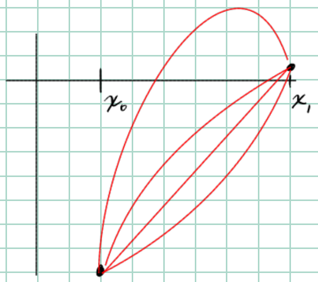
Evaluate f @ $\frac{b+a}{2}$. $\operatorname{sgn} f\left(\frac{a+b}{2}\right)$ gives a new bracket! Recurse, bisecting in halfs until desired precision

In binary, every iteration gives 1 bit

Precision $\sim \frac{b-a}{2^i}$ for i iterations. Choose wisely

Double roots are issues... dealt with in due time

New bracket



It could be very badly approximated linearly, but we don't know! Test linearly and pray

Eqn of a line
Yay, MA101

$$y - f(x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_1)$$

Zero ⇒ $y = 0$ gives linear equation

$$\boxed{x_2 = \frac{-f(x_1)(x_1 - x_0)}{f(x_1) - f(x_0)} + x_1}$$

Linear Interpolation

Evaluate $f(x_2)$ to get new bracket, repeat

We lose "proof" where this approach converges faster than bisection

Mix of methods

Linear interp for handful of iterations, bisect if no large improvement, repeat

Motivating Theme: Calculus

$$f(x) = \sum_{n=0}^{\infty} a_n x^n$$

Return to addition.cpp

Gulf War SCUD missile defense. 26 killed due to same fault as here.

Floating point arithmetic is not associative subject: Numerical Analysis

date: 17 Jan 25

keywords

topic: Floating Point Arithmetic

Lack of associativity in floating point arithmetic

4 significant figures → round or truncation

$$1000 + (0.6 + 0.6) = 1000 + 1.2 = 1001$$
$$(1000 + 0.6) + 0.6 = 1000 + 0.6 = 1000$$

We lose associativity

harmonic.cpp

$$\sum_{k=1}^{n} \frac{1}{k} \quad \text{vs} \quad \overbrace{\sum_{k=1}^{n} \frac{1}{n+1-k}}^{\text{more accurate}}$$

$n = 10{,}000{,}000$ 15.4037 15.686

Bit allocation in floats/double

$$\underset{\text{sign}}{1}\ \underbrace{11}_{\text{exponent}}\ \underbrace{53}_{\text{mantissa / sig figs}}$$

(52 bits actually needed) in binary

1-8-23 for float

1/10 in binary

```
        .00011
1010 | 1.000000 000000
       .1010
         1100
        -1010
           10
```

Error

Anything not rational w/ denom = 2^k is an approx in FPA (else ∞-decimal expansion)

Large terms need to be added later in a fpa sum to preserve significant figs

©2025 ELEANORWAISS.GITHUB.IO

subject: Numerical Analysis

date: 22 Jan 25

keywords

topic: Floating Point, V2

Binary vs Decimal

$\underline{10}$ $\underline{10}$

Double: 64 bits
- 1 sign
- 11 exponent $\sim [-308, 308]$
- 53 mantissa — 17 mantissa

$\pm$ 53 digits $\times 2^{exp}$

$[-2^{10}, 2^{10}]$

Geometry of FPA

Very close and clustered around zero and otherwise spread out fast

Non-unique zero – useful for understanding divergence

Pitfalls

Truncation error

Subtracting nearly-equal quantities

Eg take 2 #s agreeing on first 10 bits

$\Rightarrow$ 10 leading bits are useless for precision

Adding quantities of different magnitude

Worst case, $a > b \Rightarrow$ easy for $a + b = a$ for $b \neq 0$

Division by #s close to zero

Root Finding / Linear Interp

$$x_i = x_{i-1} - f(x_{i-1}) \frac{x_{i-1} - x_{i-2}}{f(x_{i-1}) - f(x_{i-2})}$$

1 new function evaluation each recursion

Let this be its own method

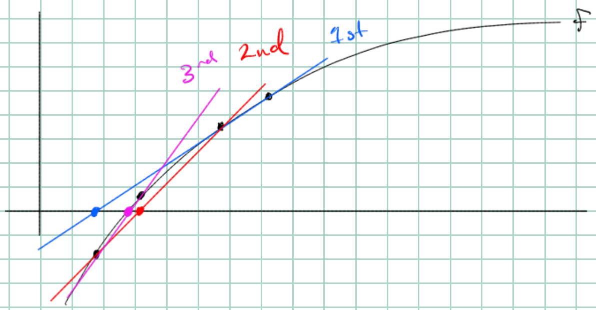
"Secant method"

↳ We no longer care about a bracket

↳ No guarantee of finding a root

Bisection – each iteration gains 1 bit of info

Secant – each iteration increases by a factor of φ



©2025 ELEANORWAISS.GITHUB.IO

subject

date

keywords

topic

Newton-Raphson Method

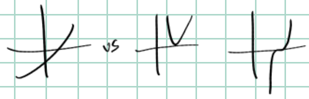
$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Increases bits by factor of 2

2 new function evals each recursion step

↳ assuming $f'(x) \neq 0$

double root



subject: Numerical Analysis | date: 24 Jan 25 | keywords:

topic: Analysis of Algorithms

Newton's Method

Taylor Series @ x_0

$$f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2}f''(x_0)(x-x_0)^2 + \dots$$

constant approx, linear approx, quadratic approx

or $+\frac{1}{2}f''(\delta_0)(x-x_0)$ — can truncate for δ_0 between x, x_0

$$= \sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!}(x-x_0)^n$$

Let $f(x)$ be zero, since we're looking for root

$$x f'(x_0) = x_0 f'(x_0) - f(x_0) - \frac{1}{2}f''(\delta_0)(x-x_0)^2$$

$$\rightsquigarrow x = x_0 - \frac{f(x_0)}{f'(x_0)} - \frac{1}{2}\frac{f''(\delta_0)}{f'(x_0)}(x-x_0)^2$$

→ goes to zero quickly as $x_0 \to x$

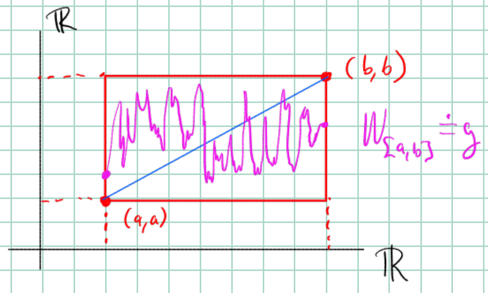
$$\rightsquigarrow x = x_0 - \frac{f(x_0)}{f'(x_0)} \quad \text{when } f'(x_0) \neq 0 \text{ (simple root)}$$

Consider $N_f(x) = x - \frac{f(x)}{f'(x)}$. Is there a point for which $N_f(x) = x$? I.e. a fixed point

Theorem — Fixed Point Theorem

If g cts on $[a,b]$ and $g:[a,b]\to[a,b]$ then g has a fixed point. Further, if g diff on (a,b) for some $0<\lambda<1$ we have $|g'(x)|\leq\lambda$ $\forall x\in[a,b]$ then the fixed point is unique

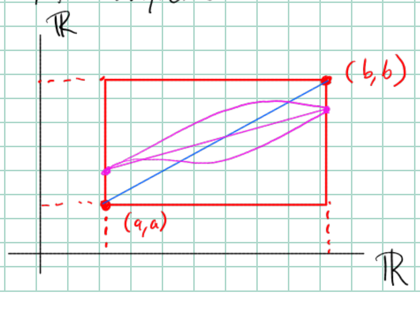
V. similar to Banach's FPT.



Proof sketch

If $g(a)=a$ or $g(b)=b$, we're done. WLOG assume not. So $g(a)>a$ and $g(b)<b$. Consider $h(x)=g(x)-x$. So $h(a)>0$ and $h(b)<0$. Yet h cts, so $\exists\, c\in(a,b)$ s.t. $h(c)=0$. So $g(c)=c$.

Now, for uniqueness. BWOC sps 2 fixed points, α β



$$f(\beta) - f(\alpha) = \beta - \alpha$$

yet $\exists\, c\in(\alpha,\beta)$ s.t. $f'(c) = \frac{f(\beta)-f(\alpha)}{\beta-\alpha} = 1$ ↯

©2025 ELEANORWAISS.GITHUB.IO

subject | date | keywords

topic

Now, for convergence. (Lipschitz)

$$|x_n - x^*| = |g(x_{n-1}) - g(x^*)| = |g'(\theta_n)||x_{n-1} - x^*| \le \lambda |x_{n-1} - x^*|$$

And on recursion, $|x_n - x^*| \le \lambda^n |x_0 - x^*| \xrightarrow{n\to\infty} 0$

so $x_n \to x^*$ ▨

Back to Newton

$$N_f(x) = x - \frac{f(x)}{f'(x)}$$

$$N_f'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)\,f''(x)}{(f'(x))^2}$$

Need $f \in C^2$? Less regularity?

Want to find a root of f, i.e. $f(x) = 0$.

This is almost an ideal map since num $\to 0$ as $x \to x_{root}$

$\sup_{[a,b]} N_f' = \lambda \to 0$. Superconvergence!

subject: Numerical Analysis

date: 27 Jan 25

keywords:

topic: Superconvergence (Quadratic)

Iteration

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Calc III and Complex

$f: \mathbb{C}^n \to \mathbb{C}^n$ or $\vec{f} = \begin{bmatrix} f_1(\vec{z}) \\ \vdots \\ f_n(\vec{z}) \end{bmatrix}$ for $f_i: \mathbb{C}^n \to \mathbb{C}$

Taylor Series: $f(x) = f(x_0) + f'(x_0)(x - x_0) +$ higher order

$f: \mathbb{C}^n \to \mathbb{C}^n$, $f(\vec{z}) = f(z_0) + [Df(z_0)](z - z_0) +$ higher order

Yay, manifolds!

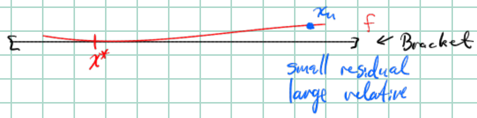
same manip ⇝ $z_{n+1} = z_n - [Df(z_n)]^{-1} f(z_n)$

Error

3 types: Absolute $|x_n - x^*|$

Relative $\frac{|x_n - x^*|}{|x^*|}$ more helpful

Residual $|f(x_n)|$



Compare against other well-understood sequences

E.g. $\{c_n\}$ w/ $\lim_{n\to\infty} c_n = 0$.

$\left\{\frac{1}{n}\right\}$ or $\left\{\frac{1}{2^n}\right\}$ or $\left\{\frac{1}{\log n}\right\}$

quick, quicker (superconverges), slow

Say $\{x_n\}$ is $O(c_n)$ if $|x_n - x^*| < k c_n$ for $k \in \mathbb{R}^+$

Calc I

$$\lim_{x\to 0} \frac{\sin x}{x} = 1 \xrightarrow{x = \frac{1}{n}} \lim_{n\to\infty} \frac{\sin\left(\frac{1}{n}\right)}{\frac{1}{n}}$$

How quickly does $\frac{\sin\left(\frac{1}{n}\right)}{\frac{1}{n}} \to 1 \iff n\sin\left(\frac{1}{n}\right) - 1 \to 0$

$$= n\left[\frac{1}{n} - \frac{1}{3!}n^{-3} + \frac{1}{5!}n^{-5} - \dots\right] - 1$$

$$= \underbrace{-\frac{1}{3!}\frac{1}{n^2}}_{\text{goes to 0 the slowest}} + \frac{1}{5!}\frac{1}{n^4} - \frac{1}{7!}\frac{1}{n^6} + \dots$$

So $\left|\frac{\sin\left(\frac{1}{n}\right)}{\frac{1}{n}} - 1\right| \le \frac{1}{3!}\frac{1}{n^2}$ ⇝ is $O\left(\frac{1}{n^2}\right)$

⇝ $\lim_{x\to 0} \frac{\sin x}{x}$ is $O(x^2)$

©2025 ELEANORWAISS.GITHUB.IO

subject	date	keywords
topic		

Superconvergence

For any iteration satisfying the hypotheses of the Fixed Point Theorem,

$$|x_n - x^*| \le k\lambda^n$$

For Newton's method, $\lambda \to 0$

This isn't strong enough to give desired info.

A sequence is said to converge w/ order p if $\lim_{n\to\infty} \frac{|x_{n+1}-x^*|}{|x_n-x^*|^p} = M \in (0,1)$

N.M. $\lim_{n\to\infty} \frac{|x_{n+1}-x^*|}{|x_n-x^*|} = \overset{=|g'(x^*)|}{\lambda} \to 0 \Rightarrow p > 1.$

Taylor's Thm gives exactness

$$g(x_n) = g(x^*) + g'(x^*)(x_n - x^*) + \frac{g''(\xi_n)}{2}(x_n - x^*)^2$$

True in general. But when g is the Newton Iteration,

$$g(x_n) = g(x^*) + \frac{g''(\xi_n)}{2}(x_n - x^*)^2$$

$$|x_{n+1} - x^*| = \left|\frac{g''(\xi_n)}{2}(x_n - x^*)^2\right|$$

$$\to \frac{|x_{n+1}-x^*|}{|x_n-x^*|^2} = \left|\frac{g''(\xi_n)}{2}\right|$$

under certain conditions

$$\to \lim_{n\to\infty} \frac{|x_{n+1}-x^*|}{|x_n-x^*|^2} = \left|\frac{g''(x^*)}{2}\right| = M < 1$$

Quadratic

$f(x)$ is 3-times diff, x^* simple root, initial guess sufficiently close to x^*

Abs err

$e_n = |x_n - x^*| \approx c\, e_{n-1}^2$ for quadratic

Secant method

$e_n \approx c\, e_{n-1} e_{n-2} \to c e_n \approx c e_{n-1}\, c e_{n-2}$

Let $d_i = \ln c e_i$

$\to d_n \approx d_{n-1} - d_{n-2} \leftarrow$ Fib! φ it's golden!

NM requires evaluating $f(x_n), f'(x_n)$. Quadratic convergence

Secant method requires only $f(x_n)$. φ convergence $\varphi^2 > 2$.

Is Secant method better?

©2025 ELEANORWAISS.GITHUB.IO

subject

date 29 Jan 25

keywords

topic

Modifications to NM

$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ $Df(x^*)$ non singular

for ideal behavior, need $f \in C^2$, x^* simple

In practice, assume f is an "expensive" program or experiment.
Do I even have access to $Df, D^2f, D^3f, \dots$

How to solve this problem?
- Secant method
- Approximate w/ $\frac{f(x_n+h)-f(x_n)}{h}$ for suitable h — Numerical approximation of derivatives is its own topic, to be continued...
- Use $Df(x_n)$ a constant for several iterations
  ↳ Mean value inequality or suprema bound on Df
  Via secant slope

Brent's Method

Richard P.

Insight: approx via parabola
Return to brackets $[x_0, x_2]$
Pick $x_1 \in (x_0, x_2)$ (could be midpoint, could be random, could be lin interp, etc)
Two points $\Rightarrow$ unique line
3 points $\Rightarrow$ unique quadratic

Find a parabola of the form $\alpha(x-x_2)^2 + \beta(x-x_2) + \gamma = Q(x)$

Eval f @ x_0, x_1, x_2, solve α, β, γ
Inverse function theorem $\Rightarrow$ If $DS(\vec{x})$ non singular, $\exists!$ soln

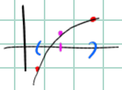
$$f(x_0) \overset{\text{want}}{=} \alpha(x_0-x_2)^2 + \beta(x_0-x_2) + \gamma$$
$$f(x_1) \overset{\text{want}}{=} \alpha(x_1-x_2)^2 + \beta(x_1-x_2) + \gamma$$
$$f(x_3) \overset{\text{want}}{=} \gamma$$

$$\rightsquigarrow \begin{bmatrix} (x_0-x_2)^2 & (x_0-x_2) \\ (x_1-x_2)^2 & (x_1-x_2) \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} f(x_0)-f(x_2) \\ f(x_1)-f(x_2) \end{bmatrix}$$

$$\rightsquigarrow \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \frac{1}{(x_0-x_2)^2(x_1-x_2)-(x_1-x_2)^2(x_0-x_2)} \begin{bmatrix} x_1-x_2 & -(x_0-x_2) \\ -(x_1-x_2)^2 & (x_0-x_2)^2 \end{bmatrix} \begin{bmatrix} f(x_0)-f(x_2) \\ f(x_1)-f(x_2) \end{bmatrix}$$

in denom

$$(x_0-x_2)(x_1-x_2)(x_0-\cancel{x_2}-x_1+\cancel{x_2})$$



$$\Rightarrow \alpha = \frac{(x_1-x_2)(f(x_0)-f(x_2))-(x_0-x_2)(f(x_1)-f(x_2))}{(x_0-x_1)(x_0-x_2)(x_1-x_2)}$$

$$\beta = \frac{-(x_1-x_2)^2(f(x_0)-f(x_2))+(x_0-x_2)^2(f(x_1)-f(x_2))}{(x_0-x_1)(x_0-x_2)(x_1-x_2)}$$

Gives Q.
Solve w/ quadratic eqn

→ guaranteed existence of unique root b/c working in bra-ket

©2025 ELEANORWAISS.GITHUB.IO

subject | date | keywords

topic

$X = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ if $-b > 0$ want to avoid subtraction

i.e. avoid via multiply by conjugate

$$X = \left\{ \frac{-b+\sqrt{b^2-4ac}}{2a} \atop {-b>0 \atop {\Downarrow \atop b<0}} \right\} \text{ or } \left\{ \frac{2c}{-b+\sqrt{b^2-4ac}} \atop {-b<0 \atop {\Downarrow \atop b>0}} \right\}$$

Avoid cancellation of leading digits ⇒ remove sig fig and add "noise"

Root of Q $\to r$ gives new bracket. $[x_0, r]$ or $[r, x_2]$

Give up braket, guarantee soln w/ ~1.84 convergence

---

31 Jan 25

Equation Solvers

Matlab
Mathematica
Python

"It just works"

The Brent-Dekker Root Finder

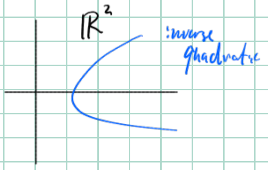
→ Previous: Quadratic interp

Side bar on solving quadratics

Instead, use inverse quadratic interp

The poly $x = Q(y)$ is found the same way

Note we need $Q(0)$ ⇒ guaranteed "root"



If we have $[a, c]$ w/ $b \in (a,c)$

$$d = Q(0) = \frac{af(b)f(c)}{(f(a)-f(b))(f(a)-f(c))} + \frac{bf(a)f(c)}{(f(b)-f(a))(f(b)-f(c))} + \frac{cf(a)f(b)}{(f(c)-f(a))(f(c)-f(b))}$$

Some analysis required for picking midpoint versus Inv. Quad. Interp

Big theme: there really is no "one method fits all". Beyond "trivial" problems, custom methods/algorithms to problem needed.

Higher derivatives ⇒ higher convergence

©2025 ELEANORWAISS.GITHUB.IO

subject | date | keywords

topic

Know $f''(x_i) \Rightarrow$ cubic convergence
Halley's Method(s)

Everything thus far has been a single root.
A degree n poly has n roots, require $\mathbb{C}$ arithmetic even for $\mathbb{R}$ roots

What about multiple roots? Polynomials $\Leftrightarrow$ eigenvalue problem

Humans: solve $\det(A-\lambda I)$
Computers: not this shit

What about other functions.
Suppose we have a root, r $(f(r)=0)$ ← numerical estimates for both r, $f(r)$

$$g(x) = \frac{f(x)}{(x-r)}$$

E.g.

$$f_0(x) = \sin x$$
$$f_1(x) = \frac{\sin x}{x}$$
$$f_2(x) = \frac{\sin x}{x(x-\pi)}$$
$$f_3(x) = \frac{\sin x}{x(x^2-\pi^2)}$$
$$\vdots$$

Derivative free methods are preferred here!

Once we have r, avoid it in future iterations

©2025 ELEANORWAISS.GITHUB.IO

subject: Numerical | date: | keywords:

topic: Linear Algebra

A Crash Course

Prototypical problem: solve system of linear eqns (over $\mathbb{R}$, $\mathbb{C}$, or $\mathbb{F}$)

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_n + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

$\rightsquigarrow$ Find values for $\{x_n\}$ s.t. these hold

$\to A\vec{x} = \vec{b}$

Linearity is a key property!

Row reduction / elimination is "easy"

↳ What is the numeric significance of this?

The process is a one-time computation

If many experiments / etc are conducted, likely that a_{ij} will stay constant, b_i change.

i.e. solve $Ax = b_1$ and $Ax = b_2$ and $\cdots$ and $Ax = b_{10^k}$ such systems

Reasonable to ask if A^{-1} is able to be used.

No!

- From a computer's POV, A^{-1} always exist due to FPA errors
- det A grows v. quickly w/ n

Two major themes:

1) Orthogonality via inner product

$A\vec{x} = \vec{b} \rightsquigarrow b = \vec{a}_1 x_1 + \vec{a}_2 x_2 + \cdots \vec{a}_n x_n$ where $\vec{a}_i$ i-th column

Special Case

Sps all $\vec{a}_i$ mutually orthogonal.

$\Rightarrow \langle a_i, a_j \rangle = 0$ if $i \neq j$

$\langle a_i, a_i \rangle = c_i > 0$ (positive definite)

$\rightsquigarrow \langle a_1, a_1 x_1 + \cdots + a_n x_n \rangle = \langle a_1, b \rangle$

$\downarrow$

$c_1 x_1 = \langle a_1, b \rangle$ so $x_1 = \dfrac{\langle a_1, b \rangle}{\langle a_1, a_1 \rangle}$ ← project a_i onto b

$c_i x_i = \langle a_i, b \rangle \to x_i = \dfrac{\langle a_i, b \rangle}{\langle a_i, a_i \rangle}$

How often does this happen naturally? Almost never. So force it!

Graham-Schmidt Orthogonalization

If this transform exists (numerically), way to solve many different linear systems.

Scale s.t. $c_i = 1$ (denominator)

subject | date | keywords

topic

2^{nd} major theme

Finding $\vec{\eta}_i$ s.t. A acts like scalar multiplication

i.e. $A\vec{\eta}_i = \lambda_i \vec{\eta}_i$

So we know $\vec{\eta}_i, \lambda_i$

$$\langle A\vec{\eta}, \vec{x} \rangle = \lambda \langle \vec{\eta}, \vec{x} \rangle$$

There exists a factorization of $A = PDP^{-1}$ for $P = [\vec{\eta}_i]$, $D = diag(\lambda_i)$

$$\rightsquigarrow DP\vec{x} = P\vec{b}$$

$$\rightsquigarrow \vec{x} = P\underbrace{D^{-1}}_{diag(\frac{1}{\lambda_i})}P^{-1}\vec{b}$$

Spectral Theorem

Let A be symmetric. Then $A = PDP^T$

subject

date 5 Feb 25

keywords

topic More Fun with Matricies

Operations preserving systems of Equations

Multiply by nonzero scalar
Addition / Linear Comb.
Interchange positions / positions

Gauss-Jordan & Reduced Row Echelon Form

If we have an nxn matrix, how difficult is Gaussian elim?

1st step takes n-1 additions over n-1 rows
2nd Step takes n-2 —〃— n-2 —〃—
⋮
n-1 step takes 1 —〃— —〃—

$$\sum_{i=1}^{n} (n-i)^2 \text{ is cubic in } n \approx \frac{n(n-1)(2n-1)}{6}$$

$\Rightarrow$ Gaussian Elim is $O(n^3)$ related b/w/o Elementary matrices

Yet, multiplication of 2 nxn matrices is $O(n^3)$

Simple example

$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ $x_1 = x_2 = 1$

done symbolically

$\begin{bmatrix} 10 & 1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $r_2 - (0.1) r_1 \approx \boxed{0 x_1} + \boxed{0.9 x_2}$

done numerically

$\begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}\vec{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \rightarrow \left[\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 0 & -10^{20} & -10^{20} \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 0 & 1 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 10^{-20} & 0 & 0 \\ 0 & 1 & 1 \end{array}\right]$ $x_1 = 0, x_2 = 1$

Always have largest #s (in magnitude) in pivots w/ a single scan down c_1 (partial pivoting)

Full pivoting is quadratic search $O(n^2)$ of all elements, interchange rows / columns

$\left[\begin{array}{cc|c} 1 & 1 & 2 \\ 10^{-20} & 1 & 1 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & 1 & 2 \\ 0 & 1 & 1 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & 1 \end{array}\right] \rightarrow x_1 = x_2 = 1$

Suppose we've solved $A\vec{x} = \vec{b}_1$, it used $O(n^3)$.
Sorry, But wait, we needed $\vec{b}_2$.
Sorry, But we actually need $\vec{b}_{10^6}$.
⋮
The steps applied to A remain unchanged.
Goal: incur a one-time cost for quicker $^{O(n^2)}$ solutions later
$A = LU$ w/ lower / upper matrices
Solve $A\vec{x} = \vec{b} \Leftrightarrow LU\vec{x} = \vec{b} \Leftrightarrow \underbrace{U\vec{x} = \vec{b}}_{O(n^2)}$ and $\underbrace{L\vec{y} = \vec{x}}_{O(n^2)}$

©2025 ELEANORWAISS.GITHUB.IO

subject

date: 7 Feb 25

keywords

topic: Matrix Shit

Recall from last time

Row reduction used 3 rules: exchange, scale, linear comb.
These rules hold over $\mathbb{R}^{\text{or } \mathbb{C}}$ — something we're not in (FPA).
↳ Order / Size matters
Goal: Determine order of operations
Full / Partial pivoting selects largest (in magnitude) element
Internal alarm bell: working w/ numbers of large difference in magnitude
Naively applied Gaussian elim is very <u>unstable</u> (in the <u>correct</u> sense of the word)

Solving $Ax=b$

$O(n^3)$ naively / using first principles of linear algebra
↳ Assumed A has n^2 elements
Eg $A = PDP^{-1}$

In reality, most matrices are sparse — many 0 elements
↳ GE on a sparse matrix becomes dense
Store (i, j, a_{ij}) for all nonzero entries.
Significantly less than n^2 (for $n >> 0$) storage

Idea $A^n\vec{x}$ converges to η
If A sparce, no fundamental changes to A are made and this is an iterative algorithm

Ex

$$\begin{bmatrix} 6 & 18 & 3 \\ 2 & 12 & 1 \\ 4 & 15 & 3 \end{bmatrix} \to \begin{bmatrix} 6 & 18 & 3 \\ 0 & 6 & 0 \\ 0 & 3 & 1 \end{bmatrix} \qquad \begin{bmatrix} 6 & 18 & 3 \\ 0 & 6 & 0 \\ 0 & 0 & 1 \end{bmatrix} U$$

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & & 1 \end{bmatrix} \xrightarrow{\text{right multiply}} \begin{bmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 2/3 & \frac{1}{2} & 1 \end{bmatrix} L$$

Scale C_1 by $\frac{1}{a_{11}}$

Claim: $A = LU$: $\begin{bmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 2/3 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} 6 & 18 & 3 \\ 0 & 6 & 0 \\ 0 & 0 & 1 \end{bmatrix} \checkmark = \begin{bmatrix} 6 & 18 & 3 \\ 2 & 12 & 1 \\ 4 & 15 & 3 \end{bmatrix}$

"Doolittle Representation"

Do not use any new storage!
Only store one matrix!

$$\begin{bmatrix} 6 & 18 & 3 \\ 2 & 12 & 1 \\ 4 & 15 & 3 \end{bmatrix} \to \begin{bmatrix} 6 & 18 & 3 \\ \boxed{1/3} & 6 & 0 \\ \boxed{2/3} & 3 & 1 \end{bmatrix} \to \begin{bmatrix} 6 & 18 & 3 \\ \boxed{1/3} & 6 & 0 \\ \boxed{2/3} & \boxed{1/2} & 1 \end{bmatrix}$$

implicit 1's on diagonal for L

Pivoting

How to incorporate pivoting?
Permutation Matrices

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} c & d \\ a & b \end{bmatrix}$$

Left multiplication: row permutation
Right mult: column permutation
Store in $O(n)$ via cyclic permutation notation
Partial pivoting → only row permuts → left multiplication
Store only <u>row swaps</u>, minimize data movement
⇝ $PA = LU$
↳ Pivots

subject | date | keywords

topic

Solve $A\vec{x} = \vec{b}$?

$\Rightarrow PA\vec{x} = P\vec{b} \stackrel{def}{=} \vec{d}$

$\rightsquigarrow LU\vec{x} = \vec{d}$

Monday Feb 10

Analysis

LUD decomp – Revised GJ-Elim

Fast forward to Section 6

Condition numbers

$|x_k - x^*|$ absolute error

$|f(x_k)|$ size of guess

$|x_k - x^*| / |x^*|$ relative error

↳ measuring distance. Good for $\mathbb{R}$ or FPA.
What about in $\mathbb{R}^n$? $\mathbb{R}^{n \times n}$?

Vectors usually use $\|\vec{x}\| = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$; i.e. $\|\cdot\|_2$ Euclidean norm

not superfluous over $\mathbb{C}$

Taxicab norm: $\|\vec{x}\|_1 = \sum_{i=1}^{n} |x_i|$

for proof: p norm: $\|\vec{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}$, $p > 1$.

↳ Supnorm: $\|\vec{x}\|_\infty = \sup x_i$

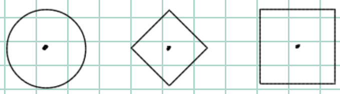
Norms: Positive definite $\|f\| \geq 0$, w/ $\|f\| = 0 \Leftrightarrow f = 0$

Homogeneous $\|\alpha f\| = |\alpha| \|f\|$

Triangular $\|f + g\| \leq \|f\| + \|g\|$

Geometry of Norms

Circles



$\|\cdot\|_2$ $\|\cdot\|_1$ $\|\cdot\|_\infty$

What about matrices?

"Dumb": Just use $\mathbb{R}^{n^2}$ or $\mathbb{C}^{n^2}$

"Better": Spectral norm: $\|A\|_s = \sqrt{\rho(A^T A)}$, ρ is largest (in magnitude) eigenvalue ← related to 2-norm

↳ This is a norm: Positive definite, homogeneous, triangle ineq

↳ Also satisfies $\|AB\|_s \leq \|A\|_s \|B\|_s$ (Operator norm w/r/t A?)

Goal

Let $\kappa(A)$ be the condition number of A and be defined as $\underbrace{\|A\|_s}\underbrace{\|A^{-1}\|_s}$

easy to compute/estimate

If $\kappa(A) = 10^p$, then a soln to $A\vec{x} = \vec{b}$ loses p sig figs

↳ Any numerical soln incurs this cost, regardless of method

©2025 ELEANORWAISS.GITHUB.IO

subject | date | keywords

topic

Idea

Don't solve $A\vec{x}=\vec{b}$. $\longrightarrow$ $\|A\|, \|\vec{x}\| \geq \|\vec{b}\|$

Solve $A'\vec{x}=\vec{b}'$, $A'=A+E$, $\vec{b}'=\vec{b}+\vec{e}$

error

$\kappa(A)$ gives a numeric value to how bad A act w.r.t error & perturbation

Consider $A\vec{x}=\vec{b}$, producing a computed solution $\hat{x}$. Check: $\vec{r}=\vec{b}-A\hat{x}$.

Residual error $\rightarrow$ $\|\vec{r}\| = \|A\vec{x}-A\hat{x}\| \rightsquigarrow \frac{\|\vec{r}\|}{\|A\|_s} \leq \|\vec{x}-\hat{x}\|$

$$\text{So} \quad A^{-1}\vec{r} = \vec{x}-\hat{x} \rightsquigarrow \|A^{-1}\|\|\vec{r}\| \geq \|\vec{x}-\hat{x}\|$$

$$\rightsquigarrow \frac{\|A^{-1}\|_s\|\vec{r}\|}{\|\vec{x}\|} \geq \frac{\|\vec{x}-\hat{x}\|}{\|\vec{x}\|}$$

$$\text{So} \quad \underbrace{\frac{\|\vec{x}-\hat{x}\|}{\|\vec{x}\|}}_{\text{relative error}} \leq \frac{\|A^{-1}\|_s\|\vec{r}\|}{\|\vec{x}\|} \leq \overbrace{\|A^{-1}\|_s\|A\|_s}^{\kappa(A)} \underbrace{\frac{\|\vec{r}\|}{\|\vec{b}\|}}_{\text{residual error}}$$

subject

date: 12 Feb 25

keywords

topic: Iterative Methods (Ch 3)

Goal

Solve $A\vec{x} = \vec{b}$

Assume A sparse

Ch 2 Theme

A is $n \times n \rightsquigarrow O(n^3)$ work

LU Decomp requires upfront $O(n^3)$ work for future $O(n^2)$

Now

A has N nonzero entries, solve in $O(N)$. (Assume all nonzero)

↳ Write $A = L + D + U$

$$\vec{b} = A\vec{x} = (L + D + U)\vec{x} = L\vec{x} + D\vec{x} + U\vec{x}$$

$$\rightsquigarrow D\vec{x} = \vec{b} - L\vec{x} - U\vec{x}$$

$$\rightsquigarrow \vec{x} = D^{-1}\left(\vec{b} - L\vec{x} - U\vec{x}\right)$$

$$= D^{-1}\vec{b} - D^{-1}(L + U)\vec{x}$$

Iteration step (w/ superscripts) $\vec{x}^{(k)} = D^{-1}\vec{b} - D^{-1}(L+U)\vec{x}^{(k-1)}$ w/ initial guess $\vec{x}^{(0)}$

So
$$x_1^{(k)} = \frac{b_1 - \sum_{i=2}^{n} a_{1i} x_i^{(k-1)}}{a_{11}}$$

$$x_2^{(k)} = \frac{b_2 - \sum_{\substack{i=1 \\ i\neq 2}}^{n} a_{2i} x_i^{(k-1)}}{a_{22}}$$

$$\vdots$$

$$x_n^{(k)} = \frac{b_n - \sum_{i=1}^{n-1} a_{ni} x_i^{(k-1)}}{a_{nn}}$$

Example

$$A = \begin{bmatrix} 4 & 1 & & \\ 1 & 4 & 1 & \\ & 1 & 4 & 1 \\ & & 1 & 4 \end{bmatrix} \quad \vec{X} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \longrightarrow \begin{cases} 4x_1 + x_2 = 1 \\ x_1 + 4x_2 + x_3 = 2 \\ x_2 + 4x_3 + x_4 = 3 \\ x_3 + 4x_4 = 4 \end{cases} \Rightarrow \begin{cases} x_1 = \frac{1 - x_2}{4} \\ x_2 = \frac{2 - x_1 - x_3}{4} \\ x_3 = \frac{3 - x_2 - x_4}{4} \\ x_4 = \frac{4 - x_3}{4} \end{cases}$$

Let $\vec{x}^{(0)} = \vec{0}$

So
$$\vec{x}^{(1)} = \begin{bmatrix} \frac{1-0}{4} \\ \frac{2-0-0}{4} \\ \frac{3-0-0}{4} \\ \frac{4-0}{4} \end{bmatrix} = \begin{bmatrix} \frac{1}{4} \\ \frac{1}{2} \\ \frac{3}{4} \\ 1 \end{bmatrix} \quad \vec{x}^{(2)} = \begin{bmatrix} \frac{1-\frac{1}{2}}{4} \\ \frac{2-\frac{1}{4}-\frac{3}{4}}{4} \\ \frac{3-\frac{1}{2}-1}{4} \\ \frac{4-\frac{3}{4}}{4} \end{bmatrix} = \begin{bmatrix} \frac{1}{8} \\ \frac{1}{4} \\ \frac{3}{8} \\ \frac{13}{16} \end{bmatrix} \quad \vec{x}^{(3)} = \begin{bmatrix} \frac{1-\frac{1}{4}}{4} \\ \frac{2-\frac{1}{8}-\frac{3}{8}}{4} \\ \frac{3-\frac{1}{4}-\frac{13}{16}}{4} \\ \frac{4-\frac{3}{8}}{4} \end{bmatrix} = \begin{bmatrix} 3/16 \\ 3/8 \\ 41/64 \\ 29/32 \end{bmatrix}$$

Jacobi's Iteration

Always works when $|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \quad \forall i$ "Diagonally dominant"

©2025 ELEANORWAISS.GITHUB.IO

subject

date

keywords

topic

Method, revised

Use new values as you go

$$x_1^{(1)} = \frac{1-0}{4}$$

$$x_2^{(1)} = \frac{2-\frac{1}{4}-0}{4} = \frac{7}{16}$$

$$x_3^{(1)} = \frac{3-\frac{7}{16}-0}{4} = 41/64$$

$$x_4^{(1)} = \frac{4-41/64}{4} = 215/256$$

$$x_1^{(2)} = \frac{1-7/16}{4} = \frac{9}{64}$$

$$x_2^{(2)} = \frac{2-9/64-41/64}{4} = \frac{39}{128}$$

$$x_3^{(2)} = \frac{3-39/128-215/256}{4} = \frac{475}{1024}$$

$$x_4^{(2)} = \frac{4-475/1024}{4} = 3621/4096$$

Gauss-Seidel Iterations

©2025 ELEANORWAISS.GITHUB.IO

subject

date 14 Feb 25

keywords

topic Iterative Methods

Eigenvalues & How to Compute Them

Find $\vec{\eta}, \lambda$ s.t. $A\vec{\eta} = \lambda\vec{\eta} = (\lambda I)\vec{\eta}$
$\rightsquigarrow (A - \lambda I)\vec{\eta} = \vec{0}$, find $\ker(A - \lambda I)$

Find nullspace? Compute $c_\lambda(A) = \det(A - \lambda I)$ characteristic polynomial

Compute roots of $c_\lambda(A)$.
HARD problem
Need numeric root finding for nontrivial setup
What about $\mathbb{C}$ sol'ns?
Expected, even w/ $A \in M_{n\times n}(\mathbb{R})$

We have to work over $\mathbb{C}$
Consider $A \in M_{n\times n}(\mathbb{C})$

Assumptions, for now → "Safe" to assume n distinct eigenvalues λ_i w/ $\vec{\eta}_i$ ← lin. indep
Order i by size: $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$

Consider "random" $\vec{x} \neq \vec{0}$. So $\exists$ rep $\vec{x} = \sum_{i=1}^n a_i \vec{\eta}_i$, but we don't know it!

$$A\vec{x} = A\left(\sum_{i=1}^n a_i \vec{\eta}_i\right) = \sum_{i=1}^n a_i \lambda_i \vec{\eta}_i$$
$$A^k\vec{x} = A\left(A^{k-1}\left(\sum_{i=1}^n a_i \vec{\eta}_i\right)\right) = \sum_{i=1}^n a_i \lambda_i^k \vec{\eta}_i$$

So, dominant term is $a_i \lambda_i^k \vec{\eta}_i$ (since $|\lambda_i|$ maximized)

Suppose we know λ_1:

$$\lambda_1^{-1} A\vec{x} = a_1\vec{\eta} + \sum_{i=2}^n \frac{\lambda_i}{\lambda_1} a_i \vec{\eta}_i$$
$$\lambda_1^{-k} A\vec{x} = a_1\vec{\eta} + \sum_{i=2}^n \left(\frac{\lambda_i}{\lambda_1}\right)^k a_i \vec{\eta}_i$$

Since $|\lambda_1|$ maximal, $\left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0$ for $k \to \infty$ w/ $i \neq 1$

Power Method

Process reveals $\vec{\eta}_1$ (since unique up to scalar a_1)
if we know λ_1 (or something very near it)

At each step, scale to unit length!

$\vec{x}_0$ random.

$\vec{y}_1 = A\vec{x}_0$; $m = \|\vec{y}_1\|_\infty$; $\vec{x}_1 = \frac{1}{m}\vec{y}_1 = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ e^{i\theta} \\ \vdots \\ \alpha_n \end{bmatrix}$ for $|\alpha_i| < 1$ ← rth index
← easiest to compute

Keep track of previous few r^{th} places.
As $k \to \infty$, (r_i, m_i) will stop changing. So $m_k = \lambda_1$ for $k \gg 1$.

Also $\vec{x}_k$ is "good" approximation of $\vec{\eta}$

PageRank is this algo!

$\left(\frac{\lambda_2}{\lambda_1}\right)^k$ gives convergence factor/order of seperation.

©2025 ELEANORWAISS.GITHUB.IO

subject | date | keywords

topic

What of λ_2? or λ_ℓ?

$A = PDP^{-1} = P\begin{bmatrix}\lambda_1 & & \\ & \lambda_2 & \\ & & \ddots \\ & & & \lambda_n\end{bmatrix}P^{-1}$

$A - \lambda_1 I = PDP^{-1} - P\begin{bmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_1\end{bmatrix}P^{-1} = P\begin{bmatrix}0 & & & \\ & \lambda_2-\lambda_1 & & \\ & & \lambda_3-\lambda_1 & \\ & & & \ddots \\ & & & & \lambda_n-\lambda_1\end{bmatrix}P^{-1}$

Come back to this ... something's wrong here

Find λ_n (the smallest)

Use power method on A^{-1}.

↳ Solve for $\vec{x}$ in $A\vec{x} = \vec{b}$, iteratively backsolve

each step is $O(n^2)$, so total $\sim O(n^3)$

λ_1, λ_n gives the condition number $\kappa(A)$.

©2025 ELEANORWAISS.GITHUB.IO

subject

date 17 Feb 2025

keywords

topic

Exam 1 posted in full — Due 26 Feb 2025

Power Method

Generate initial guess "randomly" x_0, iterate $x_i = \frac{A x_{i-1}}{\|A x_{i-1}\|_\infty}$. Converges by seperation of eigenvalues $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$ w/ order of convergence $|\frac{\lambda_1}{\lambda_2}|$. Learning multiple $\lambda_i / \vec{\eta}$ pairs is hard.

"Easy" to compute λ_n, by back solving A^{-1} w/ power method: Solve $A\vec{x}_i = \vec{x}_{i-1}$, scale accordingly (inverse power method)

Circularity issues w/ using Gauss-Seidel method to solve inv. power method problem with needing to know λ_n to compute $\kappa(A)$

We hope $|\lambda_1| < 1$ to have an axis of contraction

Sufficient and necessary condition for convergence

Jacobi Iteration: $A = L + D + U$, $(L + D + U)\vec{x} = \vec{b} \rightsquigarrow D\vec{x} = \vec{b} - (L+U)\vec{x}$

$\rightsquigarrow \vec{x} = D^{-1}\vec{b} - \underbrace{D^{-1}(L+U)}_{M:=}\vec{x}$

Need M w/ $|\lambda_1| < 1$.

Other computational topics from Lin Alg

- Graham-Schmidt Orthogonalization
- Regression / Least Squares projections *
- QR decomp
- Cholesky decomp
- Singular Value decomp

"New" *

*

Major Themes Remaining

- Curve Fitting
- Numerical differention
- Numerical integrating / solving DE

©2025 ELEANORWAISS.GITHUB.IO

subject

date: 19 Feb 25

keywords

topic: Exam 1 Scratchwork

6

a) Solve $A\vec{x}=\vec{b}$, $A=\begin{bmatrix} .0002 & 0.2 \\ 2 & 2 \end{bmatrix}$ $\vec{b}=\begin{bmatrix} 0.2 \\ 4 \end{bmatrix}$ w/ 3-digit arithmetic naively.

$$\left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 4 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & -2000 & -2000 \end{array}\right]$$

$$\sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & 1 & 1 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0 & 0 \\ 0 & 1 & 1 \end{array}\right]$$

$$\rightsquigarrow \vec{x}=\begin{bmatrix} 0 \\ 1 \end{bmatrix}$$ not close ✗

b) W/ partial pivoting $\left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0.0002 & 0.2 & .2 \end{array}\right] \sim \left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0 & 2000 & 2000 \end{array}\right] \sim \left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0 & 1 & 1 \end{array}\right]$

$$\sim \left[\begin{array}{cc|c} 2 & 0 & 2 \\ 0 & 1 & 1 \end{array}\right] \rightsquigarrow \vec{x}=\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$ closer ✓

c) $\vec{b}=\begin{bmatrix} .2 \\ 5.6 \end{bmatrix} \rightsquigarrow \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 5.6 \end{array}\right] \rightsquigarrow \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & -2000 & -1990 \end{array}\right] \rightsquigarrow \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & 1 & 0.995 \end{array}\right]$

$$\rightsquigarrow \left[\begin{array}{cc|c} 0.0002 & 0 & 0.001 \\ 0 & 1 & 0.995 \end{array}\right] \rightsquigarrow \vec{x}=\begin{bmatrix} 5 \\ 0.995 \end{bmatrix}$$

$S=\begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix} \rightsquigarrow SA\vec{x}=\left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 2 & 2 & 5.6 \end{array}\right] \rightsquigarrow \left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 0 & -2000 & -1990 \end{array}\right] \rightsquigarrow \left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 0 & 1 & 0.995 \end{array}\right]$

$$\rightsquigarrow \left[\begin{array}{cc|c} 0.002 & 0 & .01 \\ 0 & 1 & 0.995 \end{array}\right] \rightsquigarrow \vec{x}=\begin{bmatrix} 5 \\ 0.995 \end{bmatrix}$$

d) $$\begin{bmatrix} .007 & -.8 \\ -.1 & 10 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=\begin{bmatrix} .7 \\ 10 \end{bmatrix}$$

$$S \rightsquigarrow \begin{bmatrix} 0.7 & -8 \\ -.1 & 10 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=\begin{bmatrix} 7 \\ 10 \end{bmatrix}$$

$$\left[\begin{array}{cc|c} 0.07 & -0.8 & 7 \\ -.1 & 10 & 10 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.07 & -0.8 & 7 \\ 0 & -1.4 & 20 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.07 & -8 & 7 \\ 0 & 1 & -14.3 \end{array}\right]$$

$$\sim \left[\begin{array}{cc|c} 0.07 & 0 & -107 \\ 0 & 1 & -14.3 \end{array}\right] \rightarrow \vec{b}=\begin{bmatrix} -1530 \\ -14.3 \end{bmatrix}$$

6a) $\left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 4 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & -2000 & -2000 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & 1 & 1 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0 & 0 \\ 0 & 1 & 1 \end{array}\right]$

So $\vec{x} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

b) $\left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0.0002 & 0.2 & 0.2 \end{array}\right] \sim \left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0 & 2000 & 2000 \end{array}\right] \sim \left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0 & 1 & 1 \end{array}\right] \sim \left[\begin{array}{cc|c} 2 & 0 & 2 \\ 0 & 1 & 1 \end{array}\right]$

So $\vec{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

c) $\left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 5.4 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & -2000 & -1990 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & 1 & 0.995 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.0002 & 0 & 0.001 \\ 0 & 1 & 0.995 \end{array}\right]$

So $\vec{x} = \begin{bmatrix} 5 \\ 0.995 \end{bmatrix}$

$\left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 2 & 2 & 5.4 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 0 & -2000 & -1990 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 0 & 1 & 0.995 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.002 & 0 & 0.1 \\ 0 & 1 & 0.995 \end{array}\right]$

So $\vec{x} = \begin{bmatrix} 5 \\ 0.995 \end{bmatrix}$

d) $\left[\begin{array}{cc|c} 0.7 & -8 & 7 \\ & & 10 \end{array}\right] \sim \left[\begin{array}{cc|c} & & \\ & & \end{array}\right] \sim \left[\begin{array}{cc|c} & & \\ & & \end{array}\right] \sim \left[\begin{array}{cc|c} & & \\ & & \end{array}\right]$

So $\vec{x} = \begin{bmatrix} \\ \\ \end{bmatrix}$

$\begin{bmatrix} .007 & -.8 \\ -.1 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} .7 \\ 10 \end{bmatrix}$

$S \rightarrow \begin{bmatrix} 0.7 & -8 \\ -.1 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 7 \\ 1.0 \end{bmatrix}$

$\left[\begin{array}{cc|c} 0.07 & -0.8 & 7 \\ -.1 & 10 & 10 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.07 & -0.8 & 7 \\ 0 & -1.4 & 20 \end{array}\right] \sim \left[\begin{array}{cc|c} 0.07 & -8 & 7 \\ 0 & 1 & -14.3 \end{array}\right]$
$\sim \left[\begin{array}{cc|c} 0.07 & 0 & -107 \\ 0 & 1 & -14.3 \end{array}\right] \rightarrow \vec{b} = \begin{bmatrix} -1530 \\ -14.3 \end{bmatrix}$

©2025 ELEANORWAISS.GITHUB.IO

subject

date: 19 Feb 2025

keywords

topic: In Class — Lin Alg

Condition numbers

$\kappa(A) = \sqrt{|\lambda_1||\lambda_n|^{-1}}$ for λ_1 largest, λ_n smallest eigenvalues

Question:

What sort of matrices have κ that make for good numerical computations

Discussion

$\kappa(A) = \sqrt{\frac{|\lambda_1|}{|\lambda_n|}}$

minimal value is 1

Diagonal matrices ... a kind of trivial/special case

↳ Avoids this "linear algebra" problem

↳ Can have arbitrary condition numbers

$\kappa(A)$ minimized exactly when $\lambda_j = e^{i\theta}$ for $\theta \in [0, 2\pi)$ for all j!

A matrix is Unitary iff $UU^T = I$ when all columns of U are mutually orthogonal

Unitary matrices are "the best" matrices for computation!

Note: $U^T = U^{-1}$, so here we can actually, really, by god, compute U^{-1}: It's just U^T!

Spectral Decomposition Theorem

For Real Symmetric A, $\exists$ Unitary U and Diagonal D where $A = UDU^T$ for D containing the Real eigenvalues

$$A = \underbrace{U}\,\underbrace{D}\,\underbrace{U^T}$$

These are nice! Well behaved! Numerically these are what we want!

This is where all our problems lie

So... use orthogonality to solve our problems

What do we see?

$A^T A\vec{x} = A^T\vec{b}$

A $m \times n$ matrix w/ indep columns $(m \geq n)$

$A^T A$ is Real symmetric

Normally, $A\vec{x} = \vec{b}$ has no solution $(m \neq n)$ it's "overdetermined"

But... $A^T A\hat{x} = A^T\vec{b}$ has a solution, w/ $\hat{x}$ the closest we can get to $\vec{x}$

$\hat{x}$ is the least squares solution

That is, $\|A\hat{x} - b\|$ is minimized!

This is, quite literally, regression modeling

subject

date

keywords

topic

Consider $A \in M_{m \times n}(\mathbb{R})$ where the columns of A are $\vec{a}_1, \dots, \vec{a}_n$ (assume linear independence)

So... $\text{col}\, A = \text{Span}\{\vec{a}_1, \vec{a}_2, \dots, \vec{a}_n\}$

$= \text{Span}\{\vec{\alpha}_1, \vec{\alpha}_2, \dots, \vec{\alpha}_n\}$ when $\vec{a}_i$ mutually orthonormal

Apply Graham-Schmidt!

$$\alpha_1 = \frac{a_1}{\|a_1\|}$$

$$\beta_2 = a_2 - (a_2 \cdot \alpha_1)\vec{\alpha}_1, \quad \alpha_2 = \frac{\beta_2}{\|\beta_2\|}$$

$$\beta_3 = a_3 - (a_3 \cdot \alpha_1)\alpha_1 - (a_3 \cdot \alpha_2)\alpha_2, \quad \alpha_3 = \frac{\beta_3}{\|\beta_3\|}$$

$$\beta_k = a_k - \sum_{j=1}^{k-1} (a_k \cdot \alpha_j)\alpha_j ; \quad \alpha_k = \frac{\beta_k}{\|\beta_k\|}$$

etc