CS 371

Sibelius Peng

Spring 2019

Contents

Intro

Info

- Abdullah Ali Sivas
- MC 6321
- OH: F: 1:30 - 3:30 in MC 6342
- Matlab tut: next week: M: 5:30 - 6:30
- Midterm: June 17, 8:30 - 9:20
- Grade distribution
  - 40% Assignments (4 or 5 equally weighted)
  - 20% Midterm
  - 40% Final
- References
  - Course Notes (primary) Errata will be on LEARN
  - Numerical Analysis (Burden and Faires)
  - Numerical/Methods: Algorithms & Applications (Fausett)
  - Introduction to Scientific Computing (Van Loon)
  - Numerical Computing with MATLAB (Cline Noler)

Outline

- Floating Point (Chapter 1) 4 lecs

- Root-Finding (Chapter 2) 5 lecs
- Numerical Linear Algebra (Chapter 3) 6 lecs
- Polynomial Interpolation (Chapter 5) 6 lecs
- Numerical Integration (Chapter 6) 7 lecs
- Discrete Fourier Transform (Chapter 4) 8 lecs

Purpose Of The Course

Find or develop algorithms which solve a given problem computationally. These algorithms should be

- accurate: produce a result numerically close to the actual solution
- efficient: solve the problem fast and using reasonable amount of resources
- Robust: The algorithm works well for wide range of inputs

Given a problem, consider the problem itself

1. Is the problem sensitive to small changes in inputs?
   Consider $\int_0^1 \frac{1}{x^\alpha} dx$ $\begin{cases} \text{Finite Result} & \text{if } \alpha > 1 \\ \text{diverges} & \text{if } 0 < \alpha \leq 1 \end{cases}$
2. Can we still find a reasonably good numerical solution?
   Algorithm
   (a) Find an algorithm which works for all data **OR**
   (b) Find an algorithm which works for some data

   Think about the application, does any input make sense?
   Think about efficiency, because robust algorithms are usually inefficient.
   Think about robustness, because efficient algorithms are usually not robust.

CHAPTER 1

Errors and Propagation

1.1 Sources of Error

- Errors in Input
  - Rounding Errors: Unavoidable, because computers cannot do infinite precision.

    $x = 0.0034567$
    $x = 0.34567 \times 10^{-2}$ normalize
    $\hat{x} = 0.3457 \times 10^{-2}$ 4-digit precision
    Error $|x - \hat{x}| = 0.0000003$
  - Data Uncertainty: Always a possibility, especially when dealing with practical problems. For example,
    - Measurement Error: Possible large (usually 1-2 digits accuracy) e.g. engineering/economics data
    - Storing Data: rounding errors, numbers may not be exactly representable
    - Previous Computation: if the input is the output of an earlier computation
- Errors as a result of a calculation, and approximation or an algorithm
  - Rounding Error
    Computer do basic operations

$$a \oplus b \neq a + b$$

$$\begin{aligned} 3 = (\sqrt{3})^2 &= \sqrt{3} \cdot \sqrt{3} \qquad \text{Assume 2 -digits precision} \\ &= 1.73 \times 1.73 \\ &= 2.99 \end{aligned}$$

  - Truncation error

The major task of a numerical analysis is to quantify this: Many of the numerical methods we consider (e.g. trapezoid rule for numerical integration, Newton's rule for root-finding) can be thought as a truncated Taylor's series. Whatever left out from the series is called the truncation error.

In general, we will call any error we introduce through an algorithm or approximation will be called the truncation error.

How do we measure the error?

- Absolute Error: $|\underbrace{x}_{\text{Actual value}} - \underbrace{\hat{x}}_{\text{Approximation}}|$

  Note This defn is different from the notes by abs. value

- Relative Error
$$\frac{|x-\hat{x}|}{|x|}$$

Examples $p = 0.30012 \times 10^1, \hat{p} = 0.30200 \times 10^1$

$$|p-\hat{p}| = 0.188 \times 10^{-1}$$
$$\frac{|p-\hat{p}|}{|p|} = 0.626 \times 10^{-2}$$

$p = 0.30012 \times 10^{-2}, \hat{p} = 0.30200 \times 10^{-2}$

$$|p-\hat{p}| = 0.188 \times 10^{-4}$$
$$\frac{|p-\hat{p}|}{|p|} = 0.626 \times 10^{-2}$$

1.1.1 Catastrophic Cancellation (example)

$a = 0.1234567 \qquad b = 0.1234111$
Take a five-digit precision calculator
$\hat{a} = 0.12346 \qquad \hat{b} = 0.12341$
$abs\ err\ for\ a = 0.0000033 \qquad rel\ err \approx 0.0000267$
$abs\ err\ for\ b = 0.0000011 \qquad rel\ err \approx 0.000008$
$c = a - b = 0.0000456 \qquad \hat{c} = \hat{a} - \hat{b} = 0.00005$
$|c - \hat{c}| = 0.0000044 \qquad \frac{|c-\hat{c}|}{|c|} \approx 0.12$

1.1.2 Truncation Error - Taylor Series

Let f be an analytic[1] function. An a point a,

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

[1] infinitely differentiable

This power series is called the Taylor Series.

$$f(x) \approx \sum_{k=0}^{N} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

is called the n-th degree Taylor Polynomial.

The Taylor Remainder Theorem The truncation error is

$$R_n(x) = \frac{f^{(n+1)}(\xi_{n,x})}{(n+1)!}(x-a)^{n+1}$$

$\xi \in [x, a]$

Example $f(x) = \cos x$ around $a = 0$

$$\cos x = \cos 0 - \frac{\sin 0}{1!}(x-0) - \frac{\cos 0}{2!}(x-0)^2 + \ldots = 1 - \frac{x^2}{2} + \frac{x^4}{4!} + \ldots$$

We want to truncate the series at the third (fourth) term. We want to know how much error we introduce. Assume $x \in [-\pi, \pi]$.

$$|R_3(x)| = \left|\frac{f^{(4)}(\xi_{4,x})}{4!}x^4\right| = \left|\frac{\cos(\xi_{4,x})}{4!}x^4\right| \leq \left|\frac{x^4}{4!}\right| \leq \frac{\pi^4}{4!}$$

Ex Consider $\int_0^1 e^{-x^2}dx$. Can we approximate it?
Expand the Taylor Series around $x = 0$.

$$e^{-x^2} = 1 - x^2 + \frac{x^4}{2} - \frac{x^6}{6} + \frac{x^8}{24} + \ldots$$

$$\int_0^1 e^{-x^2} \approx \int_0^1 \left(1 - x^2 + \frac{x^4}{2} - \frac{x^6}{6}\right)dx = \left(x - \frac{x^3}{3} + \frac{x^5}{10} - \frac{x^7}{42}\right)\Bigg|_0^1 = \frac{26}{35} = 0.7\overline{428571}$$

Actual value

$$\frac{1}{2}\sqrt{\pi}\ \ \text{erf}(1) \approx 0.746824$$

where $\text{erf}(x) = \int_0^x e^{-t^2}dt$

$$\left|\int_0^1 R_8(x)dx\right| = \left|\int_0^1 \frac{f^{(8)}(\xi_x)}{8!}x^8\ dx\right|$$

$f^{(8)}(x) = 16e^{-x^2}(16x^8 - 224x^6 + 840x^4 - 840x^2 + 105)$

1.1.3 An algorithm can propagate the error!

Ex from textbook

$$e^x = \sum_{k=0}^{n} \frac{x^k}{k!} + R_n$$

Compute $e^{-5.5}$ for $n = 24$
$\hat{z} = 0.0057563$, actual $z \approx 0.0040868$.
The relative error is large. Because catastrophic cancellation. How can we solve the issue?

$$e^{-x} = \frac{1}{e^x} \approx \frac{1}{\sum_{k=0}^{n} \frac{x^k}{k!}}$$

$\hat{z} = 0.0040865$.

The moral of the story Truncation error is important, however, it may not be the major source of error.

1.2 Floating Point Numbers and Operations

FPNs are how to represent real numbers on a computer effectively. FPNs are finite precision approximations to real numbers.

Floating points systems have three components

$$\pm 0.\underbrace{x_1 x_2 \dots x_m}_{\text{the mantissa}} \times \underbrace{b}_{\text{the base}}^{\underbrace{\pm y_1 y_2 \cdots y_e}_{\text{the exponent}}}$$

$1 \le x_1 \le b-1$
$0 \le x_i \le b-1, \qquad i = 2, \dots, m$
$0 \le y_j \le b-1, \qquad j = 1, \dots, e$

- The base: The base of the number system we use (2 for out computers) denoted by b.
- The mantissa: Where we store the normalized value of the number represented. Maximal size denoted by M.
- The exponent: Similar to mantissa, firstly this is an integer, secondly it is basically the offset of normalization. Maximal size denoted by E.

Then we can shorthand floating point system as $F[b, M, E]$

Real Numbers	FPNs
Infinite Range	Might have symbols for $+\infty, -\infty$
Positive/Negative	
Infinite Precision	Finite Precision
Infinitely many numbers between two numbers	Finitely many points between numbers, they also may not be evenly spaced.

Example $F[b=3, M=4, E=2]$ and represent $x = (0.0011220212)_3$

Soln normalize

$$x = 0.1122|00212 \times 3^{-2}$$

rounding $\hat{x} = 0.1122 \times 3^{-2}$

Example $F[b=2, M=5, E=2]$ with rounding represent $x = (11010.101)_2$

Soln normalize

$$x = 0.1101|101 \times 2^{101}$$

Always remember to write the exponent in the correct base
rounding

$$\hat{x} = 0.11011 \times 2^{101} \qquad \to +\infty$$

A Non-standard binary computer IEEE standard is the floating point system which our computer actually use. We will consider an easy way to introduce Floating point system.

Single Precision $F[b=2, M=23, E=7]$ + 2 sign bits in total, single precision takes 32-bit of memory (4-byte). This FPN has the form

$$\Big|\underbrace{s_m}_{\substack{\text{single bit for mantissa}\\ 0\to+\\ 1\to-}}\Big| b_1 b_2 \dots b_{23} \Big| \underbrace{s_e}_{\substack{\text{single bit for exponent}\\ 0\to+\\ 1\to-}}\Big| e_1 e_2 \dots e_7 |$$

| for separation

A difference with IEEE IEEE standard implements two's complement for the exponent. So there is no sign bit for exponent, but there is the concept of bias.

On our non-standard computer, let's check few things,

The largest normalized value

$$\begin{aligned} |0|111\dots1111|0|111\dots111| &= +0.\underbrace{11\dots1111}_{23-many} \times 2^{+\underbrace{11\dots11}_{7-many}} \\ &= \left(\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\dots+\frac{1}{2^{23}}\right)\times 2^{127} \\ &= \left(1-\frac{1}{2^{23}}\right)\times 2^{127} \\ &\approx 2^{127} \approx 17\times 10^{38} \end{aligned}$$

The smallest positive normalized value

$$|0|100\ldots000|1|111\ldots1| = 0.100000 \times 2^{-127} = 2^{-1} \times 2^{-127} \approx 10^{-42}$$

The smallest positive value

$$|0|000\ldots01|1|11\ldots11| = 2^{-23} \times 2^{-127} = 2^{-150} \approx 10^{-48}$$

The machine epsilon (ε_{mach}) $\texttt{fl}(1+\varepsilon_{mach}) > \texttt{fl}(1)$

$$\begin{array}{rl} & 0.10\ldots0 \times 2^1 \\ + & 0.10\ldots0 \times 2^{-127} \\ \hline & 0.10\ldots0 \times 2^1 \\ + & 0.\underbrace{000\ldots0}_{\text{23-many}} \to 1 \times 2^1 = 0 \\ \hline & 0.10\ldots0 \times 2^1 = 1 \end{array}$$

$$\begin{array}{rl} & 0.100\ldots01 \times 2^1 \\ - & 0.100\ldots00 \times 2^1 \\ \hline & \varepsilon_{mach} \end{array}$$

Sidebar $x \in [1, 1+\frac{\varepsilon_{mach}}{2})$ $\texttt{fl}(x) = \texttt{fl}(1)$

Double Precision Not IEEE standard. $F[b=2, M=52, E=10]$ + 2 sign bits.

Exercise Compute the numbers we checked for single precision

A decimal Computer

We are more familiar with arithmetic in base 10. So let's discuss same concepts here.

The number in this computer have the form

$$\pm 0.d_1 d_2 \ldots d_k \times 10^{\pm n}$$

$1 \le d_1 \le 9, 0 \le d_i \le 9 \qquad i = 2, 3, \ldots, k$

Take a number y. Let's define $f(y)$

$$y = \pm 0.d_1 d_2 \ldots d_k | d_{k+1} \ldots d_M \times 10^{\pm n}$$

$$\texttt{fl}(x) \begin{cases} \text{rounding} \\ \text{chopping} \end{cases}$$

Chopping

$$\texttt{fl}_{chop}(y) = \pm 0.d_1 d_2 \ldots d_k \times 10^{\pm n}$$

Rounding

$$\texttt{fl}_{round}(y) = \begin{cases} \texttt{fl}_{chop}(y) & \text{if } d_{k+1} < \frac{b}{2} = 5 \\ \pm 0.d_1 d_2 \ldots (d_k + 1) \times 10^{\pm n} & \text{if } d_{k+1} \geq \frac{b}{2} = 5 \end{cases}$$

The relative error in converting a real number into a FPN

$$\delta_x \frac{\texttt{fl}(x) - x}{x}$$

To find an upper bound to δ_x, we use ε_{mach}

Proposition (ε_{mach}) $\varepsilon_{mach} = b^{1-M}$ if chopping is used.
$\varepsilon_{mach} = \frac{1}{2} b^{1-M}$ if round is useds

Theorem For any FP system under chopping $|\delta_x| \leq \varepsilon_{mach}$

1.2.1 Floating Point Operations

Let $\oplus$ denotes the floating point addition

$$a \oplus b = \texttt{fl}(\texttt{fl}(a) \underbrace{+}_{\text{regular addition}} \texttt{fl}(b))$$

Since $\delta_x = \frac{\texttt{fl}(x) - x}{x}$
$\implies \texttt{fl}(x) = x(1 + \delta_x)$

$$\begin{aligned} a \oplus b &= (\texttt{fl}(a) + \texttt{fl}(b))(1 + \eta) \\ &= (a(1 + \delta_a) + b(1 + \delta_b))(1 + \eta) \\ &= (a + b + a\delta_a + b\delta_b)(1 + \eta) \\ &= a + b + a\delta_a + b\delta_b + a\eta + b\eta + a\delta_a\eta + b\delta_b\eta \end{aligned}$$

Remember $|\delta_x| \leq \varepsilon_{mach}$

And consider

$$\begin{aligned} |a \oplus b - (a + b)| &= |a\delta_a + b\delta_b + a\eta + b\eta + a\delta_a\eta + b\delta_b\eta| \\ &\approx |a + \delta_a + b\delta_b + a\eta + b\eta \qquad \text{since } (*) \\ &= |a(\delta_a + \eta) + b(\delta_b + \eta)| \\ &\leq |a(\delta_a + \eta|) + |b(\delta_b + \eta)| \\ &\leq |a|(|\delta_a| + |\eta|) + |b|(|\delta_b| + |\eta|) \leq 2\varepsilon_{mach}(|a| + |b|) \end{aligned}$$

$(*)$: $\quad |\delta_a\eta| \ll |a\delta_a|, \qquad |\delta_b\eta| \ll |a\delta_b|$

1.2.2 Sidebar

$f(x)$ happens in CPU Kernel Operations

$$a + (b \times c) \to a \oplus (b \otimes c) \to \text{ requires 2 rounding}$$

F used Multiply Add which can do $a + (b \times c)$ with one rounding

1.2.3 Some issues with FPS and FPOs

1. The number may not be exactly representable (tied to roundoff errors)

   Try to represent $(0.1)_10$ in $F[2, 5, 2]$

2. Non-associativity
   $$a + (b + c) \neq (a + b) = c$$

3. Non-distributiveness
   $$a \times (b + c) \neq ab + ac$$

4. Cancellation

5. FP underflow / overflow

6. Safe division

   Avoiding dividing be zero is called safe division. Still you might be dividing by a really small numbers and this may cause FP overflow.

How do we judge an algorithm? We check two things.

1. Conditioning of the algorithm
2. Stability of the algorithm

1.3 Condition Number

with respect absolute error.

$$\kappa_A = \frac{\|\Delta\vec{z}\|}{\|\Delta\vec{x}\|}$$

where Δz is change in output, Δx is change in input.

If $\kappa_A \approx 1$, we call the algorithm well-conditioned.
If $\kappa_A \gg 1$, we call the algorithm ill-conditioned.

Condition Number with respect to relative error

$$\kappa_R = \frac{\|\Delta\vec{z}\|/\|\vec{z}\|}{\|\Delta\vec{x}\|/\|\vec{x}\|}$$

Example Consider $y = \frac{x}{1-x}$

$$y + \Delta y = \frac{x + \Delta x}{1 - x - \Delta x}$$

$$|\Delta y| = \left| \frac{x + \Delta x}{1 - x - \Delta x} - y \right|$$

WLOG, $\Delta x > 0$

$$\left| \frac{x + \Delta x}{1 - x - \Delta x} - y \right| \leq \left| \frac{1}{1-x} + \frac{\Delta x}{1-x} - y \right| = \left| \frac{\Delta x}{1-x} \right|$$

$\implies \frac{|\Delta y|}{|\Delta x|} \leq \left| \frac{1}{1-x} \right|$

1.4 Stability of an Algorithm

If an algorithm does not propagate the error and does not produces large errors, it is called stable.

- E_0: error in first steps
- E_n: error after few steps

If $E_n \approx C_n E_0$ (i.e. linear growth in error) where C_n is a constant $\implies$ the algorithm is stable.

If $E_n \approx C^n E_0$ for $|C| > 1$ (i.e. exponential growth in error) $\implies$ the algorithm is unstable.

Example The algorithm is $p_n = \frac{10}{3} p_{n-1} - p_{n-2}$
(Exact Soln: $p_n = c_1 \left(\frac{1}{3}\right)^n + c_2 (3)^n$)
Take $p_0 = 1, p_2 = \frac{1}{3}$. Check stability.

Soln $E_0 \to$ error in computing p_2

$$\widetilde{p_3} = \frac{10}{3}(p_2 + E_0) - p_1 = \underbrace{\frac{10}{3} p_2 - p_1}_{=p_3} + \underbrace{\frac{10}{3} E_0}_{E_1}$$

$$\widetilde{p_4} = \frac{10}{3}\widetilde{p_3} - \widetilde{p_2} = \frac{10}{3} p_3 - p_2 + \frac{10}{3} E_1 - E_0$$

$E_n = \frac{10}{3} E_{n-1} - E_{n-2}$
$c_1 = -\frac{1}{8} E_0, c_2 = \frac{9}{8} E_0$

$$\implies E_n = \left(\frac{9}{8} 3^n - \frac{1}{8} \left(\frac{1}{3} \right)^n \right) E_0 \approx \frac{9}{8} 3^n E_0$$

CHAPTER 2

Root Finding Methods - A motivational example from policy making

A simple model on population growth.

$$\underbrace{N(t)}_{\text{population at time t}} = \underbrace{N_0}_{\text{initial population}} e^{\lambda t} + \frac{\nu}{\lambda}(e^{\lambda t} - 1)$$

where λ: growth rate, ν: immigration rate.

Given $N(t^*), N_0, \nu$, we can find λ such that

$$0 = N_0 e^{\lambda t^*} + \frac{\nu}{\lambda}(e^{\lambda t^*} - 1) - N(t^*)$$

We will prefer to use numerical methods.

Computational Problem

Given $f(x) = 0$, find x^* such that $|f(x^*)| < \varepsilon$ for given $\varepsilon > 0$.

2.1 Methods to find the roots

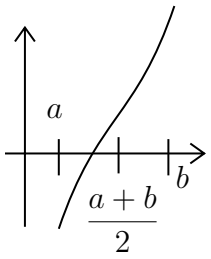
There are no methods which can guarantee to find the root always.

There are some methods which find the root under specific conditions.

2.1.1 Intermediate Value Theorem

Let f be continuous on $[a, b]$ and $c \in [f(a), f(b)]$ then $\exists x^* \in [a, b]$ such that $f(x^*) = c$.

How do we use IVT to find roots?



Algorithm
Given $f(x), a_0, b_0, i = 0$
Check if $f(a_i)f(b_i) < 0$ $(*)$
Compute $c_i = (a_i + b_i)/2$
Check if $f(a_i)f(c_i) < 0$
then $b_{i+1} = c_i$, otherwise $a_{i+1} = c_i$
$i = i + 1$
Repeat until convergence, goto $(*)$

Pros simple. guaranteed to converge

Cons has to be continuous. you should be able to find a, b such that $f(a)f(b) < 0$.

How do we measure "convergence" for Bisection Method?

Let's check

$$\underbrace{|c_k - a_k|}_{\text{or wlog } |c_k - b_k|} = \left| \frac{a_k + b_k}{2} - a_k \right| \overset{*}{=} \frac{|b_k - a_k|}{2}$$

(*) assume wlog $b_k = c_{k-1} = \frac{a_{k-1}+b_{k-1}}{2}$ $\qquad a_k = a_{k-1}$

$$= \frac{1}{2}\frac{|b_{k-1} - a_{k-1}|}{2} = \ldots = \left(\frac{1}{2}\right)^{k+1} |b_0 - a_0|$$

$$\begin{aligned}
\varepsilon > |c_k - a_k| &\implies \varepsilon > \left(\frac{1}{2}\right)^{k+1} |b_0 - a_0| \\
&\implies 2^{k+1} > \frac{|b_0 - a_0|}{\varepsilon} \\
&\implies k + 1 > \log_2 \left(\frac{|b_0 - a_0|}{\varepsilon}\right) \\
&\implies k > \log_2 \left(\frac{|b_0 - a_0|}{\varepsilon}\right) - 1
\end{aligned}$$

2.1.2 Fixed Point Method

Defn (Fixed Point)
Given a function $g(x)$, x^* is a fixed point of G iff $g(x^*) = x^*$.

Can we use this to come up with a method?

Problem Given f, find x^* such that $f(x^*) = 0$.

Assume we came up with functional $H(x)$ such that $H(0) = 0$, but necessarily non-zero everywhere else. $g(x) = x + H(f(x))$

Plug x^* in

$$g(x^*) = x^* + H(f(x^*)) = x^* + H(0) = x^*$$

$\implies$ the root of f is a fixed point of g.

$$x^* = g(x^*)$$

Let's take an initial guess x_0, and try

$$\begin{aligned} x_1 &= g(x_0) \\ x_2 &= g(x_1) \\ &\vdots \\ x_k &= g(x_{k-1}) \end{aligned}$$

Algorithm

Given g and x_0
$i = 1$
start of loop: Compute $x_i = g(x_{i-1})$
Check for convergence
$i = i + 1$
repeat, goto start of loop

Exercise $f(x) = x^3 + 4x^3 - 10$ $\quad \exists! x^* \in [1, 2]$ such that $f(x^*) = 0$. Look at the below functions

$$\begin{aligned} 0\cancel{x^*} &= \cancel{x^*} - x^{*3} - 4x^{*2} + 10 \\ g_1(x) &= x - x^3 - 4x^2 + 10 \\ g_2(x) &= \left(\frac{10}{x} - 4x\right)^{\frac{1}{2}} \\ g_4(x) &= \frac{1}{2}(10 - x^3)^{\frac{1}{2}} \\ g_4(x) &= \left(\frac{10}{x+4}\right)^{\frac{1}{2}} \end{aligned}$$

Confirm the fixed point is a root and use MATLAB to find it.

sidebar

$$\begin{aligned} x^3 + 4x^2 - 10 &= 0 \\ \implies x^4 &= 10 - 4x^2 \\ \implies x^2 &= \frac{10}{4} - 4x \\ x &= \underbrace{\sqrt{\frac{10}{x} - 4x}}_{:=g(x)} \end{aligned}$$

How do we knot if fixed point iterations will converge a head of time?

Let's check the diff between consecutive guesses

$$\begin{aligned} x_{k-1} &= g(x_{k-2}) \\ x_k &= g(x_{k-1}) \\ x_{k+1} &= g(x_k) \end{aligned}$$

$$|x_{k+1} - x_k| < |x_k - x_{k-1}|$$
$$|g(x_k) - g(x_{k-1})| < |x_k - x_{k-1}|$$

Defn (Contraction) Let g be a real-valued, continuous on $[a, b]$. If $\exists L \in (0, 1)$ such that

$$|g(x) - g(y)| \leq L|x - y| \qquad \forall x, y \in [a, b]$$

Then g is called contraction.

Manipulate the expression to get

$$\underbrace{\frac{|g(x) - g(y)|}{|x - y|}}_{\textit{looks like a derivative}} \leq L < 1$$

If $g(x)$ is differentiable we can prove that

$$L = \max_{x \in [a,b]} |g'(x)|$$

Contraction Mapping Theorem

Let g be a contraction on $[a, b]$ and $g(x) \in [a, b] \quad \forall x \in [a, b]$. Then

- g has unique fixed point x^* in $[a, b]$
- $\{x_k\}_{k=0}^{\infty}$ defined by algorithm f_x converges to x^* as $k \to \infty$ for any $x_0 \in [a, b]$.

It would be nice if we could only check $|g'(x)|$ at few points and know.

Corollary g is continuous on $[a,b], g(x) \in [a,b] \quad \forall x \in [a,b]$. Let $x^* = g(x^*)$ be the unique fixed point. Assume $\exists \delta > 0$ such that $g'(x)$ is constant in $[x^* - \delta, x^* + \delta]$. The sequence $\{x_i\}_{i=0}^{\infty}$ is defined by Algorithm f_x. Then

1. $|g'(x^*)| < 1$ then $\exists \varepsilon > 0$ such that $\{x_i\}$ converges to x^* for $|x_0 - x^*| < \varepsilon$.
2. $|g'(x^*)| > 1$ then $\{x_i\}$ will diverge.

2.1.3 Newton's Method

This method can be derived from the Taylor series. We require the function $f \in C^2[a,b]$.

Let's consider the Taylor Series of $f(x)$ around x_0

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

Let's plug the root x^* in

$$0 = f(x^*) = f(x_0) + f'(x_0)(x^* - x_0) + \underbrace{\frac{f''(\xi_{x^*})}{2}(x^* - x_0)^2}_{neglect}$$

$$\begin{aligned} 0 &\approx f(x_0) + f'(x_0)(x^* - x_0) \\ \underbrace{\delta_x}_{\substack{correction \\ term}} &= f(x_0) + f'(x_0)(x^* - x_0) \\ x_1 &= x_0 + \delta_x \end{aligned}$$

is equivalent to

$$\begin{aligned} x^* &\approx x_0 - \frac{f(x_0)}{f'(x_0)} \\ x_1 &= x_0 - \frac{f(x_0)}{f'(x_0)} \end{aligned}$$

$x_1 = g(x_0) = x_0 + H(f(x_0))$ where $H(f(x)) = -\frac{f(x)}{f'(x)}$

2.1.4 Problem with multiple roots

Multiple Root x^* is called a multiple root if $f(x^*) = 0$ and $f'(x^*) = 0$

In exact arithmetic, this turns out to be okay (we will discuss it later), but in finite precision it might cause an overflow.

Computing the derivative is not easy sometimes. f might be given as a blackbox function.

Solution: Approximate the derivative.

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

$$\text{Newton's} \quad x_{i+1} = x_i - \frac{f(x)}{f'(x_i)}$$

$$\text{Secant} \quad x_{i+1} = x_i - f(x_i)\left(\frac{f(x_i) - f(x_i - h)}{h}\right)^{-1}$$

$$\implies x_{i+1} = x_i - f(x_i)\left(\frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}\right)^{-1}$$

Notice that, now we need two initial guesses, x_0 and x_1. They should be close to the root.

Convergence Theorem for Newton's method

If $f(x^*) = 0, f'(x^*) \neq 0$ and $f \in C^2[x^* - \delta, x^* + \delta]$ for some $\delta > 0$ then the sequence $\{x_i\}_{i=0}^{\infty}$ converges to x^* quadratically.

An extension $f'(x^*) = 0$, then $\{x_i\}$ converges to x^* linearly.

Convergence theorem for secant method

If $f(x^*) = 0$, $f'(x^*) \neq 0$ and $f \in C^2[x^* - \delta, x^* + \delta]$ for some $\delta > 0$, then $\{x_i\}$ converge to x^* with rate $\frac{1}{2}\left(1 + \sqrt{5}\right)$.

Convergence Rate

$$\underbrace{e_i}_{\substack{\text{signed error} \\ \text{at iteration i}}} = x_i - x^*$$

$$\frac{|e_{i+1}|}{|e_i|^q} = \underbrace{c}_{\substack{convergence \\ constant}}$$

where q is convergence rate.

Consider $|e_{i+1}| = c|e_i|^q$

- $q = 1$, then we say iterations converge linearly.
- $q = 2$, then we say that iterations converge quadratically.

Why rate of convergence is important?

Say $f(x)$ takes 1 second to evaluate Bisection method takes 28 iterations to converge $\implies$ 28 seconds.

Secant Method takes 7 iterations

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

Observe that at each iteration (except the first) we only evaluate f once $\implies$ 8 seconds.

2.1.5 Comparison of the methods

	pros	cons
Bisection	• Guaranteed convergence • Linear-like rate • No need f'	• a, b such that $f(a)f(b) < 0$ • f may not satisfy the condition of IVT
Fixed Pt.	• Linear or higher rate • no need f'	• convergence is not guaranteed • we need a good g and x_0
Newton's	Quadratic convergence $\star$	• Convergence is not guaranteed • We need a good x_0
Secant	• Rate of convergence is $\frac{1+\sqrt{5}}{2}$ • no need f'	• Convergence not guaranteed • need to good x_0 and x_1

$\star$: except at multiple roots. Then it degrades to linear

Sidebar

$$x_2 = x_1 - \frac{\overbrace{f(x_1)}^{solve}}{f(x_1) - f(x_0)}(x_1 - x_0)$$

$$x_3 = x_2 - \frac{\overbrace{f(x_2)}^{solve}(x_1 - x_0)}{f(x_2) - f(x_1)}$$

CHAPTER 3

Numerical Linear Algebra

The problem Given a matrix $A \in \mathbb{R}^{n \times n}$ and a vector $\vec{b} \in \mathbb{R}^n$, find $\vec{x} \in \mathbb{R}^n$ such that

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

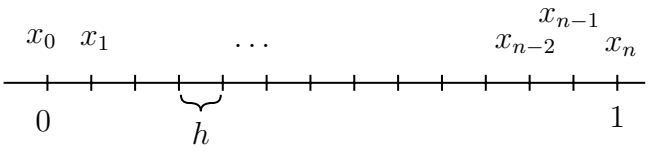
The Motivation

Find $u \in C^\infty$ for given $f \in C$ such that

$$-u'' + u = f \text{ on } [0, 1]$$

This problem is not exactly solvable (except under limited conditions). Let's look for an approximation solution.

Approximate u''. Let's introduce



Define $u_i = u(x_i)$

$$u''(x_i) = \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2}$$

(Central difference approx)

$$\begin{bmatrix} \frac{2}{n^2}+1 & -1 & 0 & \dots & \dots & 0 \\ -1 & \frac{2}{n^2}+1 & -1 & 0 & \dots & 0 \\ \vdots & \ddots & & & & \\ \vdots & \ddots & \dots & \dots & \dots & -1 \\ 0 & \dots & \dots & \dots & -1 & \frac{2}{n^2}+1 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ \vdots \\ f_n \end{bmatrix}$$

We need ways to solve linear systems efficiently and accurately.

Is this problem solvable?

Theorem Existence and Uniqueness

Consider $Ax = b$, $A \in \mathbb{R}^{n \times n}$

- If A has full row/column space $\iff$ $\det(A) \neq 0$ $\iff$ A is invertible
- $\det(A) =, b \in Range(A)$, then there exist infinitely many solutions
- $\det(A) = 0, b \notin Range(A)$, then $\nexists x$

$C(A) = span\{A'\}$ s.t. A' are linearly independent.

$|C(A)| = n \implies$ full rank

$$Range(A) = \{v | v = Ax \text{for some } x\} = C(A)$$

How do we solve it?

3.1 Gaussian Elimination

The most robust solution method known, but not efficient.

3.1.1 Modification: LU factorization

Defn Upper/lower triangular, diagonal (trivial definition)

Denote d_i by unit diagonal entries. If $d_i = 1$, then it is called unit upper/lower triangular. We call a_{ij} off diagonal entries where $i \neq j$.

Given L a unit lower triangular matrix and b, solve

$$L\vec{x} = \vec{b}$$

$$\begin{bmatrix} 1 & & 0 & \\ & \ddots & & \\ & & 1 & \\ & & & \ddots \\ & L_{(i,j)} & & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ \vdots \\ \vdots \\ b_n \end{bmatrix}$$

$$L_{11}x_1 = b_1 \implies x_1 = \frac{b_1}{L_{11}}$$
$$L_{21}x_1 + x_2 = b_2 \implies x_2 = b_2 - L_{21}x_1$$

This process is called forward substitution. Similarly, if U is an unit upper triangular matrix is given. We can solve $U\vec{x} = \vec{b}$ by backward substitution.

So we can find L, U such that $A = LU$.

Consider

$$A\vec{x} = \vec{b}$$
$$L\underbrace{U\vec{x}}_{\vec{y}} = \vec{b}$$
$$\text{solve } L\vec{y} = \vec{b}$$
$$\text{solve } U\vec{x} = \vec{y}$$

This is called Gaussian elimination (LU Decomposition variant)

Example

$$\begin{pmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 7 \\ 4 \\ 6 \end{pmatrix}$$

$$M^{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 0.3 & 1 & 0 \\ -0.5 & 0 & 1 \end{pmatrix}$$

$$A^{(1)} = M^{(1)}A = \begin{pmatrix} 10 & -7 & 0 \\ 0 & -0.1 & 6 \\ 0 & 2.5 & 5 \end{pmatrix}$$

$$M^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 25 & 1 \end{pmatrix}$$

$$\underbrace{A^{(2)}}_{=U} = M^{(2)}A^{(1)} = \begin{pmatrix} 10 & -7 & 0 \\ 0 & -0.1 & 6 \\ 0 & 0 & 155 \end{pmatrix}$$

$$M^{(2)}M^{(1)}A = U$$
$$\implies A = \left(M^{(2)}M^{(1)}\right)^{-1} U$$
$$= \left(M^{(1)}\right)^{-1}\left(M^{(2)}\right)^{-1} U$$

Inverses of lower triangular element matrices

- Keep the diagonal matrices same
- Flip the signs off-diagonal

$$\left(M^{(2)}\right)^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -25 & 1 \end{pmatrix}$$

$$\left(M^{(1)}\right)^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ -0.3 & 1 & 0 \\ 0.5 & 0 & 1 \end{pmatrix}$$

Combination property

Again, we are taking advantage of structures of $\left(M^{(1)}\right)^{-1}$ and $\left(M^{(2)}\right)^{-1}$

$$\begin{aligned} M &= \left(M^{(1)}\right)^{-1}\left(M^{(2)}\right)^{-1} \\ M_{ij} &= \left(M^{(1)}\right)^{-1}_{ij} + \left(M^{(2)}\right)^{-1}_{ij} \quad \text{for } i \neq j \\ M_{ii} &= 1 \end{aligned}$$

$$\underbrace{M}_{=L} = \begin{pmatrix} 1 & 0 & 0 \\ -0.3 & 1 & 0 \\ 0.5 & -25 & 1 \end{pmatrix}$$

$$A = LU$$

we can solve

$$L\vec{y} = \begin{pmatrix} 7 \\ 4 \\ 6 \end{pmatrix} \implies \vec{y} = \begin{pmatrix} 7 \\ 6.1 \\ 155 \end{pmatrix}$$

Now solve

$$U\vec{x} = \vec{y}$$

3.1.2 Pivoting

LU decomposition with pivoting is the most robust version of Gaussian Elimination.

Pivoting is the action of permuting a matrix s.t. all diagonal entries are non-zero (all diagonal entries are the largest entries in the row)

Defn Permutation matrix P is a matrix obtained from the identity matrix by swapping rows.

Lemma Given P a permutation matrix $P^{-1} = P^T$.

Proof Exercise

Theorem Existence of L, U
For all $A \in \mathbb{R}^{n\times n}$, there exists P a permutation matrix L unit lower triangular matrix and U upper triangular matrix s.t.

$$PA = LU$$

Exercise Apply GE[1] without pivoting to the system

$$0.3000 \times 10^{-3}x_1 + 0.5914 \times 10^{-2}x_2 = 0.5917 \times 10^{2}$$
$$0.5291 \times 10^{1}x_1 - 0.6130 \times 10^{1}x_2 = 0.4678 \times 10^{1}$$

Using four digit arithmetic with rounding. Compare to the exact solution. $x_1 = 10, x_2 = 1$.

Complete Pivoting

At each step of GE, we scan the matrix $A^{(i)}$ and find the largest entry in magnitude to the diagonal $a_{ii}^{(i)}$ by row and column permutations.

Note Column permutations also permute the solution.

$$0.5914 \times 10^{2}x_2 + 0.3000 \times 10^{-3}x_1 = sth$$
$$-0.6130 \times 10^{1}x_2 + 0.5291 \times 10^{1}x_1 = sth$$

Partial Pivoting

Complete pivoting is expensive. Maybe we can relax the condition a little bit to obtain cheaper but sill robust method.

Just compare against the column i of matrix $A^{(i)}$, then use row permutations to bring the pivot to the diagonal entry $a_{ii}^{(i)}$.

Note with row permutations, right hand side changes.

$$0.5291 \times 10^{1}x_1 - 0.6130 \times 10^{1}x_2 = 0.4678 \times 10^{1}$$
$$0.3000 \times 10^{-3}x_1 + 0.5914 \times 10^{2}x_2 = 0.5917 \times 10^{2}$$

More explanation

Given full linear system

$$Ax = b$$

which we don't know how to solve. But we know how to solve triangular systems. It would be nice if $A = LU$ and $LU\vec{x} = \vec{b}$. Create elementary matrices $M^{(i)}$, and create $A^{(i)}$ such that

$$A^{(i)} = M^{(i)}A^{(i-1)}, \qquad A^{(0)} = 0$$

[1] Gaussian Elimination

and

$$\textit{upper triangular} \leftarrow A^{(n-1)} = M^{(n-1)}A^{(n-2)}$$

$$L = \left(M^{(n-1)}M^{(n-2)}\dots M^{(1)}\right)^{-1} \quad \to \textit{Lower triangular}$$

Given

$$A = \begin{bmatrix} a_{11} & \hat{a}_{12} & \dots & \hat{a}_{1n} \\ \vdots & \ddots & & \vdots \\ \hat{a}_{n1} & \dots & \hat{a}_{nn} & \end{bmatrix}$$

Consider

$$M^{(1)} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ -\frac{\hat{a}_{21}}{a_{11}} & 1 & & \\ \vdots & & 1 & \\ \vdots & & & 1 \\ -\frac{\hat{a}_{n1}}{a_{11}} & & & 1 \end{bmatrix}$$

a_{11} is small, $\hat{a}_{21} = a_{21} + \eta$

$$A^{(1)} = \begin{bmatrix} a_{11} & \hat{a}_{12} & \hat{a}_{13} & \dots & \hat{a}_{1n} \\ 0 & \hat{a}_{22} - \frac{\hat{a}_{21}-\hat{a}_{12}}{a_{11}} & \hat{a}_{23} - \frac{\hat{a}_{21}-\hat{a}_{13}}{a_{11}} & \dots & \hat{a}_{2n} - \frac{\hat{a}_{21}\hat{a}_{1n}}{a_{11}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & & & \end{bmatrix}$$

$$\hat{a}_{ij} = \texttt{fl}(a_{ij}) = a_{ij} + \eta_{ij} = a_{ij} + \eta$$

η_{ij} may not be the same but just for simplification

Issue that might occur

1. $$\hat{a}_{2n} - \frac{(a_{21}+\eta)(a_{1n})}{a_{11}} = \hat{a}_{2n} - \frac{a_{21}a_{1n}}{a_{11}}$$
$$= \underbrace{\frac{a_{21}\eta + a_{1n}\eta}{a_{11}}}_{\text{not negligible}} - \underbrace{\frac{\eta^2}{a_{11}}}_{\textit{neglect}}$$

   η is usually small, η^2 is smaller.

2. $$\hat{a}_{22} - \frac{\hat{a}_{21}\hat{a}_{12}}{a_{11}}$$

   say a_{22} is not close to $\frac{a_{21}a_{12}}{a_{11}}$, but $\hat{a}_{22}$ might be close to $\frac{\hat{a}_{21}\hat{a}_{12}}{a_{11}} \implies$ catastrophic cancellation.

3. $\frac{\hat{a}_{21}}{a_{11}} \implies$ safe division if $|a_{11}|$ is really small.

- Complete pivoting avoids all above issues as a_{11} is the largest entry, the 3 item can be negligible (< 1), always useful but expensive.
- Partial pivoting similarly avoids the third issue but not for 1 and 2m useful for some situations and cheaper.

3.1.3 Algorithm and Computational Cost (without pivoting)

Phase 1 Obtaining L and U

Initialize $L = I$ and $U = A$

```
1  for p = 1: n-1                          -> pivot element
2    for r = p+1 : n                      -> scanning the rows below the pivot
3      m = -U(r,p) / U(p,p)               -> construction of M^(p)
4      U(r,p) = 0                         -> eliminating from U
5      for c = p+1 : m                    -> scanning the columns right
6        U(r,c) = U(r,c) + mU(p,c)        -> column update
7      endfor
8    L(r,p) = -m                          -> combination and inversion properties
9    endfor
10 endfor
```

Phase 2 Forward Substitution

Phase 3 Backward Substitution

Algorithm (Most general)

1 **foreach** *stage* $p = 1 : n - 1$ **do**
2 | Create $M^{(p)}$
3 | Compute $A^{(p)} = M^{(p)} A^{(p-1)}$
4 **end**
5 Set $U = A^{(n-1)}$
6 Set $L = \left(M^{(n-1)} \dots M^{(1)}\right)^{-1}$

Algorithm 1: general algorithm

Algorithm (LU factorization)

1 Initialize $L = I, U = A$
2 **for** $p = 1 : n - 1$ // for each stage
3 **do**
4 | % pivot = p // without pivoting (comment)
5 | **for** $r = p + 1 : n$ // loop over the rows
6 | **do**
7 | | $\cancel{M(r,p)} \overset{m}{=} \frac{U(r,p)}{U(p,p)}$ // create $M^{(p)}$ (cost A)
8 | | /* partially $M^{(p)}A^{(p-1)}$ */
9 | | $U(r,p) = 0$ // eliminating the netries below the current diagonal (cost 0)
10 | | **for** $c = p + 1 : n$ // loop over the columns
11 | | **do**
12 | | | $U(r,c) = U(r,c) + \cancel{M(r,p)} \overset{m}{U}(p,c)$ // Cost $A^* > A$
13 | | **end**
14 | | $L(r,p) = \cancel{M(r,p)} \overset{m}{}$ // Combination properly: inversion property (cost 0)
15 | | /* partially $M^{(p)}A^{(p-1)}$ (encloses the previous comment) */
16 | **end**
17 **end**

Algorithm 2: LU factorization

$$
\begin{aligned}
Total\ Cost &= \sum_{p=1}^{n-1}\left(\sum_{r=p+1}^{n}\left(A + \sum_{c=p+1}^{n} A\right)\right) \\
&= \sum_{p=1}^{n-1}\left(\sum_{r=p+1}^{n} A + \sum_{r=p+1}^{n}\sum_{c=p+1}^{n} A\right) \\
&= \sum_{p=1}^{n-1}\left((n-(p+1)+1)A + \sum_{r=p+1}^{n}(n-(p+1)+1)A\right) \\
&= \sum_{p=1}^{n-1}\left((n-p)A + \sum_{r=p+1}^{n}(n-p)A\right) \\
&= \sum_{p=1}^{n-1}((n-p)A + (n-p)^2 A) \\
&= \sum_{p=1}^{n-1}(An - Ap + An^2 - 2Anp + Ap^2) \\
&= An(n-1) - \frac{An(n-1)}{2} + An^2(n-1) - 2An\frac{n(n-1)}{2} + A\frac{n(n-1)(2n-1)}{6} \\
&\in O(n^3)
\end{aligned}
$$

Algorithm (forward substitution): Solve $Ay = b$

```
1 Set y = b
2 for r = 2 : n do
3 |   for c = 1 : r − 1 do
4 |   |   y(r) = y(r) − L(r, c)y(c)
5 |   end
6 end
```

Algorithm 3: forward substitution

$$Cost = \sum_{r=2}^{n}\sum_{c=1}^{n-1} A = \sum_{r=2}^{n} A(r-1) = A\frac{n(n-1)}{2} \in O(n^2)$$

	LU factorization	$O(n^3)$
	Forward substitution	$O(n^2)$
+	Backward substitution	$O(n^2)$
		$O(n^3)$

```
1 Initialize L = I, P = I, U = A
2 for p = 1 : n − 1                                   // loop over stages
3  do
4 |   Find the largest entry |a_ip| in column p       // O(n)
5 |   Swap rows i and p of U                          // O(n)
6 |   Swap rows i and p of L                          // O(n)
7 |   Swap rows i and p of P                          // O(n)
8 |   for ... do
9 |   |   /* same as without pivoting                 */
10|   end
11 end
```

Algorithm 4: LU with partial pivoting

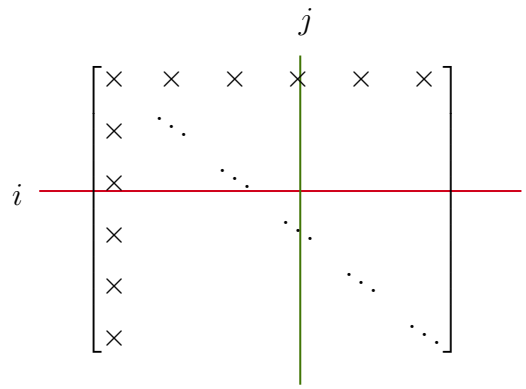
3.1.4 Determinants

Defn Determinant of a matrix $A \in \mathbb{R}^{n\times n}$ is given by

$$\det(A) = \sum_{j=1}^{n}(-1)^{i+j} a_{ij} \det(A_{ij})$$

for fixed i, where A_{ij} is the principle submatrix.

Defn The principal submatrix A_{ij} of the matrix A is the matrix obtained by deleting row i and column j from the matrix A.



Identities for determinants

- $\det(BC) = \det(B)\det(C)$, $(B, C \in \mathbb{R}^{n\times n})$
- U upper triangular $\det(U) = \prod u_{ii}$
- L lower triangular $\det(L) = \prod \ell_{ii}$
- P permutation matrix $\det(P) = \begin{cases} +1 & \text{if even number of swaps} \\ -1 & \text{otherwise} \end{cases}$

Proposition $\det(A) \neq 0$ iff $\det(U) \neq 0$ $(PA = LU)$

Proof

$$\begin{aligned} &\det(PA) = \det(LU) \\ \implies &\det(P)\det(A) = \det(L)\det(U) \\ &\det(A) = \frac{1}{\underbrace{\det(P)}_{\pm 1}} + \underbrace{\det(L)}_{1}\det(U) \\ \implies &\det(A) = \pm\det(U) \end{aligned}$$

Corollary If $\det(A) \neq 0$, then we can solve $Ax = b$ by LU factorization.

3.2 Condition of the Problem

Reminder The condition number w.r.t absolute error is

$$\frac{\left\| \overbrace{\Delta z}^{Change\ in\ output} \right\|}{\left\| \underbrace{\Delta x}_{change\ in\ output} \right\|}$$

Problem find x given $Ax = b$ where A, b are inputs.

Equivalently, find $x = f(A, b)$.

How do we define the norm of a matrix?

The natural matrix p-norm

For $p = 1, 2$ or ∞

$$\|A\|_p = \max_{\|\vec{x}\|_p \neq 0} \frac{\|A\vec{x}\|_p}{\|\vec{x}\|_p}$$

Reminder: Vector Norms

$$\|x\|_1 = \sum_i |x_i|$$
$$\|x\|_2 = \sqrt{\sum_i x_i^2}$$
$$\|x\|_\infty = \max_i |x_i|$$

3.2.1 Matrix Norms

$$\|A\|_1 = \max_{\|x\| \neq 0} \frac{\|Ax\|_1}{\|x\|_1}$$

Consider (proof of prop. 3.4)

$$\begin{aligned}
\|Ax\|_1 &\overset{\substack{defn\ of \\ mat\cdot vec}}{=} \|\sum_i A_i x_i\| \qquad A_i\text{: ith column of } A \\
&\leq \sum_i \|A_i x_i\|_1 \qquad \text{triangle ineq. for vector norms} \\
&= \sum_i |x_i| \|A_i\| 1 \qquad \text{homogeneity, } \|\alpha\vec{b}\| = |\alpha|\|\vec{b}\| \\
&\leq \|A_j\|_1 \sum_i |x_i| \qquad \exists j, \forall i,\ \|A_j\| \geq \|A_i\| \\
&= \|A_j\|_1 \|x\|_1 = \|A\|_1 \|x\|_1
\end{aligned}$$

Let's pick $\vec{x} = \vec{e^j} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, where j^{th} row is 1.

Consider

$$\|A\vec{e^j}\| = \|\sum_i A_i \vec{e}_i^j\| = \|A_j\|$$

$$\implies \frac{\|Ax\|_1}{\|x\|_1} \leq \|A_j\|_1$$

and $\vec{x} = \vec{e^j}$ maximize $\frac{\|Ax\|_1}{\|x\|_1}$

$$\implies \|A\|_1 = \max_j \sum_i \|a_{ij}\|, \quad \text{maximum absolute column sum}$$

Prop Matrix norms satisfy

$$\|A\vec{x}\|_p \leq \|A\|_p \|\vec{x}\|_p$$

Matrix norms are norms

$$\|A\|_p \max_{\|x\|\neq 0} \frac{\|Ax\|_p}{\|x\|_p}$$

if $A = 0$, then $\|A\|_p = 0$, trivial
if $\|A_p\| = 0$, then $\|A\| = 0$ trivial (use of properties of matrix norms)
$\|A\|_p \geq 0$ trivial $\implies$ confirms $\|A\|_p \geq 0$

and $\|A\|_p = 0$ iff $A = 0$

$$\begin{aligned}
\|\alpha A\|_p &= \max_{\|x\|\neq 0} \frac{\|\alpha Ax\|_p}{\|x\|_p} \\
&= \max_{\|x\|\neq 0} \frac{|\alpha| \|Ax\|_p}{\|x\|_p} \\
&= |\alpha| \max_{\|x\|\neq 0} \frac{\|Ax\|_p}{\|x\|_p} \\
&= |\alpha| \|A\|_p
\end{aligned}$$

$\implies$ second condition is satisfied.

It satisfies triangle inequality and the proof is in course notes (Prop 3.5).

3.2.2 Condition at the problem $A\vec{x} = \vec{b}$

Example $\delta = 10^{-4}$

$$\overbrace{\begin{bmatrix} 1 & 1 \\ 1-\delta & 1+\delta \end{bmatrix}}^{A} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \implies \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\Delta A = \begin{bmatrix} 0 & 0 \\ \delta & 0 \end{bmatrix}$$

$$(A + \Delta A)\vec{x} = \vec{b}$$

$$\begin{bmatrix} 1 & 1 \\ 1 & 1+\delta \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

$$\left[\begin{array}{cc|c} 1 & 1 & 2 \\ 1 & 1+\delta & 2 \end{array}\right] \xrightarrow{-R_1+R_2} \left[\begin{array}{cc|c} 1 & 1 & 2 \\ 0 & \delta & 0 \end{array}\right]$$

$$\implies \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$$

Let's consider the perturbed system

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

Case 1 Assume $\Delta A = 0$

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

A is invertible,

$$\Delta x = A^{-1}\Delta b$$

Take norms of both sides

$$\begin{aligned} \|\Delta x\| &= \|A^{-1}\Delta b\| \\ &\leq \|A^{-1}\| \cdot \|\Delta b\| \qquad \textit{prop } 3.3 \\ \frac{\|\Delta x\|}{\|\Delta b\|} &\leq \|A^{-1}\| \end{aligned}$$

$$Ax = b \implies \|Ax\| = \|b\| \implies \|A\| \cdot \|x\| \geq \|b\| \implies \|x\| \geq \frac{\|b\|}{\|A\|}$$

Let's consider

$$\frac{\|\Delta x\|}{\|x\|} \leq \|A^{-1}\| \cdot \|A\| \cdot \frac{\|\Delta b\|}{\|b\|}$$

$$\underbrace{\frac{\|\Delta x\|/\|x\|}{\|\Delta b\|/\|b\|}}_{x_R} \leq \underbrace{\|A\| \cdot \|A^{-1}\|}_{\substack{\text{defn} \\ \text{The condition} \\ \text{number of} \\ \text{a matrix}}}$$

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

Case 2: $\Delta b = 0$, $\Delta A \neq 0$

$$\begin{aligned} (A + \Delta A)(x + \Delta x) &= b \\ \Delta x &= -A^{-1}\Delta A x - A^{-1}\Delta A \Delta x \\ \|\Delta x\| &= \| - A^{-1}\Delta A x - A^{-1}\Delta A \Delta x\| \\ &\leq \| - A^{-1}\Delta A x\| + \|A^{-1}\Delta A \Delta x\| \\ &\leq \|A^{-1}\| \cdot \|\Delta A\| \cdot \|x\| + \|A^{-1}\| \cdot \|\Delta A\| \cdot \|\Delta x\| \\ \implies \|\Delta x\| &\leq \|A^{-1}\|\|\Delta A\|(\|x\| + \|\Delta x\|) \\ \implies \frac{\|\Delta x\|}{\|x\| + \|\Delta x\|} &\leq \|A^{-1}\|\|\Delta A\|\frac{\|A\|}{\|A\|} \\ \implies \frac{\|\Delta x\|/(\|x\| + \overbrace{\|\Delta x\|}^{neglible})}{\|\Delta A\|/\|A\|} &\leq \|A^{-1}\|\|A\| \qquad \text{Since } \|\Delta x\| \ll \|x\| \\ \implies \frac{\|\Delta x\|/\|x\|}{\|\Delta A\|/\|A\|} &\leq \kappa(A) \end{aligned}$$

Case 3 Assignment 2, Q5

Prop 3.7 $A \in \mathbb{R}^{n\times n}$

$$\kappa_2(A) = \|A^{-1}\|_2\|A\|_2 = \frac{\delta_{\max}(A)}{\delta_{\min}(A)}$$

proof in the course note

3.2.3 Stability of LU algorithm

We discussed it right after partial and complete pivoting.

Moral of the story Pivoting makes LU algorithm more stable.

Midterm is up until here!

3.3 Iterative methods for solving $Ax = b$

Recall that GE solves $Ax = b$ as long as A is invertible at the cost of $O(n^3)$. Can we put some conditions on A and reduce the complexity?

In exact arithmetic, GE solves $Ax = b$ exactly $\implies \|b - Ax_s\| = 0$

Do we need to be zero?

So far, we (implicitly) were talking about full matrices.

Defn A matrix A is called a full matrix iff almost all of its entries are nonzero (Dense Matrices)

Defn (Sparse Matrix I) A matrix $A \in \mathbb{R}^{m\times n}$ is called sparse iff number of nonzeros at each row are $O(1)$ w.r.t n.

Defn (Spares Matrix II)
$A \in \mathbb{R}^{n\times n}$ is called a sparse matrix iff number of nonzeros in A is much smaller than n^2.

Why sparse matrices?

A is full, what is the complexity of mat-vec? $O(n^2)$.

A is sparse, there are $O(n)$ nonzero's in A. What is the complexity of matrix-vec? $O(n)$

Why mat-vec?

Let's $\underbrace{A}_{sparse} = M - N$ where M is invertible, sparse and N is sparse.

$$\begin{aligned} Ax &= b \\ (M-N)x &= n \\ &\vdots \\ x &= M^{-1}(b + Nx) \end{aligned}$$

Fixed point problem. The type of iterative solvers we are going to cover are fixed point type iterations.

How do we pick M?

$$A = \begin{bmatrix} \ddots & & -F & \\ & \ddots & & \\ & & D & & \\ & & & \ddots \\ & -E & & & \ddots \end{bmatrix}$$

$$A = \underbrace{D}_{=M} \underbrace{-E-F}_{=-N}$$

Sidebar Computational cost of solving $Dx = b$

$$b = \begin{bmatrix} \frac{1}{\alpha_{11}x_1} \\ \vdots \\ \frac{1}{\alpha_{i1}x_i} \\ \vdots \\ \frac{1}{\alpha_{n1}x_n} \end{bmatrix} \implies O(n)$$

$$\implies x = M^{-1}(b + Nx) \;\; \& \;\; M = D, N = E + F$$

cost: $O(n) + O(n) + O(n) = O(n)$
continue until convergence

$$\vec{x}_i = M^{-1}(\vec{b} + N\vec{x}_{i-1})$$

total cost will depend on convergence criteria and the matrix A.

3.3.1 Two types of FP methods

$M = D, N = E + F \implies$ Gauss-Jacobian method

$$\vec{x}_i = D^{-1}([E + F]\vec{x}_{i-1} + b) \rightarrow \text{matrix form}$$

Component wise form:

Let $x^{(i)}$ denote the i-th entry of vector $\vec{x}$.

$$x_i^{(k)} = \frac{1}{D_{kk}} \left(b^{(k)} - \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj} x_{i-1}^{(j)} \right) \quad k = 1, 2, \ldots, n$$

$$M = D - E \ (or \ D - F), \quad N = F \ (or \ E) \implies \text{Gauss-Seidel Method}$$

matrix form:

$$\vec{x}_i = (D - E)^{-1}(F x_{i-1} + b)$$

Exercise: what is the cost of solving $Lx = b$?$Ux = b$?

Component wise form:

$$x_i^{(k)} = \frac{1}{D_{kk}} \left(-\sum_{j=1}^{k-1} a_{kj} x_i^{(j)} - \sum_{j=k+1}^{n} a_{kj} x_{i-1}^{(j)} + b_k \right)$$

visualization

$$\begin{bmatrix} a_{11} & & 0 & & \\ \vdots & a_{22} & & & \\ \vdots & & \ddots & & \\ \vdots & & & \ddots & \\ a_{n1} & \ldots & \ldots & \ldots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1^{new} \\ x_2^{new} \\ \vdots \\ \vdots \\ x_n^{new} \end{bmatrix} = \begin{bmatrix} 0 & a_{12} & \ldots & \ldots & a_{1n} \\ & 0 & \ldots & \ldots & \vdots \\ & & \ddots & & \vdots \\ & & & \ddots & a_{n-1,\ n} \\ & & 0 & & 0 \end{bmatrix}$$

$$x_1^{new} = \frac{1}{a_{11}}(a_{12} x_2^{old} + a_{13} x_3^{old} + \ldots + a_{1n} + x_n^{old})$$

$$a_{21} x_1^{new} + a_{22} x_2^{new} = a_{23} x_3^{old} + \ldots + a_{2n} x_n^{old} + b_2$$

$$\implies a_{22} x_2^{new} = -a_{22} x_2^{new} + a_{23} x_3^{old} + \ldots + a_{2n} x_n^{old} + b_2$$

General convergence result

$$M\vec{x} = N\vec{x} + b$$

$$\vec{x}_i = M^{-1} x_{i-1} M^{-1} b$$

Assume $\vec{x}_i \to \vec{x}^*$,does $\vec{x}^*$ solve $Ax = b$?

$$\vec{x}^* = M^{-1} N \vec{x}^* + M\vec{b} \implies (\underbrace{M - N}_{A})\vec{x}^* = \vec{b}$$

So, yes.

When do these iterations converge?

$$\begin{aligned}\vec{x}_i &= M^{-1}N\vec{x}_{i-1} + M^{-1}b \\ &= M^{-1}(M-A)\vec{x}_{i-1} + M^{-1}b \\ &= \underbrace{(I - M^{-1}A)}_{G \text{ iteration matrix}}\ \vec{x}_{i-1} + \vec{f}\end{aligned}$$

Theorem Let G be a square matrix s.t. $\|G\| < 1$ in any matrix norm. Then $I - G$ is non-singular and the iterations converge for any $\vec{x}_0$ initial guess. (Proof is exercise)

Theorem If A is strictly diagonally dominant, then GJ, GS iterations converge.

Defn A is called strictly diagonally dominant iff for any i

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$$

Correction

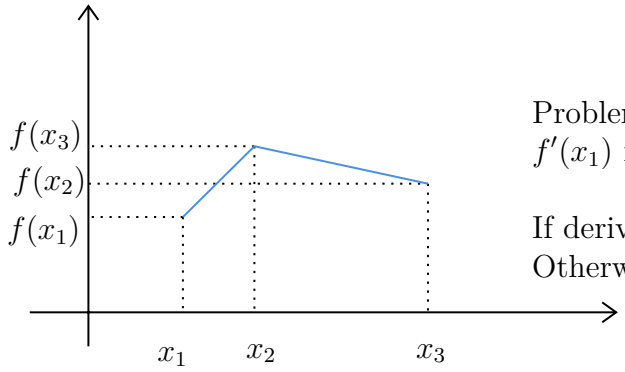
Intermission (p21): Vector Norms 1-norm is NOT induced by an inner product.

CHAPTER 4

Interpolation

Motivating Problem Take an experiment. Observe a value at discrete time points. How do you find the value for time points in between.

Actual Problem Given $\{x_i\}$ with $x_i \neq x_j$ for $i \neq j$ and $\{f(x_i)\}$ find a continuous function y, s.t. $y(x_i) = f(x_i)$.



Problem with straight line connections:
$f'(x_1)$ is not defined.

If derivative is not needed, good.
Otherwise, not so much

Is there a polynomial p which approximates a continuous f for any given tolerance? If there is, how do we find it?

Weierstrass Approximation Theorem

Given a continuous function $f(x)$ in $[a, b]$ there exists a polynomial $p(x)$ for given $\varepsilon > 0$ s.t. $|f(x) - p(x)| < \varepsilon$ for all $x \in [a, b]$.

4.1 Polynomial Basis

4.1.1 Monomial Basis

$\Phi = \{1, x, x^2, \ldots\}$
$\Phi_k = \{1, x, x^2, \ldots, x^k\}$.

We can write any polynomial p as

$$p(x) = \sum_{i=0}^{j} \alpha_i \varphi_i = \sum_{i=0}^{k} \alpha_i x^i$$

Let $p(x) \approx f(x)$ and p interpolates f $(p(t_i) = f(t_i))$

$$\alpha_0 t_i^0 + \alpha_1 t_i^1 + \ldots + \alpha_k t_i^k = p(t_i) \implies \underbrace{\begin{bmatrix} t_0^0 & t_0^1 & t_0^2 & \ldots & t_0^k \\ t_1^0 & t_1^1 & t_1^2 & \ldots & t_1^k \\ \vdots & & & & \\ t_k^0 & t_k^1 & t_k^2 & \ldots & t_k^k \end{bmatrix}}_{\text{Vandermond Matrix } (V)} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_k \end{bmatrix} = \begin{bmatrix} f(t_0) \\ f(t_1) \\ \vdots \\ f(t_k) \end{bmatrix}$$

Possible issues: as k very large, t_i^k can be either very large or small causing over/underflow

Theorem $\det(V) \neq 0$

Theorem p is unique (come from monomial basis)

Overflow Ex Introduce Inf in the matrix V.

Underflow Ex $\begin{bmatrix} 1 & 0.15 & 0 & 0 & \ldots & 0 \\ 1 & 0.2 & 0 & 0 & \ldots & 0 \\ 1 & 0.1 & 0 & 0 & \ldots & 0 \end{bmatrix}$ if rows are close to each other, determinant = 0, no solution

First issue

The Vandermond matrix is ill-conditioned.

This issue can be partially solved by introducing the auxiliary variable c and d and by scaling and translating t_i's

$$x_i = \left(\frac{t_i - c}{d}\right), \quad c = \frac{t_0 + t_k}{2}, \quad d = \frac{|t_k - t_0|}{2}$$

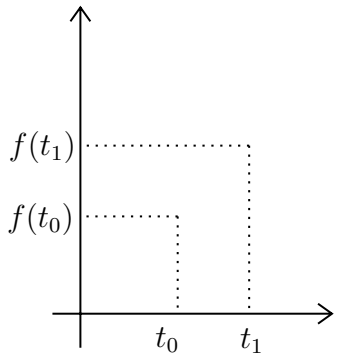
Then $x_i \in [-1, 1]$. Solves overflow issue.

Second issue

GE is expensive ($O(n^3)$). We might want to consider cheaper options if available.

Can we construct interpolating polynomial directly?

Example



$$p_1(t) = \frac{t - t_1}{t_0 - t_1} f(t_0) + \frac{t - t_0}{t_1 - t_0} f(t_1)$$

4.1.2 Lagrange Basis

Give $\{t_i\}$, n-th degree basis conditions

$$L_i(t) = \prod_{k=0, k \neq i}^{n} \frac{t - t_i}{t_k - t_i}$$

Then the interpolating polynomial is

$$p(t) = \sum_{i=0}^{n} L_i(t) f(t_i)$$

Some properties of L_i

1. $L_i(t_i) = 1$
2. $L_i(t_k) = 0$ if $k \neq i$

Theorem The interpolating polynomial is unique

Review

- Interpolation
  - Motivation Problem
  - Actual problem

  - Weierstrass Approximation Theorem
  - Polynomial Basis
    * Vandermonde Matrix, $O(n^3)$
    * Ill-conditioned
  - Lagrange Basis
    * Directly construct, interpolating polynomial
    * $O(n^2)$
- The interpolating polynomial is unique

Proof Let p and q of the same degree k, and p, q interpolate the data $\{(x_i, f_i)\}_{i=0}^k$ ($p(x_i) = f_i$, and $q(x_i) = f_i$)

Consider $r(t) = p(t) - q(t)$, k-th degree polynomial $\overset{FTA}{\Longrightarrow}$ k many roots
Observe $r(x_i) = p(x_i) - q(x_i) = f_i - f_i = 0$
$k+1$ zeros $\implies$ $r(x) = 0$ $\implies$ $p(x) = q(x)$ ■

4.1.3 Updated Problem (First) Barycentric Form of Lagrange Interpolation

Remind $L_i(x) = \prod_{j=0, j\neq i}^{k} \frac{x - x_j}{x_i - x_j}$, $L(x) = \sum_{j=0}^{k} L_j(x) f_j$

$$
\begin{aligned}
L_i(x) &= \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)} \\
&= \frac{x - x_i}{x - x_i} \cdot \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)} \\
&= \frac{\overbrace{\prod(x - x_i)}^{l(x)}}{(x - x_i)\underbrace{\prod_{j\neq i}(x_i - x_i)}_{l'(x)}} \qquad \text{Observe } l'(x) = \sum_{n=0}^{k}\Big(\prod_{j=0, j\neq n}^{k}(x - x_j)\Big) \text{ and } l'(x_i) = \prod_{j=0, j\neq i}^{k}(x_i - x_j) \\
&= \frac{l(x)}{x - x_i} w_i \qquad \to w_i = \frac{1}{l'(x_i)}
\end{aligned}
$$

$$
L(x) = \sum_{j=0}^{k} \frac{l(x) w_i}{x - x_i} f_i = l(x) \sum \frac{w_i}{x - x_i} f_i
$$

Ex $\{(0,1), (1,2), (2,0)\}$

$$
L(x) = (x-0)(x-1)(x-2) \sum_{j=0}^{2} \frac{w_i}{x - x_j} P_j
$$

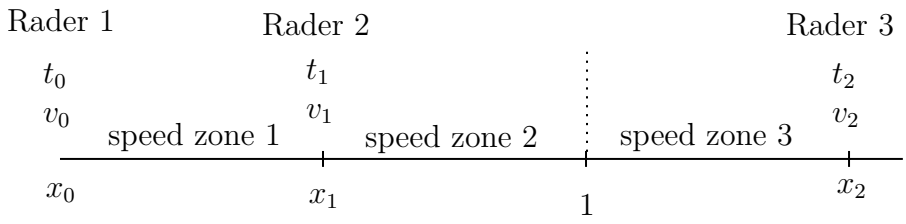
$$w_0 = \frac{1}{l'(x_0)} \frac{1}{(0-1)(0-2)} = \frac{1}{2}$$
$$w_1 = \frac{1}{l'(x_1)} = \frac{1}{(1-0)(1-2)} = -2$$
$$w_2 = \frac{1}{l'(x_2)} = \frac{1}{(2-0)(2-1)} = \frac{1}{2}$$

$$L(x) = x(x-1)(x-2)\left[\frac{1}{2}\frac{1}{x} - 1\frac{2}{x-1} + \frac{1}{2}\frac{0}{x-2}\right] = x(x-1)(x-2)\left[\frac{1}{2x} - \frac{2}{x-1}\right]$$

New data come in: (3, 1).

$$L^n(x) = l^n(x)\sum_{j=0}^{3} \frac{w_j^n}{x - x_j} \quad \rightarrow \text{subscript for new}$$
$$= x(x-1)(x-2)(x-3)\left[\frac{w_0^n}{x}f_0 + \frac{w_1^n}{x-1}f_1 + \frac{w_2^n}{x-2}f_2 + \frac{w_3^n}{x-3}f_3\right]$$

$$w_0^n = \frac{w_0}{0-3} = \frac{1}{6} \qquad w_1^n = \frac{w_1}{1-3} = \frac{1}{2} \qquad w_2^n = \frac{w_2}{2-3} = -\frac{1}{2} \qquad w_3^n = \frac{1}{(3-0)(3-1)(3-2)} = \frac{1}{6}$$



- Update Problem $O(n^2) \implies O(n)$
- Evaluation of the Interpolant $O(n^2) \implies O(n)$

4.1.4 Hermite Interpolation

We are looking for a polynomial p s.t.

$$\left.\begin{array}{ll} p(x_i) = f_i & \rightarrow k+1 \text{ conditions} \\ p'(x_i) = \underbrace{f_i'}_{*} & \rightarrow k+1 \text{ conditions} \end{array}\right\} 2k+2 \text{ conditions} \implies p \text{ is } 2k+1 \text{ degree}$$

$*$: data point, measurement of the derivative

$$P(x) = \sum_{i=0}^{2k+1} \alpha_i x^i$$

$$P(x_j) = f_j \implies \alpha_0 + \alpha_1 x_j + \alpha_2 x_j^2 + \ldots + \alpha_{2k+1} x_j^{2k+1} = f_j$$
$$P'(x_j) = f_j' \implies \alpha_1 + 2\alpha_2 x_j + \ldots + (2k+1)\alpha_{2k+1} x_j^{2k} = f_j'$$

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^{2k+1} \\ \vdots & & & & \\ 1 & x_k & x_k^2 & \dots & x_k^{2k+1} \\ 0 & 1 & 2x_0 & \dots & (2k+1)x_0^{2k} \\ \vdots & & & & \\ 0 & 1 & 2x_k & \dots & (2k+1)x_k^{2k} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \vdots \\ \vdots \\ \alpha_{2k+1} \end{bmatrix} = \begin{bmatrix} f_0 \\ \vdots \\ f_k \\ f_0' \\ \vdots \\ f_k' \end{bmatrix}$$

Question: Can we do sth similar to the case with Lagrange Interpolation?

- Include information from the derivatives
  - Higher degree polynomial
  - Vandermond Matrix
- Can we do sth similar to the case with Lagrange Interpolation?

4.1.5 Lagrange-Like Hermite Interpolation

$$\{x_i, f_i, f_i'\}_{i=0}^N$$
$$H(x_i) = f(x_i)$$
$$H'(x_i) = f'(x_i)$$

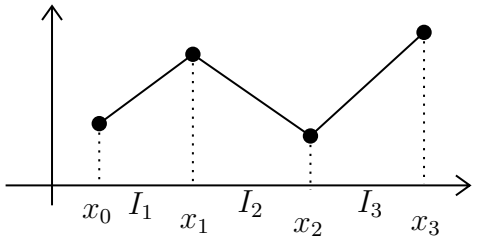
$H \implies 2n+1$-degree

$$H(x) = \sum_i f(x) \underbrace{(\ )}_{1+2(x-x_i)L_i'(x_i)} L_i^2(x) + \sum_i f'(x_i)(x-x_i)L_i^2(x)$$

$$H'(x) = \dots$$

Some questions

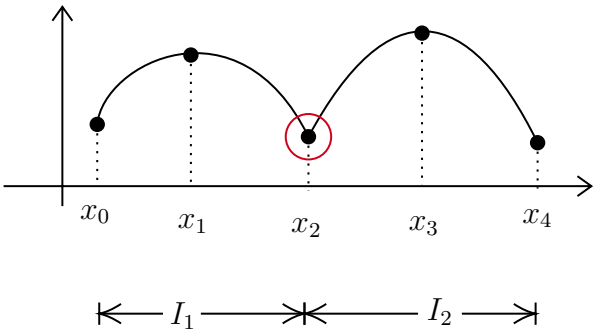
- How well does polynomial interpolation approximation f? (Ass 3)
- Can we approximate every smooth function well using poly interpolation? (Ass 3)
- What are some issues with high degree interpolation?
  - f may not be that smooth
  - too expensive for large n
  - poor extrapolation $\iff$ overfitting



4.2 Piecewise Linear Interpolation

$$y_i = \frac{(x - x_i)}{(x_{i-1} - x_i)} f(x_{i-1}) + \frac{(x - x_{i-1})}{(x_i - x_{i-1})} f(x_i) \qquad i = 1, 2, 3$$

$$y(x) = \begin{cases} y_1(x) & x \in I_1 \\ y_2(x) & x \in I_2 \\ y_3(x) & x \in I_3 \end{cases}$$



4.2.1 Spline Interpolation

Continuity $\Longrightarrow$ Interpolation Condition
First derivatives continuous $\Longrightarrow$ first smoothness condition
$\vdots$

Cubic Spline Interpolation

In each interval I_k

$$y_k(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3$$

$$\begin{array}{ll} y_1(x_0) = f_0 & \\ y_1(x_1) = f_1 & y_1'(x_1) = y_2'(x_1) \\ y_2(x_1) = f_1 & \\ y_2(x_2) = f_2 & y_2'(x_2) = y_3'(x_2) \\ \vdots & \end{array}$$

$$\left.\begin{array}{l} y_1'(x_0) = 0 \\ y_n'(x_n) = 0 \end{array}\right\} \text{clamped cubic spline}$$

$$\left.\begin{array}{l} y_1''(x_0) = 0 \\ y_n''(x_n) = 0 \end{array}\right\} \text{free cubic spline}$$

$$y_1'(x_0) = y_n'(x_n) \quad \text{periodic cubic spline}$$

Vandermonde-Like

$$f_0 = y_1(x_0) = \alpha_0^{(1)} + \alpha_1^{(1)} x_0 + \alpha_2^{(1)} x_0^2 + \alpha_3^{(1)} x_0^3$$
$$f_1 = y_1(x_1) = \alpha_0^{(1)} + \alpha_1^{(1)} x_1 + \alpha_2^{(1)} x_1^2 + \alpha_3^{(1)} x_1^3$$
$$0 = y_1'(x_0) = \alpha_1^{(1)} + 2\alpha_2^{(1)} x_0 + 3\alpha_3^{(1)} x_0^2$$
$$y_1'(x_1) = y_2'(x_1)$$

June 24 review

- Hermite Interpolation: in terms of Lagrange polynomials
- The problems with high order polynomial interpolations
  - f may not be that smooth
  - it is expensive
  - it may not be actually good (ass 3, q3)
- Questions related to polynomial interpolation
  - How well does the interpolation polynomial approximate f? (A3 q2)
  - piecewise polynomial interpolation $O(n)$
  - P. Linear Interpolation
- Spline Interpolation
  - Interpolation condition
  - nth smoothness conditions
    $$y_i^{(n)}(x_i) = y_{i+1}^{(n)}(x_i)$$
    where y_i interpolates f in $[x_{i-1}, x_i]$, y_{i+1} interpolates f in $[x_i, x_{i+1}]$

$\implies$ Cubic Splines. Interpolation condition + first smoothness condition + boundary conditions

4.2.2 Improving Form of The Sparse Matrix (for Cubic splines)

In $I_k = [x_{k-1}, x_k]$. Let

$$S_k(x) = a_k + b_k(x - x_k) + c_k(x - x_k)^2 + d_k(x - x_k)^3$$

be the Interpolating polynomial for f.

Interpolation Condition

$$S_k(x_k) = f_k \implies a_k = f_k$$

$$f_k = S_{k+1}(x_k) = \underbrace{a_{k+1}}_{f_{k+1}} + b_{k+1}(\overbrace{x_k - x_{k+1}}^{:=h_{k+1}}) + c_{k+1}(x_k - x_{k+1})^2 + d_{k+1}(x_k - x_{k+1})^3$$

$$\implies f_k - f_{k+1} = b_{k+1}h_{k+1} + c_{k+1}h_{k+1}^2 + d_{k+1}h_{k+1}^3$$

1st Smoothness Condition $S_k'(x_k) = S_{k+1}'(x_k)$

$$b_{k+1} + 2c_{k+1}(x_k - x_{k+1}) + 3d_k(x_k - x_{k+1})^2 = b_k$$

2nd Smoothness Condition $S_k''(x_k) = S_{k+1}''(x_k)$

$$c_k = c_{k+1} + 2d_{k+1}(x_k - x_{k+1})$$

Exercise pick a boundary condition. write down the resulting matrix.

Correction Cubic splines require interpolation, continuity, 1st smoothness, 2nd smoothness conditions

Why splines? - cheaper to compute - more stable

How to find them? Integration

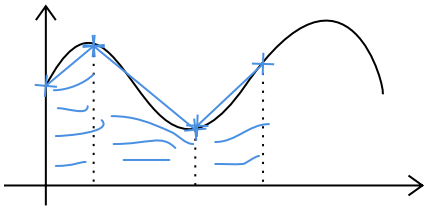
CHAPTER 5

Numerical Integration

Say p is first degree

$$\begin{aligned}\int_a^b f(x)\,dx \approx \int_a^b p(x)\,dx &= \int_a^b \left(\frac{x-a}{b-a}f(b) + \frac{x-b}{a-b}f(a)\right)dx \\ &= \frac{1}{a-b}\left[\int_a^b (a-x)f(b)dx + \int_a^b (x-b)f(a)dx\right] \\ &= \frac{1}{a-b}\left[f(b)\left(ab - \frac{b^2}{2} - a^2 + \frac{a^2}{2}\right) + f(a)\left(\frac{b^2}{2} - b^2 - a^2 + ab\right)\right] \\ &= \ldots = \frac{b-a}{2}[f(b) - f(a)]\end{aligned}$$

Trapezoid Rule



Quadrature $\iff$ Numerical Integration

If a Quadrature rule is desired by interpolating a function, we call that rule an interpolatory rule.

If interpolation is done on equi-distant points the resulting quadrature rule is called a Newton-Cotes rule.

Existence

If $f : \mathbb{R} \to \mathbb{R}$ is bounded[1] on $[a, b]$, then the Riemann integration $I(f)$ exists and unique.

$$\int_a^b (f + g)dx = \int_a^b f \; dx + \int_a^b g \; dx$$

Condition of the Problem $I(f)$

$$\|f - \hat{f}\| = \max_{x \in [a,b]} |f(x) - \hat{f}(x)|$$

exercise: prove that is a norm.

$$\begin{aligned} |I(f) - I(\hat{f})| &= |I(f - \hat{f})| \\ &= \left| \int_a^b (f - \hat{f})dx \right| \\ &\leq \left| \int_a^b \|(f - \hat{f})\|dx \right| \\ &= |b - a|\|f - \hat{f}\| \\ \frac{|I(f) - I(\hat{f})|}{\|f - \hat{f}\|} &\leq |b - a| \end{aligned}$$

5.1 Composite Integration

5.1.1 Trapezoidal Rule

See above

5.1.2 Midpoint Rule

$$\begin{aligned} \int_a^b f(x) &\approx \int_a^b f\left(\frac{a+b}{2}\right) dx \\ &= \underbrace{f\left(\frac{a+b}{2}\right)}_{:=m}(b - a) \end{aligned}$$

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

[1] f is bounded and continuous almost everywhere on $[a, b]$, $m < f(x) < M$, $\forall x \in [a, b]$

observe $a-m=(a-b)/2$, $\;b-m=(b-a)/2$. Integrate both sides

$$I(f)=\underbrace{f(m)(b-a)}_{:=M(f)}+0+\underbrace{\frac{f''(m)}{24}(b-a)^3}_{:=E(f)}+\dots$$

$$f(b)=f(m)+f'(m)(b-m)+\frac{f''(m)}{2}(b-m)^2+\dots$$
$$f(a)=f(m)+f'(m)(a-m)+\frac{f''(m)}{2}(a-m)^2+\dots$$
$$\implies f(a)+f(b)=2f(m)+f''(m)(b-m)^2+\dots \implies f(m)=\frac{f(a)+f(b)}{2}-\frac{f''(m)(b-m)^2}{2}-\dots$$

$$I(f)=\underbrace{(b-a)\left(\frac{f(a)+f(b)}{2}\right)}_{T(f)}\underbrace{-(b-a)\frac{f''(m)(b-m)^2}{2}+\frac{f''(m)}{24}(b-a)^3}_{E(f)}+\dots+\dots$$

Note $b-m=\frac{b-a}{2}$, then

$$E(f)=\left(-\frac{(b-a)^3}{8}+\frac{(b-a)^3}{24}\right)f''(m)=-\frac{(b-a)^3}{12}f''(m)$$

5.1.3 Error

$$E_M(f)=\frac{(b-a)^3f''(m)}{24},\qquad E_T(f)=-\frac{(b-a)^3f''(m)}{12}$$

We are assuming these are the most important source of error.

5.2 Composite Quadrature Rules

Given $[a,b]$ and f integrable on $[a,b]$

$$\int_a^b f(x)dx=\int_a^c f(x)dx+\int_c^b f(x)dx=\int_{x_0=a}^{x_1}f+\int_{x_1}^{x_2}f+\dots+\int_{x_{n-1}}^{x_n=b}f$$

5.2.1 Composite Midpoint Rule

$$\begin{aligned}\int_a^b f(x)dx&\approx\int_{a=x_0}^{x_1}f\left(\frac{x_0+x_1}{2}\right)dx+\int_{x_1}^{x_2}f\left(\frac{x_1+x_2}{2}\right)dx+\dots+\int_{x_{n-1}}^{x_n=b}f\left(\frac{x_{n-1}+x_n}{2}\right)dx\\&=(x_1-x_0)f\left(\frac{x_0+x_1}{2}\right)+\dots+(x_n-x_{n-1})f\left(\frac{x_{n-1}+x_n}{2}\right)\\&=h\sum_{i=0}^{n-1}f\left(\frac{x_i+x_{i+1}}{2}\right)\end{aligned}$$

$O(n)$

5.2.2 Composite Trapezoidal Rule

$$\int_a^b f(x)dx \approx (x_1 - x_0)\frac{f(x_1)+f(x_0)}{2} + \ldots + (x_n - x_{n-1})\frac{f(x_n)+f(x_{n-1})}{2} \qquad \Longrightarrow\ 2n \approx O(n)$$
$$= \frac{h}{2}(f(x_0) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-1}) + f(x_n)) \qquad \Longrightarrow\ n+1 \approx O(n)$$

$$|I(f) - CM(f)| = \left| \sum_{i=0}^{n-1} \underbrace{\left(\int_{x_i}^{x_{i+1}} f(x)dx - (x_i - x_{i-1}) f\left(\frac{x_i + x_{i-1}}{2}\right)\right)}_{\approx E_M(f) \text{ on } [x_{i-1}, x_i]} \right|$$
$$= \left| \sum_{i=0}^{n-1} \frac{(x_i - x_{i-1})^3}{24} \cdot f''\left(\frac{x_i + x_{i-1}}{2}\right) \right|$$
$$= \left| \sum_{i=0}^{n-1} \frac{h^3}{24} f''\left(\frac{x_i + x_{i-1}}{2}\right) \right| \qquad \text{since } n = \frac{b-a}{h}$$
$$\leq n\frac{h^3}{24} \max_i \left| f''\left(\frac{x_i + x_{i-1}}{2}\right) \right|$$
$$= \frac{(b-a)h^2}{24} M$$

Exercise Error analysis for composite Trapezoidal Rule

5.2.3 Simpson's Rule

$$\int_a^b f(x)dx \approx \int_a^b p_2(x)dx$$
$$= \int_a^b \left(L_0(x)f(a) + L_1(x) f\left(\frac{a+b}{2}\right) + L_2(x)f(b) \right) dx$$
$$= \int_a^b L_0(x)f(a)dx + \int_a^b L_1(x) f\left(\frac{a+b}{2}\right) dx + \int_a^b L_2(x)f(b)dx$$
$$= f(a)\overbrace{\int_a^b L_0(x)dx}^{w_0} + f\left(\frac{a+b}{2}\right)\overbrace{\int_a^b L_1(x)dx}^{w_1} + f(b)\overbrace{\int_a^b L_2(x)dx}^{w_2}$$
$$= \sum_{i=0}^{2} w_i f(x_i)$$

Exercise Error analysis for Simpson's rule

Let $Q(f)$ stand for a quadrature rule on f,

$$|Q(f) - Q(\hat{f}) = \left| \sum w_i f(x_i) - \sum w_i \hat{f}(x_i) \right|$$
$$= \left| \sum w_i (f(x_i) - \hat{f}(x_i)) \right|$$
$$\leq \sum |w_i| \cdot \|f - \hat{f}\|$$

As p, degree of polynomial interpolation, goes to infinity $\sum_i^p \|w_i\| \to \infty$

5.2.4 Order of accuracy Numerical Integration

Take $1, x, x^2, \ldots$

$$Q(1) \stackrel{?}{=} I(1)$$
$$Q(x) \stackrel{?}{=} I(x)$$
$$Q(x^2) \stackrel{?}{=} I(x^2)$$
$$\vdots$$

Lowest equality not satisfied gives the order of integration

Review Composite Midpoint rule and Composite Trapezoid Rule both have computational complexity $O(n)$.

5.2.5 Comparison of Rules

	Func Eval	Order of Accuracy
Midpoint	1	2
Trapezoid	2	2
Simpson's	3	4

For Newton-Cotes Rules, n-point rule is order $n+1$ if n is odd and order n if n is even.

Remind as $n \to \infty$, $\sum_{i=0}^n |w_i| \to \infty$ and for $n \geq 11$ for some of w_i's are negative.

So far We picked x_i and determined w_i to get a rule

$$\int_a^b f(x)dx \approx \sum w_i f(x_i)$$

5.3 Gaussian Quadratures

Let $Q(f) = w_0 f(x_0) + w_1 f(x_1)$ be a quadrature rule. Consider

$$2 = \int_{-1}^1 1dx = w_0 + w_1$$
$$0 = \int_{-1}^1 xdx = w_0 x_0 + w_1 x_1$$
$$\frac{2}{3} = \int_{-1}^1 x^2 dx = w_0 x_0^2 + 2w_1 x_1^2$$
$$0 = \int_{-1}^1 x^3 dx = w_0 x_0^3 + w_1 x_1^3$$

$x_0 = -\frac{1}{\sqrt{3}}, \quad x_1 = \frac{1}{\sqrt{3}}$ $w_0 = w_1 = 1$

$$Q(f) = G_2(f) = f(-1/\sqrt{3}) + f(1/\sqrt{3})$$

$\Longrightarrow$ Order of Accuracy

An orthogonal polynomial to the monomial basis $S = \{1, x, x^2, \ldots\}$ on $[-1, 1]$ is a polynomial p such that

$$\langle p(x), x^n \rangle = \int_{-1}^{1} p(x)x^n = 0 \quad \text{for } \forall n$$

Scaled Legendre polynomials

$p_n(1) = 1$ where $p_n(x)$ is Legendre polynomial. Then for $G_n(f)$, x_i is i-th root of $p_n(x)$

$$w_i = \frac{2}{(1 - x_i^2)[p_n'(x_i)]^2}$$

$\Longrightarrow$ Gauss-Legendre Quadrature Example (change of interval)

Use G_2 to approximate $\int_a^b f(x)dx$

$$\begin{aligned} \int_a^b f(x) &= \int_{-1}^{1} f(x(t))\frac{b-a}{2}dt \qquad \text{define } t = \frac{2x-b-a}{b-a},\ dt = \frac{2}{b-a}dx \\ &= \frac{b-a}{2}\int_{-1}^{1} f(x(t))dt = \frac{b-a}{2}G_2(f(x(t))) \end{aligned}$$

Error Formula for Gaussian Legendre Quadrature

$$R_n = \frac{2^{2n+1}(n!)^4}{(2n+1)[(2n!)]^3} f^{(2n)}(\xi) \qquad \text{where } \xi \in [-1, 1]$$

! For GL, w_i are all positive - unlike N-C quadrature where some w_i are negative for 11-point rules and over.

5.4 Integration Problems

We learned how to approximate proper integrals.

5.4.1 Integration from tabular data

x_i	f_i
1	5
3	3.14
10	$2.71^{3.14}$

Composite Trapezoidal Rule
or Cubic spline Interpolation + Integration

5.4.2 Improper Integrals

$$\int_a^\infty f(x)dx = \lim_{b\to\infty}\int_a^b f(x)dx = \int_a^c f(x)dx + \underbrace{\lim_{b\to\infty}\int_c^b f(x)dx}_{\text{if negligible}} \approx \int_a^c f(x)dx$$

$\int_a^b f(x)dx$, but $f(x) \to \infty, x \to 0$ for some $c \in [a,b]$.

ex

$$\int_0^{\pi/2} \frac{\cos x}{\sqrt{x}}dx \overset{x=t^2}{=} \int_0^{\sqrt{\frac{\pi}{2}}} \frac{\cos(t^2)}{\sqrt{t^2}}2t\ dt = 2\int_0^{\sqrt{\frac{\pi}{2}}} \cos(t^2)dt$$

5.4.3 Double Integrals

$$\int_a^b \underbrace{\int_c^d f(x,y)dx}_{g(y)}\, dy = \int_a^b g(y)dy$$

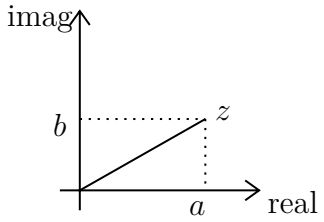
CHAPTER 6

Discrete Fourier Transform

- Complex Numbers
- Fourier Basis
- How to approximate a continuous $f(x)$?
- Any cont function can be written as linear combinations of cosine, sine.
- How to find that linear combination?
- How to find it fast?

6.1 Complex numbers

A complex number $z = a + ib$ where $a, b \in \mathbb{R}$ and $i = \sqrt{-1}$



6.1.1 Properties of Complex Numbers

For $z = a + ib$, we have $\begin{cases} r = |z| = \sqrt{a^2 + b^2} \implies \text{modulus} \\ \theta = \arctan(\frac{b}{a}) \implies \text{phase change} \end{cases}$. Then $z = r(\cos\theta + i\sin\theta)$.

Euler's Formula $e^{i\theta} = \cos\theta + i\sin\theta$.
$z = re^{i\theta}$ Polar Form

Complex Conjugate $\bar{z} = a - ib$
$|z|^2 = z \cdot \bar{z} = a^2 + b^2$

real part: $\mathrm{Re}(z) = a$, imag part: $\mathrm{Im}(z) = b$

6.2 Definitions

Periodic function A function f is called periodic iff $\exists T > 0$

$$f(t) = f(t+T) \qquad \forall t$$

where T is period (frequency: $f = \frac{1}{T}$)

Ex $f(x) = \cos(2\pi x)$. Since $\cos(x + 2\pi) = \cos(x)$

Frequency is not unique. If $\frac{1}{T}$ is a frequency of function f then $\frac{1}{NT}$ is also a frequency of function f for integer N.

Principal period Lowest period of function f.

$$f(t) = \cos(\frac{2\pi k}{T}t) \implies \text{period } \frac{T}{k}$$
$$f(t) = \sin(\frac{2\pi k}{T}t) \implies \text{period } \frac{T}{k}$$

if $k \in \mathbb{Z}$, then $\sin(\frac{2\pi k}{T}t)$ and $\cos(\frac{2\pi k}{T}t)$ are T-periodic

Angular Frequency $\omega = 2\pi f$

Example $\sin(10\pi t)$
$T = \frac{1}{5},\ f = 5,\ \omega = 10\pi$

Any linear combination of $\sin(\frac{2\pi k}{T}t)$ and $\cos(\frac{2\pi k}{T}t)$ - which are T periodic, when $k \in \mathbb{Z}$, are T-period as well. T does not have to be an integer.

$$\left\{\sin\left(\frac{2\pi k}{N}t\right), \cos\left(\frac{2\pi k}{N}t\right)\right\}_{k=0}^{\infty}$$

forms a basis and given any N-period continuous function f can write

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

This is called the Fourier series of f with

$$a_k = \frac{2}{b-a}\int_a^b f(t)\cos\left(\frac{2\pi k}{N}t\right)dt$$
$$b_k = \frac{2}{b-a}\int_a^b f(t)\sin\left(\frac{2\pi k}{N}t\right)dt$$

when $t \in [0, 2\pi]$, and $N = 2\pi$, observe

1. $\int_0^{2\pi} \cos\left(\frac{2\pi k}{N}t\right)\sin\left(\frac{2\pi k}{N}t\right)dt = 0$
2. $\int_0^{2\pi} \cos\left(\frac{2\pi k}{N}t\right)\cos\left(\frac{2\pi j}{N}t\right)dt = 0 \quad$ if $j \neq k$
3. $\int_0^{2\pi} \sin\left(\frac{2\pi k}{N}t\right)\sin\left(\frac{2\pi j}{N}t\right)dt = 0 \quad$ if $j \neq k$
4. $\int_0^{2\pi} \sin(kt)dx = 0$
5. $\int_0^{2\pi} \cos(kt)dx = 0$

Take $\langle f, g\rangle = \int_0^{2\pi} f(x)\cdot g(x)dx$, then $B = \{1, \cos(kt), \sin(kt)\}$ is an orthogonal set.

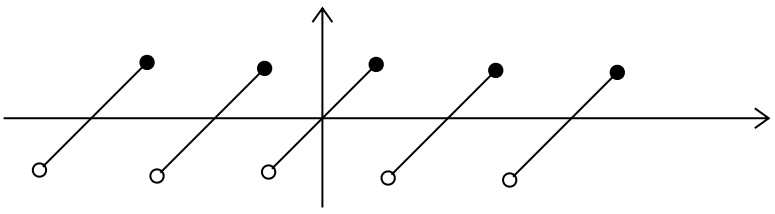
Take f_i, $i = 0, \ldots, k$ from set B.

Assume

$$f_0(x) = \sum_{i=1}^{k} c_i f_i(x)$$
$$\implies f_0(x)f_0(x) = \sum_{i=1}^{k} c_i f_i(x)f_0(x)$$
$$\implies \underbrace{\int_0^{2\pi} f_0^2(x)dx}_{\neq 0,\text{ ex, confirm}} = \underbrace{\sum_{i=1}^{k}\int_0^{2\pi} c_i f_i(x)f_0(x)dx}_{=0\text{ by properties}}$$

Hence by contradiction, B is a linear independent set and since B spans N-period periodic continuous function. B is a basis for N-period continuous function.

Example Find a Fourier series for $f(x) = x$ for $-2 < x \leq 2$.



periodic extension

How to compute coefficients of the Fourier series?

$$f(t) = a_0 \sum_{k=1}^{\infty} a_k \cos(kt) + \sum_{k=1}^{\infty} b_k \sin(kt)$$

$$\|f(t)\|_2 = \sqrt{\langle f, f \rangle}$$

Take $b_i \in B$ and consider

$$\langle f, \beta_i \rangle = \langle 1a_0, \beta_i \rangle + \sum_{k=1}^{\infty} \langle a_k \cos(kt), \beta) i \rangle + \sum_{k=1}^{\infty} \langle b_k \sin(kt), \beta) i \rangle$$

1. $\langle f, \beta_i \rangle = \int_0^{2\pi} a_0 \beta_i(t) dt, \quad \beta_i(t) = 1 \implies a_0 = \dfrac{\langle f, \beta_i \rangle}{\int \beta_i(t) dt}$
2. $\langle f, \beta_i \rangle = \int_0^{2\pi} a_k \cos(kt) \beta_i(t) dt. \quad \beta_i(t) = \cos(kt) \implies a_k = \dfrac{\langle f, \beta_i \rangle}{\int_0^{2\pi} \cos(kt) \beta_i(t) dt}$
3. $\langle f, \beta_i \rangle = \int_0^{2\pi} b_k \sin(kt) \beta_i(t) dt. \quad \beta_i(t) = \sin(kt) \implies b_k = \dfrac{\langle f, \beta_i \rangle}{\int_0^{2\pi} \sin(kt) \beta_i(t) dt}$

B is a basis for N-periodic cont. functions

Going back to the example

$$\frac{a_0}{2} = \frac{\langle f, 1 \rangle}{\int_{-2}^{2} 1 \; dt}$$

$$a_k = \frac{\left\langle f, \cos\left(\frac{2\pi}{4} kt\right) \right\rangle}{\left\langle \cos\left(\frac{2\pi}{4} kt\right), \cos\left(\frac{2\pi}{4} kt\right) \right\rangle}$$

$$b_k = \frac{\left\langle f, \sin\left(\frac{2\pi}{4} kt\right) \right\rangle}{\left\langle \sin\left(\frac{2\pi}{4} kt\right), \sin\left(\frac{2\pi}{4} kt\right) \right\rangle}$$

Note rather than a_0 we usually use $\frac{a_0}{2}$.

Ex Find Fourier series of $f(x) = x$ on $(-2, 2)$

$$\frac{a_0}{2} = \frac{2}{4} \int_{-2}^{2} x \cdot 1 dx = 0$$

$$a_k = \frac{1}{2} \int_{-2}^{2} x \cos\left(\frac{2\pi k}{4}\right) x dx = IBP = 0$$

$$b_k = \frac{1}{2} \int_{-2}^{2} x \sin\left(\frac{2\pi k}{4}\right) x dx = \begin{cases} -\frac{4}{k\pi} & k \text{ even} \\ +\frac{4}{k\pi} & k \text{ odd} \end{cases}$$

Then

$$g(x) = \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} \sin\left(\frac{k\pi x}{2}\right)$$

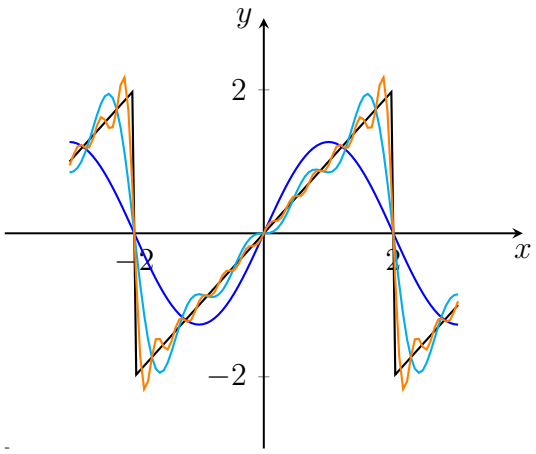


Figure 6.1: Gibbs Phenomenon

Exercise Find a fourier series for

$$f(x) = \begin{cases} -2 & -1 \le x \le 0 \\ 2 & 0 < x \le 1 \end{cases} \qquad \text{on } [-1, 1]$$

Soln $a_k = a_0 = 0$, $b_k = \begin{cases} 0 & \text{if } k \text{ is even} \\ \text{non-zero} & \text{if } k \text{ is odd} \end{cases}$

Proposition If f is an odd function, then $a_k = 0$ for $k = 1, \ldots, \infty$.

If f is an even function, then $b_k = 0$ for $k = 1, \ldots, \infty$.

Q Which function can be approximated by a Fourier series?

A Theorem 4.1 in the course note

Fundamental Convergence Theorem for Fourier Series

Let $V = \{f | \sqrt{\int_{-\pi}^{\pi} f^2(x)dx} < \infty\} = L^2[-\pi, \pi]$, then for all f in $L^2[-\pi, \pi]$, $\exists a_0, a_k$ and b_k, s.t. if we define

$$g_n(x) = \frac{a_0}{2} + \sum_{k=1}^{n} (a_k \cos(kx) + b_k \sin(kx))$$

then $g_n(x) \to f(x)$ as $n \to \infty$ in the sense $\sqrt{\int_{-\pi}^{\pi}(g_n(x)-f(x))^2dx} \to 0$ (we simply denote $\lim_{n\to\infty} g_n(x) =: g(x)$)

$$L^2 = L^2[-\pi, \pi]$$
$$\langle f, g\rangle = \int_{-\pi}^{\pi} f(x)\cdot g(x)dx$$
$$\|f\|_{L^2} = \sqrt{\int_{-\pi}^{\pi} f^2(x)dx}$$

Some facts about L^2

- $L^2 \supseteq \mathbb{C}$
- L^2 contains many of the discontinuous function
- L^2 is a vector space $v_1, v_2 \in L^2$ then $c_1, c_2 \in \mathbb{R}$. And norm on L^2 is $\|\cdot\|_{L^2}$
- distance between two elements f, g is $\|f-g\|_{L^2}$

6.3 Complex Form of Fourier Series

More natural way of writing the Fourier Series of a function f.

Defn The complex Fourier Series of a function f is given by

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{ikt}$$

with $f(t)$ defined on $[-\pi, \pi]$ and period 2π.

With this form, we just need to find one set of coefficients c_k

$$c_k = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)e^{-ikt}dt$$

$$B = \{e^{ikt}\}_{k=-\infty}^{\infty}$$
$$\langle f(t), g(t)\rangle = \int_{-\pi}^{\pi} f(t)g(t)dt \implies c_k = \frac{\left\langle f(t), e^{-ikt}\right\rangle}{\left\langle e^{-ikt}, e^{-ikt}\right\rangle}$$

Prop The relationship between a_k, b_k and c_k

1. $a_{-k} = a_k, b_{-k} = -b_k$
2. $\overline{c_k} = c_{-k}$
3. $a_k = -2\,\mathrm{Re}(c_k), b_k = -2\,\mathrm{Im}(c_k)$

4. $\begin{cases} \text{Im}(c_k) = 0 & \text{if } f \text{ is even} \\ \text{Re}(c_k) = 0 & \text{if } f \text{ is odd} \end{cases}$
5. $b_0 = 0, c_0 = \frac{1}{2}a_0$

Theorem The complex and "real"(trigonometric form) of the Fourier series for a function f are equivalent.

What is the importance of this theorem?

6.4 Discrete Fourier Transform

Rather than a continuous periodic function $f(t)$, we have $\underbrace{\text{data}}_{\substack{\textit{discrete} \\ \textit{time} \\ \textit{signal}}}$ $\underbrace{f[n]}_{\text{period is } n}$.

6.4.1 Background

Roots of unity

for $z \in \mathbb{C}$, the roots of the equation

$$z^n = 1$$

are called n roots of unity.

Denoting k-th root of N distinct roots

$$w_N^k = e^{i\frac{2\pi k}{N}} \qquad 0 \leq k < N$$

Properties

- $(w_N^k)^N = 1$
- $w_N^{-k} = w_N^{N-k}$

Defn The DFT $\mathcal{F}[k]$ of the discrete time signal $f[n]$ is

$$\mathcal{F}[k] = DFT\{f[n]\} = \frac{1}{N}\sum_{n=0}^{N-1} f[n]w_N^{-kn} \qquad \forall k \in \{0, \ldots, n-1\}$$

Defn The IDFT of the discrete frequency signal $\mathcal{F}[k]$ is

$$f[n] = IDFT\{\mathcal{F}[k]\} = \sum_{k=0}^{N-1} \mathcal{F}[k]w_N^{kn} \qquad \forall n \in \{0, \ldots, N-1\}$$

Matrix form

$$Wf = \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ \vdots & & & & & \\ \vdots & & w_N^{-kn} & k \geq 1 & & \\ \vdots & & & & & \\ 1 & & & & & \\ 1 & & & & & \end{bmatrix} \begin{bmatrix} \frac{1}{N} f[0] \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \frac{1}{N} f[N-1] \end{bmatrix} = \begin{bmatrix} \mathcal{F}[0] \\ \mathcal{F}[1] \\ \vdots \\ \vdots \\ \vdots \\ \mathcal{F}[N-1] \end{bmatrix}$$

The matrix is what we have, and it $\implies \begin{bmatrix} 1 & a & a^2 & \dots \\ 1 & b & b^2 & \dots \\ \vdots & & & \end{bmatrix}$. LHS vector is given, and RHS vector is asked.

inverse of W $\quad W^{-1} = N\overline{W}$

$$\implies \text{IDFT:} \quad N\overline{W}\mathcal{F} = f$$

what does periodic in n means?

$$f[n] = f[n+N]$$

$\implies$ F is periodic in k with period N.

Example Find the DFT for $(x_0, x_1, x_2, x_3) = (0, 1, 0, 0)$

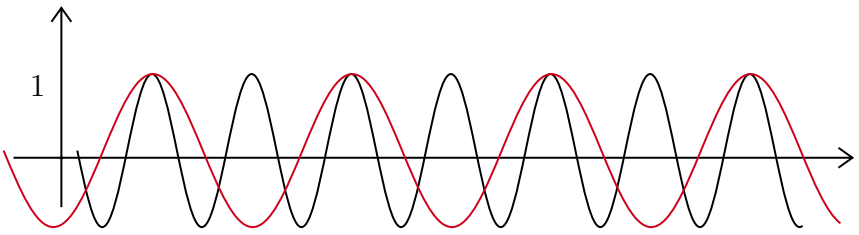
$$X[k] = DFT\{x_n\} = \frac{1}{N} \sum_{n=0}^{N-1} x_n w_N^{-kn} = \frac{1}{4} \sum_{n=0}^{3} x_n w_4^{-kn}$$

$$w_4^{-kn} = (e^{\frac{2\pi i}{4}})^{-kn} = e^{\frac{-2\pi kn}{2}}$$

$$X[k] = \frac{1}{4} w_k^{-k \cdot 1} = \frac{1}{4} e^{\frac{-i\pi k}{2}}$$

$$\implies X[0] = \frac{1}{4}, X[2] = -\frac{1}{4}, X[1] = -\frac{i}{4}, X[3] = \frac{i}{4}$$

$$x_n = \sum_{k=0}^{3} X[k] w_4^{kn}$$



Prop For any real time signal $f[n]$, the frequency $F[k]$ satisfies

1. $\mathrm{Re}(F[-k]) = \mathrm{Re}(F[k])$
2. $\mathrm{Im}(F[-k]) = -\mathrm{Im}(F[-k])$
3. $\overline{F[k]} = F[-k]$
4. If $f[-n] = f[n]$, then $\mathrm{Im}(F[k]) = 0$
5. If $f[-n] = f[n]$ then $\mathrm{Re}(F[k]) = 0$

F is periodic with period $N \implies F[k] = F[k + SN], \quad s \in \mathbb{Z}$.
combining with (3) $\quad \overline{F[k]} = F[N - k]$

Aliasing In DFT, for any given time signal, frequency signal will contain more than the actual underlying frequencies. This is due to some frequencies are "aliases" to other frequencies.

Nyquist Frequency (Sampling Theorem) In order to avoid aliasing errors, you sample twice the frequency of maximum frequency existent in the signal.

The Computational Cost of Fourier Transform

$$\frac{1}{N}\sum_{n=0}^{N-1} x_n w_N^{-kn} = X[k]$$

$$Cost(DFT) = (N \cdot M + (N-1)\cdot A)\cdot N = 2N^2 \cdot A$$

A is for addition

$$\begin{aligned}
X[k] &= \frac{1}{N}\sum_{n=0}^{N-1} x_n w_N^{-kn} \\
&= \frac{1}{N}\left(\sum_{n=0}^{\frac{N}{2}-1} x_n w_N^{-kn} + \sum_{n=\frac{N}{2}}^{N-1} x_n w_N^{-kn}\right) \\
&= \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1}\left(x_n + \underbrace{w_N^{-k\frac{N}{2}}}_{*} x_{n+\frac{N}{2}}\right) w_N^{-kn} \\
&= \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1}\left(x_n + (-1)^k x_{n+\frac{N}{2}}\right) w_N^{-kn}
\end{aligned}$$

* here

$$w_N^{-k\frac{N}{2}} = (e^{\frac{2\pi i}{N}})^{-k\frac{N}{2}} = e^{-\frac{2\pi i k N}{2N}} = (e^{-i\pi})^k = (-1)^k$$

cost: $\frac{N}{2}(1A + 1M) = N$ (M is multiplication)

$\implies$ computational cost of this formulation is N^2 addition. Can we do this recursively?

case 1 k is even

$$\overline{X}_\ell = X_k = \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1}\left(x_n + x_{n+\frac{N}{2}}\right)w_N^{-kn}$$

Since k is even, $k = 2\ell$

$$X_k = \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1}\left(x_n + x_{n+\frac{N}{2}}\right)w_N^{-2\ell n} = \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1}\left(\underbrace{x_n + x_{n+\frac{N}{2}}}_{y_n}\right)w_N^{-\ell n} = \frac{1}{2}DFT\{y_n\}$$

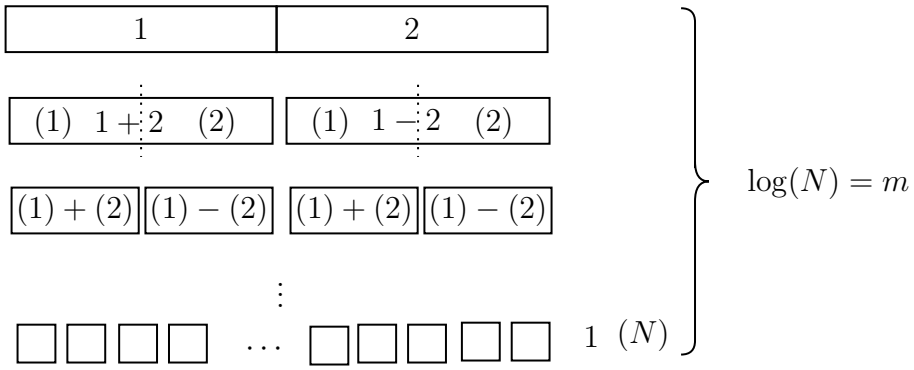
case 2 k is odd, $k = 2\ell + 1$

$$\widetilde{X}_\ell = X_k = \frac{1}{N}\sum_{n=0}^{\frac{N}{2}-1}\left(x_n - x_{n+\frac{N}{2}}\right)w_N^{-\ell n} = \frac{1}{2}DFT\{z_n\}$$

$z_n = (x_n - x_{n+\frac{N}{2}})w_{N/2}^{-\ell n}$

cost $2\frac{N}{2}(\frac{N}{2}) + 2\frac{N}{2}(\frac{N}{2}) = N^2$

Assume $N = 2^m$



Cost of this recursive algorithm is $O(N \log N)$. Divide-and-conquer strategy. Fast Fourier Transform

How do we use FFT on $\{x_n\}_{n=0}^N$ when $N \neq 2^m$?

Option 1 If $N = 2^m r$, where $2 \nmid r$. m levels DDT, last level DFT $O(r^2)$. Total: $O(m2^m r + mr^2)$

Side bar Colley-Tukey Alg (Mixed-Radix Alg) (Option $*$)

Option 2

$$X[k] = \sum_{n=0}^{N} x[n]w_N^{-kn}$$

$N = 2^m r, 2 \nmid r \quad \hat{N} = 2^M, d = 2^M - 2^m r$

$$\hat{X}[n] = \begin{cases} X[n] & \text{if } n \leq 2^m r \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{X}[k] = \sum_{n=0}^{\hat{N}} \hat{X}[n] w_{\hat{N}}^{-kn} = \sum_{n=0}^{N} \hat{X}[n] w_{\hat{N}}^{-kn} = \sum_{n=0}^{N} X[n] w_N^{-kn}$$

this is called zero-padding.

6.5 DFT and bases

Take $\mathbb{R}^n$ with inner product

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i, \quad \forall x, y \in \mathbb{R}^n$$

$B = \{e_i\}_{i=1}^n$ is a basis for $\mathbb{R}^n$ where e_i has 1 in i-th entry.

$$\vec{f} = \sum_{i=1}^{n} f[i] e_i$$

$$f[n] = \sum_{k=1}^{N} F[k] w_N^{kn} = F[n] \vec{E}_{k+1}$$

Define

$$\vec{E}_k = \begin{bmatrix} w_N^{-k\cdot 0} \\ w_N^{-k\cdot 1} \\ \vdots \\ w_N^{-k(N-1)} \end{bmatrix}$$

$$\implies f = \sum F[k] \vec{E}_{k+1}$$

6.6 Polynomial Multiplication

Given $p(x), q(x)$ polynomials of degree n, find $r(x) = p(x)q(x)$

$$r_i = \sum_{\max\{0, i-(n-1)\} \le k \le \max\{i, n-1\}} p_k q_{i-k} \implies O(n^2)$$

$r(x_k) = p(x_k)q(x_k) \implies$ cost of interpolation is $O(n^2)$

What happens if $x_k = w_N^{-k}$? $\implies$ Vandermonde matrix V will equal to DFT matrix $O(n \log n)$