Physics 480/680  Astro 690
COMPUTATIONAL PHYSICS
with NUMERICAL RECIPES
James Sethna, sethna@lassp.cornell.edu
http://www.lassp.cornell.edu/sethna
Clark 528, 255-5132
Grader: Woosong Choi, wc274@cornell.edu
[Not TA]

Numerical Methods

Matrices
Solve $A\hat{x} = \hat{b}$, $A^{-1}$
Eigenvalues

Equations
Differential
[2 point boundary value
  Integral
  Partial Differential]

Functions
Interpolation
Evaluation
Integration
Minimization
Root Finding
Fourier Transforms

Know how 'black boxes' function
- When reliable (subtle?)
- Physics at failure points
- Repairs

[Data analysis]
• Know how 'black boxes' function
  - When reliable
  - Physics at failure points
  - Repairs
• Special routines for special tasks [Research]
• Apply ideas in new contexts [Invention]

Structure [Pass out sign-in sheets]
• Mostly pass-fail (or see me)
• Homework every other week
• Group projects 50% of effort
• Computer lab, class time (12-2) Friday (roughly alternate weeks) Group Projects
  Rockefeller BS (one floor down, NE corner)
  (May bring laptops)
• Languages poll (all are OK):
  - Matlab/Octave
  - C++
  - Fortran
  - Python
  - Mathematica
  - None
Structures under external loads

Elasticity theory: solving for stress and strain of engineering

(Bevel Gear (Integrated Group)

Finite Element Model, Spiral)
(3) Linear physical system: fuse network

\[ \text{Net current out of node } i = \sum_j I_{ij} = 0 \]

Ohm's law \[ I_{ij} = \frac{(V_i - V_j)}{R_{ij}} \]

\[ L \cdot (L-2) \text{ Interior } \sum_j (V_i - V_j) / R_{ij} = 0 \]

\[ L \text{ Top Surface} \quad \sum_j (V_i - V_j) / R_{ij} + V_i / R_{i\text{IN}} = V_{\text{IN}} / R_{i\text{IN}} \]

\[ L \text{ Bottom Surface} \quad \sum_j (V_i - V_j) / R_{ij} + V_i / R_{i\text{OUT}} = V_{\text{OUT}} / R_{i\text{OUT}} \]

\[
\begin{pmatrix}
\sum_j (1 / R_{ij}) & -1 / R_{ij} \\
-1 / R_{ij} & \ddots & \ddots \\
& \ddots & \ddots & -1 / R_{ij} \\
& & \ddots & \ddots & \ddots \\
& & & \ddots & \ddots & \ddots \\
\end{pmatrix}
\begin{pmatrix}
V_0 \\
V_1 \\
\vdots \\
V_{L-2} \\
V_L \\
\end{pmatrix}
= 
\begin{pmatrix}
V_{\text{IN}} / R_{i\text{IN}} \\
0 \\
\vdots \\
0 \\
\end{pmatrix}
\]

[Sherman–Morrison Trick: Find new solution after fuse blows]

\[ A^{-1} = A^{-1} \begin{pmatrix}
-1 / R_{\text{Run}} & 1 / R_{\text{Run}} \\
1 / R_{\text{Run}} & -1 / R_{\text{Run}} \\
\end{pmatrix} \text{ four non-zero entries} \]
II.2 Methods: Dense Matrices

\((AB)^{-1} = B^{-1}A^{-1}\): Factor into invertible parts

<table>
<thead>
<tr>
<th>Method</th>
<th>Picture</th>
<th>Time</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invert</td>
<td>(A^{-1})</td>
<td>(N^3)</td>
<td>Poor Accuracy</td>
</tr>
<tr>
<td>LU</td>
<td>(LU)</td>
<td>(\frac{3}{2}N^3)</td>
<td>Non-Square, Range=Null</td>
</tr>
<tr>
<td>SVD</td>
<td>((LU^T)(W)(LV^T))</td>
<td>((slw)N^3) improving</td>
<td>Symmetric +Pos. Def.</td>
</tr>
<tr>
<td>Cholesky</td>
<td>(LL^T)</td>
<td>(\frac{1}{6}N^3)</td>
<td>Update (A \rightarrow A + uu^T)</td>
</tr>
<tr>
<td>QR</td>
<td>(QR)</td>
<td>(\frac{2}{3}N^3)</td>
<td>Sherman-Morrison</td>
</tr>
</tbody>
</table>

Why possible? How implemented? Errors?

- See Numerical Recipes -

\((\cdot \cdot) = \text{Orthogonal, } Q^TQ = I \) [Like \(NN^T\), Rotation, Easy to Invert]

\((\cdot \cdot) = \text{Diagonal, Easy to Invert, } (w_0^{-1}w_1^{-1}w_n^{-1})\)

\((\cdot \cdot), (\cdot \cdot) = \text{Lower, Upper, Triangular, Easy to Apply Inverse}\)
LU decomposition: Workhorse

\[ A \cdot \vec{x} = L \cdot (U \cdot \vec{x}) = \vec{b} \]

\[ L = \begin{pmatrix} x_{00} & 0 & 0 & \cdots \\ x_{01} & x_{11} & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ \end{pmatrix} \]

First solve for \( y \)

Forward Substitution
\[ x_{00} y_0 = b_0 \Rightarrow y_0 = \frac{b_0}{x_{00}} \]
\[ x_{01} y_0 + x_{11} y_1 = b_1 \]
\[ \Rightarrow y_1 = \frac{b_1 - x_{01} y_0}{x_{11}} \]

Then solve \( U \cdot \vec{x} = y \)

Backward Substitution
(same idea)

\[ \sim N^{3/2} \text{ multiply-adds} \]
\[ \sim N \text{ divisions} \]

\[ \rightarrow \sim N^2 \text{ multiply-adds} \]
(same as applying \( A^{-1} \), more accurate)
II.3 What's tricky about solving $A\vec{x} = \vec{b}$?

Hint: Non-square matrices $M \times N$ $A$
- $M > N$: Too many equations
- $M < N$: Too many unknowns

In general, a linear operator $A$ has a range and a null space.

- If $\vec{b}$ is not in range, $A\vec{x}$ has no solution.
- If $\vec{b}$ is in range, and $\vec{x}$ is a solution, so is $\vec{x} + \vec{n}$ for $\vec{n}$ in null space.

This also happens for square matrices, if $A$ is singular.

How to find the range (to check if solvable)?

How to find the null space (to give general solution)?
Note: $A$ has $M=2$ but $\text{Range}(A)$ has one dimension ($A$ is singular)
Singular Value Decomposition (SVD)

\[ A = U \Sigma V^T \]

- **Other Conventions**:
  - \([\text{Mathematica}]\)
  - \((M \times M)(M \times N)(N \times N)\)

- **Diagonal**

\[ V^T V = V V^T = I \]

\[ M \times N: \quad U^T U = I \]

\[ M < N: \quad U^T U = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{pmatrix} \quad (\text{if } M < N) \]

**Example**: \( A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} \)

\[ = \begin{pmatrix} \sqrt{2} & -\sqrt{2} & 0 \\ \sqrt{2} & \sqrt{2} & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \sqrt{2} & \sqrt{2} & 0 \\ -\sqrt{2} & \sqrt{2} & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} \]

\[ = \begin{pmatrix} u_0 \\ u_1 \\ 0 \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix} \]

- Plot \( u_0, u_1, v_0, v_1, v_2 \) on 'Tricky Plot'
Diagonal elements of $W$, $w_0, w_1, \ldots, w_{N-1}$ are the singular values of $A$. All may be chosen $\geq 0$ (by flipping signs) of $V_n$.

Is there a more friendly meaning to 'singular value'? Consider

$$A^TA = VWU^TUWV^T = VW^2VT$$

**Eigenvalues of $M = Eigenvalues of OMO^{-1}$**

(eigenvectors of $OMO^{-1} = 0$, eigenvectors of $M$)

**Eigenvalues of $A^TA = Singular values of A$ squared $w_n^2$**

IF $A$ symmetric, singular values $= |eigenvalues|$.
Columns of $U$ and $V$ are orthonormal:

$$U^T U = \sum_{x=0}^{N-1} (U^T)_{xj} U_{xj} = \sum_{x=0}^{N-1} U_{xj} U_{xj} = U_{xj} U_{xj} = I_{ij} = 5_{ij}$$

Range + Null space

$$(A \cdot \bar{x})_i = \sum_{n,a,j} U_{in} W_{n\alpha} (V_j^T)^{\alpha} \bar{x}_j$$

$$= \sum_{n,j} U_{in} W_{n\alpha} V_j \alpha \bar{x}_j$$

$$A \cdot \bar{x} = \sum_n \hat{U}^{(n)} W_n \hat{V}^{(n)} \bar{x}$$

$$= \sum_n \left( W_n \hat{V}^{(n)} \bar{x} \right) \hat{U}^{(n)}$$

Let $w_0, \ldots, w_{p-1} > 0$, $w_{p+1} \ldots w_N = 0$

$\Rightarrow$ Range $(A)$ has orthonormal basis $\hat{U}_0 \ldots \hat{U}_{p-1}$.

$\Rightarrow$ Null space of $A$ are vectors with $\hat{V}^{(n)} \cdot \bar{x} = 0$ for $0 \leq n < P$

$\Rightarrow \bar{x} = \sum_{n=P}^N c_n \hat{V}^n$ (build $\bar{x}$ from rest of basis in null space)

$\hat{V}_P, \ldots, \hat{V}_{N-1}$ is orthonormal basis for Null $(A)$. 

$A \cdot \bar{x} = 6$
How do you use SVD to solve $A\hat{x} = \tilde{b}$?

$A^{-1} = V \cdot W^{-1} \cdot U^T$

$A^{-1}A = (V W^{-1} U^T) (W W V^T)$

$= V W^{-1} \cdot W V = V V^T = I$

Problems: $\tilde{W}^{-1}$ has $\infty$'s if $P \neq N$, $\text{some } w_n \neq 0$

- What if $\tilde{b}$ is in range, but some $W_n$ are zero?
- What if $\tilde{b}$ is in range, but zero $W_n$ have errors?
  $[4 \times 10^{-16} \neq 0! \text{ When is it rounding errors?}]$
- What if $\tilde{b}$ is slightly out of range (errors)?
- What if $\tilde{b}$ is just not in range?

Define $\tilde{W}^{-1} = \begin{pmatrix} \tilde{w}_0^{-1} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & \cdots \end{pmatrix}$

SVD Solution

$\hat{x}_0 = (V \tilde{W}^{-1} U^T b) = \sum_{n=0}^{p-1} \tilde{w}_n ^{(n)} U^{(n)} b$

General solution

$\hat{X} = \hat{x}_0 + \sum_{n=p}^{N-1} C_n \tilde{w}_n$

$\|\hat{X}\| = \hat{x}_0 + \sum_{n=p}^{N-1} C_n \geq \hat{x}_0$

SVD Solution is shortest.
What if $\tilde{b}$ not in range?
(Small errors, or just outside)

SVD solution:

$A \cdot x_0 = A \cdot (V \tilde{W} U^T \cdot \tilde{b})$

= $U \cdot \sum_{n=0}^{p} \tilde{v}_n \cdot U^T \cdot \tilde{b}$

Perpendicular projection of $\tilde{b}$ onto range

$= \sum_{n=0}^{p} \tilde{v}_n \cdot U^T \cdot \tilde{b}$

basis for range

coefficients of $\tilde{b}$ in basis

(minimizes residual $\| A \tilde{x}_0 - \tilde{b} \|$.)

SVD and Principal Component Analysis

Data matrix

$N \times M$ matrix, $N$ experiments, $M$ measurements

$A_{ij}$

RNA microarray 'gene chip'

What features of exp's caused most of the variation in the data?

Biggest variation

Data Space

$A_m = a_{m0} \cdot v_0 + a_{m1} \cdot v_1$

Large singular values

= largest effect on data

$\hat{u} = \text{effect} \quad \hat{v} = \text{origin}$

Also, small singular values = Well constrained relationships

Orly Alter: $\hat{v} = \text{which machine of two used to take data?}$
Problem Demands

<table>
<thead>
<tr>
<th>DOF</th>
<th>1,400,000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>920,000</td>
</tr>
<tr>
<td></td>
<td>327,000</td>
</tr>
<tr>
<td></td>
<td>214,000</td>
</tr>
</tbody>
</table>

Initial Flaw Size and Location
II.5 Sparse Matrices & Iterative Improvement

Different methods are needed for large linear systems:

[Show FEM figure: 0.92 - 1.4M eqns]

1000 x 1000 fuse network \( \Rightarrow \) \( \mathbb{L}^2 = N = 10^6 \) eqns

A is a million x million matrix of doubles:

How much RAM to store it? 8 bytes/double

\[ 8 \text{ bytes/double} \rightarrow \frac{8 \text{ bytes}}{\text{byte}} = 8 \text{ terabytes} \]

Impossible.

But each equation has only 5 non-zero terms \( \Rightarrow 8 \times 8 \times 10^6 \) 5x10^6 non-zero entries. Store as 
\( (i, j, A_{ij}) = (4+4+8) \times 10^6 \) needs 80 Mbyte - quite feasible.

Sparse matrix storage: more data for efficient access

NR discusses compressed column storage

Sparse matrix algorithms?

- A sparse, but \( A^{-1} \) dense

LU, SVD, QR, ... Also dense.

Impossible to use any of these methods for large linear systems.
Preferred methods
* Conjugate Gradient
  • Mainstream, discussed in NR
  • Also used for minimizing nonlinear functions

* Iterative improvement
  • Useful for dense matrices too
  • Introduces condition number
II.6 Iterative Improvement

Suppose I have an approximate way of solving $Ax = b$. Can I use the error to improve it?

When is this useful?

- Bad inverses for huge matrices
- Bad accuracy for near singular matrices (homework)
- Sparse matrix: need sparse approximation to inverse (we'll use $A^{-1} \approx eA^T$)

If solution algorithm is linear, let $B \approx A^{-1}$ be the 'effective' linear operator.

Define $\tilde{x}_0 = B \cdot b$ [Want est. for $\tilde{x} - \tilde{x}_0$]

\[
\tilde{\delta}b = A(\tilde{x} - \tilde{x}_0) \quad [\text{Error in result}]
\]

\[
= b - A\tilde{x}_0
\]

$\Rightarrow \tilde{x} - \tilde{x}_0 = A^{-1} \tilde{\delta}b \approx B \cdot \tilde{\delta}b = B \cdot (b - A\tilde{x}_0)$

$\Rightarrow$ Next approximation

$\tilde{x}_i = \tilde{x}_0 + B \cdot (b - A\tilde{x}_0)$
Repeat process
\[ \tilde{x}_{n+1} = \tilde{x}_n - A \tilde{x}_n \]

iterate to fixed point \( \tilde{x}^* \) \((b - Ax = 0 \text{ fixed})\), to \( \tilde{x}^* \)

Will this converge? How fast?
\[ \tilde{x}_{n+1} - \tilde{x} = \tilde{x}_n - \tilde{x} + B \cdot (\tilde{x} - A \tilde{x}_n) \]
\[ = \tilde{x}_n - \tilde{x} + B \cdot A (\tilde{x} - \tilde{x}_n) \]
\[ = (I - B \cdot A) (\tilde{x}_n - \tilde{x}) \]
\[ = R (\tilde{x}_n - \tilde{x}) \]

- If \( R \) is symmetric, this converges if all of its eigenvalues lie between -1 and 1, \( \lambda_i \) \( \in \mathbb{C} \), \( |\lambda_i| < 1 \)

- Generally, it depends on the
  
  For general \( R \), depends on the norm of the norm
  
  \[ \| R \| = \max_{v \neq 0} \frac{\| R \cdot v \|}{\| v \|} \leq 1 \]

  then the error decreases in length by at least \( \| R \| \)

  \[ \| v \| \text{ is Euclidean L}_2 \text{ norm, this is biggest singular value of } R \]

  \( R \cdot \hat{v}^{(0)} = \hat{R} \cdot \hat{v}^{(0)} \) gives \( \max \frac{\| R \cdot v \|}{\| v \|} = \| R \| \leq 1 \).
If it converges, the rate of convergence is no worse than $\frac{1}{2^k}$. The error after $k$ iterations is bounded by $W_0 = (\max |d_i|)^k$ times the original error.
Note: Iterative improvement useful for sparse systems, if $B$ is sparse too; only applies $A + B$.

Can we find an approximate $B$ that is sparse, and that has $\|R\| = 1 - BA \ll 1$?

$A^T$ is as sparse as $A$ is.

Try $B = \varepsilon A^T$.

$R = I - \varepsilon A^T A$ has singular values (eigenvalues)

\[
|1 - \varepsilon a_0^2|, |1 - \varepsilon a_1^2|, \ldots, |1 - \varepsilon a_{N-1}^2|
\]

Make $\varepsilon$ big enough so $1 - \varepsilon a_0^2 = -\left(1 - \varepsilon a_{N-1}^2\right)$

\[
\varepsilon = \frac{\varepsilon (a_{N-1}^2 + q_0^2)}{1 - \varepsilon a_{N-1}^2}
\]

$\|R\| = r_0 = 1 - \varepsilon a_0^2 = 1 - \frac{2a_{N-1}^2}{a_{N-1}^2 + q_0^2} = 1 - \frac{2}{\varepsilon^2 + 1}$

where $C = \frac{a_{N-1}}{a_{N-1}^2} = \text{Condition number for matrix}$
Condition number = Ratio of largest/smallest singular value (eigenvalue if symmetric)

- Iterative improvement with $B = AT$ converges slowly (by factor $\frac{1}{1 + \epsilon}$) as $\epsilon$ for large $C$

- Truth for many algorithms: matrices with large $C$ hard to solve (nearly singular)

- Professionals use preconditioning; solve $(P \cdot A) \tilde{x} = P \cdot b$
  for suitable $P$ chosen to reduce condition number.

- Big matrices, preconditioners, sparse systems - use package library like PETSc, PETSc.
III. Interpolation and Extrapolation

Given $f(t)$ at $t_0, \ldots, t_{N-1}$, estimate $f(t')$.

Interpolation:

$t_0 < t' < t_{N-1}$

Fancy methods work well

Extrapolation: $t < t_0$, $t > t_{N-1}$

- Usually useful only for small ranges (weather prediction, ...)

How can this be? Many functions share the same points!

What justifies our prejudice that interpolation is smooth?
III.1 Analytic Functions

Two branches of advanced calculus:

Real Analysis

- Focuses on measurable functions
- Many pathological (sick) cases
  \[ f(t) = \begin{cases} 1 & \text{if } t \text{ rational} \\ 0 & \text{if } t \text{ irrational} \end{cases} \]
- \( f(t_n) \) puts no limitations on \( f(t) \)

Methods for
- Interpolation, integration, minimization, root finding, solving differential equations
  all impossible to do elegantly.

But generally,
all fail: fancy methods fail worse.

- These functions arise often in physics
  [Transparency]
- One rarely wishes to interpolate, integrate, or minimize them.
  (Monte Carlo methods)
(Limiting sequences, real analysis)
Martenstic Microstructure

(Avalanches, fractals, and cracking noise)

Devil's Staircase
(circle map)

... hard to interpolate, integrate...

Non-analytic functions are
**Complex Analysis**

- Focuses on **Analytic functions**
  - Infinitely differentiable, integrable
  - Many weird properties, confusing
  - My first B+ in a math course

**Physicist’s Definition:**
A function $f(t)$ is analytic at $t_0$ if it has a convergent power series expansion

$$f(t) = \sum_{n=0}^{\infty} c_n (t-t_0)^n$$

where $r$ is the radius of convergence.

- **Radius of Convergence $R$**
  - $c_n = \frac{1}{n!} \frac{d^n f}{dt^n} \bigg|_{t_0}$
  - Taylor’s theorem

- **Radius of convergence $R$**
  - Distance to nearest singularity in complex plane
  - Ratio test
    $$R = \lim_{n \to \infty} \left| \frac{c_n}{c_{n+1}} \right|$$
    (Better! $c_n \sim \frac{c_n}{R^n}, \text{ some } F$)
Note: \( c_n = \frac{1}{n!} f^{(n)} \Rightarrow \frac{1}{n!} f'' \sim R^{-n} \) (use later)

Generally

\( F = \text{scale of } f(t) \approx f(t_0) \)

\( R = \text{scale of variation of } f(t) \)

(Since \( e \) radius of convergence is \( \infty \) but \( R = 1 \))

Weird facts explained

- Derivative \( \frac{df}{dt} = \sum_{n=0}^{\infty} n c_n (t-t_0)^{n-1} \) analytic.

Must show convergence.

Radius of convergence?

\[ R_{\text{def}} = \lim_{n \to \infty} \sqrt[n]{\frac{1}{n c_n}} = R_f \text{ (Same as } f(t)\text{)} \]

because \( \sqrt[n]{n} \to 1 \), \( (e^{\log n/n} \to e^0 = 1) \)

\[ \Rightarrow \text{Infinitely differentiable,} \]

- Cauchy's theorem [Weird rigidity]

Integrate analytic function around contour \( C \) on complex plane

\( \int_C f(z-\zeta_0) \, dz = 0 \quad \ast \)

\[ \int_a^b f(t) \, dt = \sum_{n=0}^{\infty} \frac{a^n}{n!} (z-\zeta_0)^n \bigg|_a^b = 0 \text{ same radius of convergence \( = 0 \) closed contour.} \]
Note: Allows one to deform contours
\[ \oint_a^b f(t) \, dt + \oint_c^d f(z) \, dz = 0 \]

- Cauchy's Integral Formula
\[ f(t) = \frac{1}{2\pi i} \oint_{c-t} \frac{f(z)}{z-t} \, dz \]
counter-clockwise contour

---

An Amazing Preponderance of Functions in Physics are Analytic

- Normal functions \( x^n, e^x, \sin, \cos, \arccsc, \ldots \)
- Products, sums, solutions of analytic equations, differential equations with analytic coefficients
- Linear response of systems to external waves \( \chi(w) \) [from causality, Kramers-Kröning]
- Any properties inside phases (basis of perturbation theory)
III.2 Examples of Interpolation

Linear Interpolation

\[ f(t) = \frac{t-t_2}{t_1-t_2} f(t_1) + \frac{t-t_1}{t_2-t_1} f(t_2) \]

Two points \(\leftrightarrow\) two coefficients
Discontinuous \(f'\)

Cubic Interpolation

\[ f(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 \]

Four points \(\leftrightarrow\) four coefficients
Discontinuous \(f'\)

Cubic Spline

\(f''(i)\) Next interval
Previous \(f'(i)\)
\(f(t)\)

4 \(c_n\) \(\leftrightarrow\) 2 points + 2 deriva
Continuous \(f', f''\)
Discontinuous \(f''\)

Barycentric Rational

\[ f(t) = \frac{\sum c_n t^n}{\sum d_n t^n} \]

2N coefficients \(\leftrightarrow\) N points
\(c_n, d_n\) - Allows to be smooth
All derivatives continuous
Poles in \(C\) - may be true
Expensive to evaluate, large \(N\)

Myers: Cubic spline for fast sine.

Gutenkunst: Spline for ODE sol'n, needed as input for backward adjoint method of sensitivity analysis...

Warning: Nonanalyticities in interpolation can confuse further numerical methods!
### 3. Analyticity and Convergence

Why does interpolation work for analytic functions?

- **Analytic Function**
  \[ f(x) = \sum_{n=0}^{\infty} C_n (x-x_0)^n \]
  \[ \text{Radius of convergence } R = \lim_{n \to \infty} \frac{C_{n+1}}{C_n} \]

  - "" = \[ \left| x - x_0 \right| \] for alternating series, \( c_{2n} = 0, \ldots \)
  - \( C_n = \frac{1}{n!} f^{(n)}(x_0) = \left. \frac{d^n f}{dx_n} \right|_{x_0} \)
  - \( R \leq \left| x - x_0 \right| \leq \left| x - x_0 \right| f/R^n \)

- **Linear Interpolation**
  - Interval size \( h \)
  - Exact for linear function
  - \( \text{error} \propto \left( \frac{h}{2} \right)^2 \)
  - \( \text{fractional error} \propto \left( \frac{1}{2} \right)^2 \left( \frac{h}{2} \right)^2 \approx \left( \frac{h}{2R} \right)^2 \)

When \( h \ll R \), linear interpolation is good.

Radius of convergence \( \propto \) Scale on which function varies
Cubic Interpolation

- Exact for cubic polynomial

\[ \text{Error} \approx \frac{8}{4!} f^{(4)}(\xi) \left( \frac{h}{2} \right)^4 \times \left( \frac{h}{2R} \right)^4 \]

of analytic functions

As \( h \to 0 \), interpolations converge to the correct answer roughly as a power law in \( \frac{h}{R} \).
IV. Integration of Functions (Quadrature)

Area under the curve
\[ \approx h \left( \frac{1}{2} f(t_0) + \frac{1}{2} f(t_{\text{toth}}) \right) \]

Trapezoidal rule

Error in one segment
\[ \approx \int_{t_0}^{t_{\text{toth}}} \frac{1}{2} f''(t) (t_{\text{toth}} - t_0)^2 \, dt = \frac{1}{6} f''(t_{\text{toth}})^3 \]

\[ = \frac{1}{6} f'' h^3 \text{ compared with } f \cdot h \]

Fractional relative error
\[ \approx \frac{1}{3} \left( \frac{f''}{f} \right) \left( \frac{h}{R} \right)^3 = \frac{1}{3} \left( \frac{h}{R} \right)^3 \]

Error in \( N = \left( \frac{b-a}{h} \right) \) segments
\[ \leq \frac{N}{6} f'' h^3 \leq \frac{b-a}{6} f'' h^2 = \frac{(b-a) f' \left( \frac{1}{2} \right) (R)}{6} \left( \frac{h}{R} \right)^2 \]

Relative error \( \leq \left( \frac{h}{R} \right)^2 \)

Euler-Maclaurin formula: Error
\[ = -\frac{1}{6} \frac{h^2}{R^2} (f_0' - f_0') + \frac{1}{30} \frac{h^4}{4!} (f_0''' - f_0'') + \cdots - \frac{1}{42} \cdots \]
Cubic approximation to \( f \rightarrow \) Simpson's Rule

\[
\int_a^b f(t) \, dt = h \left[ \frac{1}{3} f(a) + \frac{4}{3} f(a+h) + \frac{2}{3} f(a+2h) + \ldots + \frac{2}{3} f(b-2h) + \frac{4}{3} f(b-h) + \frac{1}{3} f(b) \right]
\]

Error in one segment \( \sim \frac{1}{5!} f^{(5)}(\xi) \)

\( \rightarrow \) Relative error \( \sim \left( \frac{h}{R} \right)^6 \)

As \( h \rightarrow 0 \), integrals of analytic functions converge to the correct answer roughly as a power law in \( \frac{h}{R} \).

Basic workhorse of integration...
IV.1. Exponential convergence of integrals

One can do much better than power law convergence.

Notice! errors in convex regions nearly cancel concave region errors.

For integrals \( \int_{-\infty}^{\infty} f(t) \, dt \), functions \( f \to 0 \) at \( \pm \infty \) spaced, trapezoidal rule (equally weighted points) converges exponentially. Error \( \leq Ce^{-AR/t} \).

- Much faster than any power law
- Near-perfect cancellation of errors in neighboring regions.

Why? (1) Quick answer: (2) Good answer:

1. Euler-Macaulay formula

\[
\int_{a}^{b} f(t) \, dt - \text{Trap} = \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!} h^{2k} \left[ f^{(2k)}(b) - f^{(2k)}(a) \right]
\]

Power-law errors Zero if \( f \to 0 \) as \( h \to 0 \), \( \varepsilon \to \frac{\varepsilon^2}{6} + \text{exponential convergence} \)

Not very satisfying...
(2) Good answer (demands complex analysis + Fourier analysis + physics assumptions)

- View trapezoidal rule as the integral of
\[ f(t) \] times \( \Delta(t) = \sum_{n=-\infty}^{\infty} s(t-nh) \]

\[ \text{Trapezoidal Rule} = h \sum_{n=-\infty}^{\infty} f(nh) = \int_{-\infty}^{\infty} f(t) \sum_{n=-\infty}^{\infty} S(t-nh) \, dt \]

- Write \( \Delta(t) \) as Fourier series

\[ \Delta(t) = \frac{1}{h} \sum_{n=-\infty}^{\infty} e^{i\omega_0 t} \]

\[ \omega_0 = \frac{2\pi}{h} \]

\[ \text{Trapezoidal rule} = \int_{-\infty}^{\infty} f(t) \sum_{n=-\infty}^{\infty} e^{i\omega_0 t} \, dt \]

\[ = \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} f(t) e^{i\omega_0 t} \, dt \]

\[ = \sum_{n=-\infty}^{\infty} \hat{f}(n\omega_0) \]

\[ = \int_{-\infty}^{\infty} f(t) \, dt + \sum_{n=1}^{\infty} [\hat{f}(n\omega_0) + \hat{f}(-n\omega_0)] \]

\[ \text{Error} \]
Fourier sampling theorem, special case

If \( \hat{f}(\omega) = 0 \) for \( |\omega| > \omega_0 = \frac{2\pi}{h} \),

integral (and entire function) defined by values at \( f(nh) \).

\[ \text{Re}\hat{f}(\omega) \quad \text{Im}\hat{f}(\omega) \]

\[ \text{How fast does it decay?} \]

**Physicist's theorem**: The Fourier transform \( \hat{f}(\omega) \) of an analytic function [caustics here] decays exponentially as \( \omega \to \pm \infty \).

\[ 15(\omega) \leq e^{-\frac{2\pi R}{h}} \]

Note: This implies trapezoidal rule converges exponentially.

\[ \sum_{n=1}^{\infty} (\hat{f}(n\omega_0) + \hat{f}(-n\omega_0)) e^{-\frac{2\pi R}{h}} = e^{-\frac{2\pi R}{h}} \]

How do we know this?

1. Quick argument
   - Fourier transform of \( \delta(x) = 1 \)
   - Fourier transform of step = \( \frac{1}{\omega} \)
   - \( |X| = \frac{1}{\omega} e^{\infty} \)
   - \( \frac{dX}{dt} = \omega \hat{f}(\omega) \to N \text{ derivatives, } \hat{f}(\omega) \sim \frac{1}{\omega} \)
   - Analytic, \( \hat{f}(\omega) \) dies faster than any power
   - \( \hat{f}(\omega) \) does exponentially
(2) Complex analysis argument.

\[ \hat{f}(\omega) = \int_{-\infty}^{\infty} f(t) e^{i\omega t} \, dt \]

\[ \hat{f}(\omega) = \int_{-\infty}^{\infty} f(t+ir) e^{i\omega(t+ir)} \, dt \]

\[ = e^{-\omega r} \int_{-\infty}^{\infty} f(t+ir) e^{i\omega t} \, dt \]

Deform contour 

Assume no singularities for range \( r \) above real axis

Exponential decay

Fourier Transform of different function
Equal weights, equally spaced points
\rightarrow \text{exponential convergence of}
\text{analytic & no endpoints } \int_{-\infty}^{\infty} f(t) \, dt.

\text{A) Why does Simpson's rule weigh odd points twice as much?}

\text{B) What about finite intervals } \int_{a}^{b} f(t) \, dt ?

\text{C) What about functions with singularities?}

\text{A) Why does Simpson's rule weigh odd points more heavily, when equal spacing is best (on the infinite interval)?}

The error in integrating an analytic function is dominated by the end points.

\begin{array}{c|c}
\text{Internal Error} & \text{End-point Error} \\
\hline
\text{Trap. Rule} & e^{-B/h}, \text{some } B \\
\text{Simpson's} & e^{-2B/h}, \text{some } B \\
\text{Double spacing between points} & (h/R)^2, (h/R)^4, \text{even for giant ranges}
\end{array}
B) What about finite intervals, \( \int_a^b f(t) \, dt \)? Can we get better than power-law convergence?

Yes. Trick: As \( h \to 0 \), increase the order \( M \) of the approximation (using the extra points), so that the error \( (h/R)^M \) dies faster. Two examples...

**B1) Romberg Integration:**

**Trapezoidal Rule**

Extrapolate Trapezoidal Rule to \( h \to 0 \)!

It turns out that halving each doubling of \( h \) increases order of extrapolated approximation by 2.

\[
h = h_0 2^{-(m-2)/2} \quad \rightarrow \quad M \sim -2 \log_2 \left( \frac{2h}{h_0} \right) = A \log h + \text{const}
\]

Error \( (h/R)^M \sim h^{A \log h} \)

Since \( \log h \to -\infty \) as \( h \to 0 \), error decays faster than any power-law, (but not exponentially...)

\[
\varepsilon = (h/R)^M = h_0 2^{2 \log_2 \left( \frac{2h}{h_0} \right)}
\]
**B2) Gauss Points.**

Pick \( M \) points \( x_j \), weights \( w_j \), so that not equally spaced and not equal

\[
\int f(t)\,dt \approx \sum_{j=0}^{M} w_j f(x_j).
\]

is exact for \( f(t) = t^m \), \( m = 0, \ldots, 2M-1 \)

<table>
<thead>
<tr>
<th>( M )</th>
<th>( w_0 )</th>
<th>( w_1 )</th>
<th>( x_0 )</th>
<th>( x_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>(-\sqrt{3})</td>
<td>(\sqrt{3})</td>
</tr>
<tr>
<td>3</td>
<td>(\frac{5}{3})</td>
<td>(\frac{8}{9})</td>
<td>(-\sqrt{15} )</td>
<td>(0)</td>
</tr>
</tbody>
</table>

Not equally spaced!

\( x_0 = \pm \frac{1}{\sqrt{3}} = \pm \sqrt{\frac{3}{3}} \) would be equal.

\( x_0 = \pm \sqrt{\frac{3}{3}} = \pm 1 \) vs \( x_0 = \pm \sqrt{\frac{3}{3}} \)

Order of the method \( M \rightarrow M \)

\[\text{Error} \propto (h/R)^M \]

\( h \propto 1/M \), \( \text{e}^{-2M \log M} \)

Converges exponentially fast.

[Show transparency]
\[
\left( \pm \sqrt{\frac{3}{5}} h, \pm \sqrt{\frac{3}{5}} h \right) \quad 25/324 \\
\left( 0, \pm \sqrt{\frac{3}{5}} h \right) \quad 10/81 \\
\left( \pm \sqrt{\frac{3}{5}} h, 0 \right) \quad 10/81 \\
\]

Equilateral Triangle T
Radius of Circumscribed Circle = \( h \)

25.4.63

\[
\frac{1}{3} \sqrt{3} h^2 \int \int f(x,y) \, dx \, dy = \sum_{i=1}^{n} w_i f(x_i, y_i) + R 
\]

\[
(x_i, y_i) \quad w_i \\
(0,0) \quad 3/4 
R = O(h^4) \\
(h,0) \quad 1/12 
R = O(h^4) \\
\left( -\frac{h}{2}, \pm \frac{h}{2} \sqrt{3} \right) \quad 1/12 
\]

Regular Hexagon H
Radius of Circumscribed Circle = \( h \)

25.4.64

\[
\frac{1}{3} \sqrt{3} h^2 \int_{H} f(x,y) \, dx \, dy = \sum_{i=1}^{n} w_i f(x_i, y_i) + R 
\]

\[
(x_i, y_i) \quad w_i \\
(0,0) \quad 27/60 \\
(h,0) \quad 3/60 \\
\left( -\frac{h}{2}, \pm \frac{h}{2} \sqrt{3} \right) \quad 3/60 
R = O(h^4) \\
\left( -\frac{h}{2}, 0 \right) \quad 8/60 \\
\left( \frac{h}{2}, \pm \frac{h}{2} \sqrt{3} \right) \quad 8/60 \\
\]

\[
\left( \frac{h}{2}, \pm \frac{h}{2} \sqrt{3} \right) \quad 5/72 
R = O(h^4) \\
\left( h, 0 \right) \quad 5/72 
\]
What about functions with singularities?
(Or functions with $-\infty$ and/or $\infty$ as limits?)

Various cases.

(1) Known singularities at end points
    (and endpoints at $\pm \infty$)
    $\Rightarrow$ Change variables
    $\Rightarrow$ Variations on Gauss points

(2) Singularities at known internal points
    $\Rightarrow$ Break up integral at singular points

(3) Other
    $\Rightarrow$ Adaptive quadrature

[See text]
II. Evaluation of Functions

Grab-bag chapter: lots of useful stuff. Discuss a few.

* Polynomials
* Rational Functions
* Continued Fractions (My post-doc)

II.1 Infinite Series: Accelerating Convergence

\[ S_n = \sum_{k=0}^{n} a_k \quad \text{Partial Sum} \]

Can we take \( \lim_{n \to \infty} S_n \) without doing the sum, extrapolating to \( n \to \infty \)? Not in general?

Is this like extrapolating \( \frac{1}{N^2} \to 0 \) in Romberg integration? \( \int_a^b f(x) \, dx = \lim_{N \to \infty} \sum_{n=1}^N f(x_n) \)

II. Basically no. Romberg integration and other works for any function which is analytic in \( \text{Lab 1} \).
Example: \( a_k = \frac{1}{2^k} \), \( a_k \) random \# in \([-1, 1]\)

Given \( S_0, \ldots, S_n \), what is best guess for \( S_{\infty} \)?

\[ \langle S_{\infty} \rangle = S_n \text{, since } \langle a_{n+1} \rangle = \langle a_{n+2} \rangle = \ldots = 0. \]

For a general sum, there is no way to accelerate convergence. Higher terms must be related to lower ones.

- Very different from Romberg integration,
- "Richardson's deferred approach to the limit" which works for general analytic functions even when higher derivatives are unrelated to lower ones (other than rapidly converging).

Good acceleration methods work only when terms \( a_k \) have special analytic structure.

\textbf{Aitken's \( \Delta^2 \) process}

\[ S_n' = S_n - \frac{(S_{n+1} - S_n)^2}{S_{n+2} - 2S_{n+1} + S_n} = S_n - \frac{(\Delta S_n)^2}{\Delta^2 S_n} \]

- Exact for geometrical series \( \sum x^n \) (Derive it from this!)
- \( a_k = \frac{(-1)^k}{k^n} \rightarrow \text{new } \Delta_{\text{new}} \approx \frac{(-1)^k}{k^{n+2}} \) (My fiddling!)

Also extends radius of convergence!
*Recurrence Relations (occasionally quite useful)
*Complex arithmetic (Never used)
*Quadratic equations
  \[ \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \]
  (How to calculate, without dying from rounding errors, often used)
V.2 Numerical Differentiation

How to evaluate $f'(x)$ on the computer:

**Best method:**
- Find $df/dx$ analytically
- Program it into the computer
  [Works only for simple $f(x)$]

**Even better:**
- Have computer analyze program that calculates $f(x)$
- Do automatic differentiation of the program
  [Can be very powerful; doesn't work well for 'if ($x > 3$)' and variable stepsize routines; we use it in finding optimal parameters]

**Worst case**

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} = f'(x) + \frac{1}{2} h f''(x) + ...$$

But if $h \approx 10^{-16}$, truncation error seems OK? **Wrong**.
Small $h \rightarrow$ bad rounding error.
- Roundoff error in $h$: Avoidable
  $x=1$, $h = 1.0 \times 10^{-7} = 0.0000001$

  $x+h^\ast = 1.0000000100000005838$
  \(\not\text{not an exact binary number}\)
  use this for $h$ instead

- Fractional Evaluation error in $f \sim (Ef)$
  - Always at least $2.2 \times 10^{-16}$
  - Sometimes $f$ is itself a numerical integral, or a solution to a differential equation $\Rightarrow Ef \approx 10^{-6}$, or even $10^{-4}$

  \[
  \frac{f(x+h) (1 \pm Ef) - f(x) (1 \pm Ef)}{h} \approx \frac{f(x+h) - f(x)}{h} \pm \frac{f(x+h) \pm f(x)}{h} \\
  \text{Rounding error} \sim \frac{F Ef}{h}
  \]

Minimum error

\[
\text{Error} \sim h f'' + \frac{F Ef}{h} \quad \frac{d \text{Error}}{dh} = f'' \frac{F Ef}{h^2} \Rightarrow h \sim \sqrt{\frac{F Ef}{f''}} \sqrt{\frac{R}{E}}
\]

Best $h \sim \sqrt{E F R}$

Best Error $\sim \sqrt{E F} F/R$

Remember!

Scale of variation $R$

$f'' \sim F/R^2$ $\frac{F Ef}{h^2} \sim R$
Better:
\[ f'(x) = \frac{f(x+h) - f(x-h)}{2h} \]

Even better (classy):
\[ f'(x + \frac{h}{2}) = \frac{f(x+h) - f(x)}{h} \]

Example: E + M
- Gauge field \( A \) naturally lives on bonds

\( h \nu (E) \frac{1}{3} R \)
error \( \sim (E) \frac{3}{2} \)

Good:
\( \frac{f(x+h/2) + f(x-h/2)}{h} \)
Derivative lies on bonds connecting sites on discrete grid

Richardson extrapolation, o o o
Chebyshev, fitting derivative of noisy data ...
I.3

First, some motivation.

Function $f(x)$ [say $\sin(x)$ on $-\pi, \pi$]

Best possible $M$-parameter approximation to $f$

[say, polynomials up to $x^{M-1}$]

What does the error look like?

What is ‘best’?

Here minimax: minimize maximum absolute error,

$$E_{\text{MM}} = \| \epsilon \|_{\infty}$$

If one extremum less bad than the others,

vary parameters to lower others

Expect $M+1$ extrema, all equal to $E_{\text{MM}}$.
II.3.2 Fourier fits to periodic functions

- Fourier sampling theorem
- Convergence of Fourier series for analytic functions
- Exponential convergence of trapezoidal rule for periodic analytic functions

\[ f(\theta) = \sum_{m=0}^{\infty} a_m \cos m \theta + b_m \sin m \theta \]

Assume \( f(\theta) \) even (explained later...)

\[ \approx \sum_{m=0}^{M-1} a_m \cos m \theta + \text{error} \]

Error = \( a_M \cos M \theta + \sum_{m=0}^{M-1} a_m(\ldots) \)

If first term dominates (exponentially fast convergence with big \( M \)), then \( \cos M \theta \) has \( M+1 \) extrema and \( \sum_{m=0}^{M-1} a_m \approx \frac{\text{Total error}}{M} = \frac{e^{-\frac{M\theta}{\pi}} - e^{-\frac{\theta}{\pi}}}{1 - e^{-\frac\theta\pi}} \) (within factor \( 1/e \) of best possible)

(We will see that truncated Fourier series minimizes least-square error \( \| \cdot \|_2 \), rather than \( \| \cdot \|_1 \)).

How to find coefficients \( a_m \)?
- Fast Fourier Transform (later)
- Integrals \( a_m = \frac{1}{\pi} \int_{-\pi}^{\pi} f(\theta) \cos m \theta \, d\theta \)
Chebyshev Approximation

What about polynomial fits on an interval?

Can we find a rapidly converging set of polynomial approximants to \( f(x) \) on \([a, b]\)? Rescale to \([-1, 1]\)

Fourier: \( f(\theta) = \sum a_m \cos m \theta \) on circle

Chebyshev: \( g(x) = \sum c_m T_m(x) \) on \([-1, 1]\)

Trick: Project \( f \) onto circle

\[
\cos 0 = 1 = T_0(x) \\
\cos \theta = x = T_1(x) \\
\cos 2\theta = \cos \theta \cos \theta - \sin \theta \sin \theta \\
\quad = 2 \cos^2 \theta - 1 = 2x^2 - 1 = T_2(x) \\
\cos 3\theta = \ldots = 4x^3 - 3x = T_3(x)
\]

Backward? \( x^n = \cos^n \theta = \frac{e^{i\theta} + e^{-i\theta}}{2^n} = \left( e^{i\theta} + ne^{i(n-2)\theta} + \frac{n(n-1)}{2} \cos (n-4)\theta + \ldots \right) \frac{1}{2^n} \)

\[
T_n(x) = \cos \left( n \arccos(x) \right) = \cos n \theta = \frac{1}{2^n} \left( \cos n \theta + n \cos (n-2) \theta + \frac{n(n-1)}{2} \cos (n-4) \theta + \ldots \right)
\]
$T_n(x)$ has $(n+1)$ local maxima & minima, all at $\pm 1$

$\Rightarrow$ If $c_n$ converges quickly,

error $\approx \left( \frac{1}{1-e^{-n^2}} \right) \cdot \text{min max error}$

$W-$form

Series

$\alpha_m = \sum_{k=0}^{N-1} \delta \left( \frac{k}{N}(k+\frac{1}{2}) \right) \cos \left( \frac{\pi m}{N} \left( k+\frac{1}{2} \right) \right)$

$= \sum_{k=0}^{N-1} \delta (\theta_k) \cos (m \theta_k)$

$\Rightarrow C_m = \sum_{k=0}^{N-1} \delta (\theta_k) T_m (x_k)$

$x_k = \cos \left( \frac{\pi}{N} \left( k+\frac{1}{2} \right) \right)$

$= \sum_{k=0}^{N-1} \delta (\cos \theta_k) \cos (m \theta_k)$

$= \sum_{k=0}^{N-1} \delta (x_{\theta_k}) \cos \left( \frac{\pi m}{N} \left( x_{\theta_k} \right) \right)$

$\Rightarrow C_m = \sum_{k=0}^{N-1} \delta (x_{\theta_k}) T_m (x_{\theta_k})$

$\Rightarrow C_m \approx \text{equally spaced around circle}$

Close relative of Gauss points quadrature

$W = \frac{\sqrt{3}}{2N} \frac{\pi}{N} x_k = \cos \frac{\pi}{N} \left( k+\frac{1}{2} \right)$

tuned for integrals with singularities $\frac{1}{\sqrt{x-a}}, \frac{1}{\sqrt{b-x}}$ at endpoints.
- $x = \exp(\frac{2\pi}{\beta})$

\[
\begin{align*}
(8.19) & \quad 488092130664x^{12} - 1655000819068x^{9} + 4203503503x^{6} - 14465454434x^{3} + 9050x^{0} \\
& + 92740808732x^{7} + 3749089372x^{4} - 101585555564x^{2} + 30371124x^{3}6 + 27950800x^{0} \\
& - 846628x^{9} + 273520x^{6} - 9205054x^{2} + 7716x^{12} + 2362x^{7} - 7951x^{6} + 252054x^{2} + 8x^{0} \\
& - 72x^{9} + 2148x^{6} + 2148x^{12} + 90x^{2} + 14x^{0} + \frac{12x^{12} + 14x^{6} + 214x^{12} + 792x^{6} - 792x^{12} + 90x^{2} + 14x^{0}}{m = 1 - 2x^{9} - 12x^{6} - 14x^{2} + 2}.
\end{align*}
\]

Pade Approximants

Approximant

Diagram curve is 9/10 Pade

Ising model, to 27 terms

Theory, magnetization for

Perturbation

 poles
functions with
Great for
degree $M+N$, $N$ to fit terms of
degree $M$ over
polynomials
forms ratio of
approximant

Pade

Cluster contributes $x_{20}$

Term, 20 broken bonds

Pade Approximants
VI: Special Functions
Where do they come from?
Many origins.

A. Symmetry

• $\sin(x), \cos(x), e^{i\theta} \leftrightarrow \text{Translation Invariance}$
• Spherical Harmonics $\leftrightarrow \text{Rotation } \text{so}(3), \text{su}(2)$
  $Y_{lm}(\theta, \phi)$
  (Legendre polynomials)
• Gegenbauer Polynomials $\leftrightarrow \text{Rotation } \text{so}(4)$
  Invariance (4D)
• Bessel Functions $\leftrightarrow \text{Euclidean group in } 2D \mathbb{E}$
  (Rotations + Translations)
• Spherical Bessel $\leftrightarrow \mathbb{E}_3$ 3D Rot & Trans
  Jacobi functions
• Hermite Polynomials $\leftrightarrow \text{Quantum group}$
  Associated Laguerre polynomials

"Special Functions - A Group Theoretic Approach"
James D. Talman & Eugene P. Wigner

Group Representation Theory
Also: Poincaré group $\rightarrow$ Spin, Mass as quantum numbers
B. Integrals

\[ \int_{-\infty}^{\infty} \frac{e^{-t}}{t} \, dt = Ei(x) \quad \text{Exponential Integral} \]

\[ \int_{0}^{\infty} e^{-t^2} \, dt = \frac{\sqrt{\pi}}{2} \, \text{erf}(x) \]

\[ \int_{0}^{\infty} e^{-t} \, t^{n-1} \, dt = \Gamma(n) \quad \text{And so } n! = \Gamma(n+1) \]

\[ \Gamma(n, x) = \Gamma(n) - \int_{x}^{\infty} t^{n-1} e^{-t} \, dt \]

\[ \int_{0}^{1} t^{a-1} (1-t)^{b-1} \, dt = B(a, b) = \frac{\Gamma(a) \Gamma(b)}{\Gamma(a+b)} \quad \text{Beta Function} \]

\[ \int_{0}^{\infty} e^{-\frac{x^2}{2}} \, dx = \sqrt{2\pi} \quad \text{Dawson's Integral} \]

\[ \int_{0}^{\infty} \cos\left(\frac{\pi}{2} t^2\right) \, dt = C(x) \quad \text{Fresnel Integrals} \]

\[ \int_{0}^{\infty} \frac{\sin t}{t} \, dt = S(x) \quad \text{Cosine Integral} \]

\[ \int_{0}^{\infty} \frac{\sinh t}{t} \, dt = \text{Si}(x) \quad \text{Sine Integral} \]

\[ \int_{0}^{\infty} \frac{\cosh t-1}{t} \, dt = \gamma + \ln(x) + C i(x) \quad \text{Elliptic Integral} \]

\[ \int_{0}^{\infty} \frac{dt}{\sqrt{(a+b) t^2 + (a^2+b^2) t^4 + (a^3+b^3 t^6) + \cdots}} \]

\[ = 0 \]
C. Statistics
You've heard of
Exponential
Gaussian
Poisson
Binomial
Cauchy
I've heard of
Log normal
Student-t
Logistic
Weibull
Gumbel
Frechet

D. Other

Hypergeometric

Airy function
III. Sorting and Selection

- Sorting a long list of N elements takes $N \log N$ time, and not much extra storage.

- Sorting a short list can be done faster by simple algorithms, but for more than 20 elements do a good job.

- A good black-box routine should do well in both short and long lists.

- You may need to write your own, if you are sorting in a weird context. (But sorting pointers should be considered.)
VII. Random Numbers

Desired: Sequence \( R_0 R_1 R_2 \ldots \) of random numbers, with probability distribution \( p(r) \).

Monte Carlo: Encryption (add \( R \)'s to signal), games, ... 

Random: No measurable correlations between different elements of sequence.

Example: Randg 1 \( (\text{seed } j) \)

- Constructor \( \rightarrow \) Uses \( j \) to set internal variable \( V \)
  \[
  V = 64 \text{ bit integer } \quad \overline{0101110111100}
  \]

- Call \( \rightarrow \) Scrambles internal \( V \)
  \[
  V^\wedge = V \gg 21 \quad \wedge = \text{Exclusive or} \\
  V^\wedge = V << 35 \quad \gg = \text{Right bit shift} \\
  V^\wedge = V \gg 4 
  \]

- Returns scrambled \( V \)
  \[
  \text{return } V \times 2^{64} \text{ mod } 2^{64} \text{ (implicitly)} \\
  \]

- int 32() return (int) \( \text{int64()} \)
  [Relatively prime, so \( \text{mod } 2^{64} \) wastes 32 bits]

- double() return \( 2^4 \times \text{int64()} \)
  [More long ints than doubles \( \rightarrow \) hits everyone]

Notice: 'Scrambling' should be 1-1 map which cycles through all integers [except zero] on an irregular fashion.

\( \Rightarrow \text{Exclusive or} \)

\( \Rightarrow \text{Multiplication by } x \)

\( \Rightarrow (a \times V + b) \ \text{mod } m \) with a, b carefully chosen...
Monte Carlo Integration

\[ \int_{V} f \, dx = V \bar{f}. \]

Sample \( f \) at \( N \) random points

\[ \int_{V} f \, dx \approx V \frac{\sum f(k_n)}{N} \]

How accurate is our integral?

Mean-Square Variance

\[ \text{Variance} = \left\langle \left( \frac{V \sum f(k_n)}{N} - V \bar{f} \right)^2 \right\rangle \]

\[ = \left\langle \frac{V^2}{N^2} \left( \sum (f(k_n) - \bar{f}) \right)^2 \right\rangle \]

Cross terms \( \left\langle (f(k_n) - \bar{f})(f(k_m) - \bar{f}) \right\rangle = 0 \)

\[ = \frac{V^2}{N^2} \sum \left\langle (f(k_n) - \bar{f})^2 \right\rangle \]

\[ = \frac{V^2}{N} \bar{f}^2 - 2 \bar{f} \left\langle \bar{f}(f(k_n) - \bar{f}) \right\rangle + \bar{f}^2 \]

Sample means

\[ \bar{f}^2 = \frac{V^2}{N} \left( \bar{f}^2 - \frac{\bar{f}^2}{N} \right) = \frac{V^2}{N-1} \left( \bar{f}^2 - \left\langle f^2 \right\rangle \right) \]

Accounts for errors in \( f \).

Root-mean-square fractional error

\[ = \frac{V}{\sqrt{N}} \sqrt{\bar{f}^2 - \bar{f}^2} = \frac{1}{\sqrt{N}} \sqrt{\frac{\bar{f}^2}{V^2} - 1} \]

Trapezoidal rule, error \( \propto h^2 \approx \frac{1}{N^2} \) \( \Rightarrow \) why Monte Carlo?
(1) Irregular, nasty functions

If $f''$ huge/divergent, but $f^2$ not bad.

(2) High dimensions

$$N = \frac{V}{hd} \rightarrow h^2 \approx \frac{1}{N^{1/d}}$$

Trapezoidal rule loses for $d > 4$

Simpson's rule loses for $d > 8$

(3) Irregular, weird regions
VII.2 Other distributions

Uniform

\[ P(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \]

Exponential

\[ e^{-y} \]

What is \( P(y(x)) \)?

\[ P(y) \, dy = P(x) \, dx \]

\[ P(y) = P(x) \frac{dx}{dy} \]

\[ \frac{dx}{dy} = e^{-y} \]

\[ \int_{0}^{\infty} dx = \int_{y_0}^{\infty} e^{-y} \, dy \Rightarrow e^{-y} = x_0 \]

\[ \Rightarrow y = -\log x \]

Exponential decay with mean time \( \tau \)

\[ \tau \text{ random()} \]

Olden days, only \( 2^{32} - 1 \) random numbers

\[ \Rightarrow \text{zero would show up remarkably often} \]

\[ = -\tau \log(1 - \text{random()}) \]
7.3 Horror stories

- When in doubt, try changing the random number generator.
  - First decades of numerical simulations using random numbers [I hear] mostly wrong, poor generators.
  - Olga and I
    - NR edition #1 had an amazingly poor random number generator (ran2 ??)
    - Olga and I spent weeks looking for a bug:
      - Three random #s i1, i2, i3
      - Flip spin at (i1, i2, i3)
      - Never flipped certain spins! ($1000 reward for flaws in 2nd edition generator, now rescinded)
  - Nick Trefethan (Oxford) and I spent evening
    - Random 17 x 17 matrices, from Matlab
    - Many zero eigenvalues
    - [Linear Consequential algorithm $I_{ij} = (q I + c) \mod m$]
    - Generates points on hyperplanes, etc.
When you are serious, set your own seeds.

Many packages will initialize the random number generator for you.

Mathematica; many other systems

Time of Day, in milliseconds.

- Don't start your batch jobs at midnight.
- Don't launch several jobs in parallel.

Could be unlucky - only rarely would two jobs get exactly the same milliseconds.

[Erratic bugs are the worst.]

Windows & Netscape is historical:

- Time of day + Process id + Username
- Used for SSH, other encryption
- Snooper can find info, = time of day within 1 sec
  ➔ Crack encoding.

Your code could have a bug that occurs rarely.

- If you don't know the seed, you can't reproduce the bug.

When not just fooling around

- Set your seeds
- Don't use seeds twice (except debugging)
- Record as part of filename
IX. Root Finding: \( f(x) = 0 \), find \( x_0 \) \text{ Solve} and

X. Minimization: \( f(x) \) minimized, find \( x_0 \) \text{ Optimize}

Pictures

One dimension

Two dimensional Minimization

\( f(\vec{x}) \) \( x_1 \) \( x_2 \)

Contour Plot

Lines of constant \( f(x) \)
Two-dimensional Root Finding \( \hat{F}(\hat{x}) = 0 \)

\[ F_0(x_0, x_1) = 0 \]
\[ F_1(x_0, x_1) = 0 \]

One dimension
- Fast
- Easy to do
- Satisfying to do well

Multidimensional
- Hard to get close
- Minimization tricky even then
  (Long valleys)
- Root finding harder but even trickier.
IX-X.1 Bracketing (in One Dimension)

Desired: A "guarantee" that the answer is in between $a < x_0 < b$.

Root Finding: $f(a)$ and $f(b)$ have opposite sign

Minimization: $a < b < c$, $f(b) < f(a), f(b) < f(c)$
How to bracket?

Best Method: Explore, think, & bound

Usually similar root finding / minimization

millions of times: find approximate answer,
range, bracket analytically.

No other good method, unless bracketing is easy.

- Check the limits \( \pm \infty \) by doubling
- Fit parabolas...

Except: \( x/e^{-x} \) root

\[ x(e^{-x} + 1) \]
I. X. 1.2 Badly behaved functions
Where analyticity is not helpful

**Root Finding! Bisection**

![Graph: Bisection method diagram]

Pick midpoint
Block in interval
with sign change

To get convergence to tolerance \( \varepsilon \approx \log_2 \left( \frac{1}{\varepsilon} \right) \)
(Not bad: machine precision in 52 steps)

If \( f'(x_0) \approx 0 \), expect near-machine precision in \( x_0 \).

**Minimization! Golden Search**

Note! Expect only half of machine precision
in a minimum.

\[ f(x_0 + \varepsilon) - f(x_0) = \frac{1}{2} \varepsilon \frac{d^2 f}{dx^2} \]

Constant when \( \varepsilon \gg \Delta M \)

(No way to improve minimum)

**Subdivide**?

Maybe \( bc \) \( (f(c) < f(a)) \),
but that presumes smoothness.

Avoid putting \( b \) at a midpoint.
Which interval to subdivide?

\[ ab \text{ is larger; subdivide it,} \]

Where best to put new point \( d \)?

Optimize worst case: make \( dc = ab \).

Converges to self-similar solution

\[
\frac{2zE_0 - E_0}{zE_0} = \frac{b-d}{b-a} = \frac{c-b}{c-a} = \frac{E_0 - zE_0}{E_0}
\]

\[
\frac{2z-1}{z} = 1 - z \quad 2z - 1 = z - z^2
\]

\[
z^2 + z - 1 = 0
\]

\[
z = \frac{-1 \pm \sqrt{1+4}}{2} = \frac{\sqrt{5} - 1}{2} = 0.61803
\]

\[
= \text{Golden Mean} \quad \{ \text{Actually,} \quad \frac{1}{z} = \frac{\sqrt{5} + 1}{z} = \text{Golden mean} \}\]

Converges to \( \sqrt{E_m} \) precision (best possible)

when \[ E_0 Y^{-n} = \sqrt{E_m} \Rightarrow n = -\log_Y \sqrt{E_m} \]

\[ \Rightarrow n = \frac{-1}{2} \log_Y E_m \]

Best possible in \( 38 \) function evaluations. (Again, not bad for worst case.)
One-dimensional smooth functions

A. No derivatives available - root finding

Linear interpolation
\( a, c \)
Keep two points closest to zero \( \Rightarrow \) Secant method
(but lose bracketing!)

Keep bracketing points \( a, b \) \( \Rightarrow \) False position method
(but slow convergence, distant point \( b \) stays forever...)

Parabolic interpolation (with linear and bisection)
\( \Rightarrow \) van Wijngaarden, Dekker, Brent
(Usually best \( * \) works when things are good or tricky).

* Quadratic convergence: \( \varepsilon_{n+1} = C \varepsilon_n^2 \) when good known
\( \Rightarrow \) Extra digits double with each iteration

\( \Rightarrow \) Usually no need for higher-order methods
B. No derivatives - Minimization

Parabolic interpolation
(with 'golden search'
and roundoff control)

$\rightarrow$ Brent's method

C. Root finding, 1D, with derivatives

**Newton's Method**

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2} f''(x) = 0$$

$\Rightarrow$ pick $x_{n+1} = x + h = x - \frac{f(x)}{f'(x)}$

**Error $\epsilon_i$:**

$$\epsilon_{n+1} = \epsilon_i - \frac{f(x + \epsilon_i)}{f'(x)} = \epsilon_i - \frac{\epsilon_i^2 f''(x)}{2 f'(x) + \epsilon_i f''(x)}$$

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

$$= \frac{\epsilon_i^2 f''(x)}{2 f'(x)}$$

Number of decimal points doubles with each iteration.
Problems with Newton

Find low derivative

=> Outer space

Stuck in limit cycle

Combine Newton-Raphson with bisection

D. One dimensional minimization with derivatives
- Numerical recipes suggests not using fancy fits, but finding a root of $f'(x)$ with the secant method

Common Error:
$ \alpha = f(x) \neq f'(x)$
E. Multiple roots and deflation

What about functions with several roots? How to avoid returning to old ones?
- Bracket them individually
- If you know that they are simple
  \[ f(x_0) = 0, \quad f'(x_0) \neq 0 \]
  divide \( f \) by \( (x - x_0) \)
  \textbf{Deflation}
  This is especially useful for polynomials root finding
  Know how many roots to look for (in complex plane)

Doesn't work in higher dimensions
(except complex roots of analytic functions)
IX - X 4 Multiple Dimensions: Root Finding

A. Newton-Raphson root finding

\[ F_i(x + \delta x) = F_i(x) + \sum_{j=0}^{n} \frac{2F_i}{\partial x_j} \delta x_j + O(\delta x^2) \]

\[ J_{ij} = \frac{\partial F_i}{\partial x_j} \text{ Jacobian} \]

\[ F(x + \delta x) = F(x) + J \cdot \delta x \]

\[ \delta x = -J^{-1}F \]

Solve \( J \cdot \delta x = -F \) using LU, SVD, ...

Iterate to convergence

- Typically will not converge unless \( x_0 \) close to a root. (Perversity, not deep truth).

Don't be fooled into minimizing \( F^2 \)

- Much less efficient
- Much lower accuracy (\( \sqrt{EM} \))
- Trapped in local minima

\[ F(x + \delta x) = F(x) + J \cdot \delta x \]

\[ = F(x) - x F'(x) \delta x + o(\delta x^2) \]

\[ x_{n+1} = x_n + \chi \delta x \]

Vary Newton step until \( F(x_{n+1}) < F(x_n) \), see NR

\[ F(x + \delta x) = F(x) + J \cdot \delta x \]
B. Newton-Raphson in Function Space

- Nash Embedding Theorem
  Every Riemannian manifold of dimension $m$ can be embedded in $\mathbb{R}^n$ (with $n \geq m^2 + 5m + 4$) preserving all distances and angles isometrically.

- Klein bottle is 2-D manifold, needs 4-D space to avoid self-intersection

- Torus $x^2 + y^2 = z^2 + w^2 = R^2$ in 4D has no curvature
  \[ \implies \text{needs 4D} \]

Nash used Newton's method to look for a non-self-intersecting, isometric solution (and clever smoothing techniques).

Similar ideas used in proving the KAM theorem, 'solving' the famous three-body problem [transparency]
The KAM Theorem

Why the Earth has not left the Solar System

- Torus Cross Sect
- Chaotic Regions
- Commensurate Orbits
- Irrational Jovian

Earth, Big Jupiter, Sun

KAM Torus
Phase Space

Torus Cross Section = Curve

Chaotic Regions = Commensurate Orbits

Irrational Jovian Year: Safe
C. Root Finding in Multidimensions w/o Derivatives
- Broyden's Method

- **Approximate Jacobian** $B$
  (start with identity matrix $I$?)

- **New point** given by $B \cdot \delta x = -F$

- **New $B$ chosen to include information from new force evaluation**
  \[
  B_{i+1} \cdot \delta x_{i+1} = \delta F = F(x_{i+1}) - F(x_i)
  \]
  [Secant method along $\delta x_i$]

- **Minimal change to $B$ consistent with new information**
  Remove error when handed $\delta x_i$
  \[
  B_{i+1} = B_i + (\delta F_i - B_i \delta x_i) \otimes \frac{\delta x_i}{\delta x_i \cdot \delta x_i}
  \]

- **Use Sherman-Morrison trick (or variant using QR decomposition) to update $B$ and $B^{-1}$ for Newton's method**
Multiple Dimensions: Minimization

Best methods depend a lot on the kind of problem. I'll mention two cases: Glassy and Sloppy.

A. Glassy systems

- $M$ atoms $\Rightarrow$ $3M$ or $N$ coordinates
- Local interactions
- Local metastability (one atom in 10, say, has two positions metastable)

$$2^{N/10} = 2^{3N/10} \approx e^{3 \log^2 N} \text{ (local minima)}$$

- Use local moves, large $N$
  - Monte Carlo move one atom at a time
  - Quick min: accelerate downhill, stop each atom when $V_i \cdot E_i > 0$

- Genetic Algorithms
- 'Fourier-acceleration', other methods for long wavelength relaxation, polishing
  - Not conjugate gradient, quasi Newton, simplex in general
Global minima hard or impossible to find
- Often NP-complete (read: exponentially hard as $N \to \infty$)
  
  Time to $\mathcal{O}(N^2)$

(Roughly, check all local minima)
- Genetic algorithms, ...
B. Sloppy systems

Multiparameter optimization
- Fitting models to protein dynamics in cells
- Optimizing performance of accelerators (ILC)
- Insect flight, fitting exponentials/polynomials...

Optimizable function $f(k)$ grows quickly along only a few 'stiff' directions.

Other 'sloppy' directions form long, narrow valleys - often $1000:1$ aspect ratios.

Key challenge: Force points in wrong direction

Want 'scale invariant' methods that eventually invariant under linear coordinate changes.
Initial Search: Nelder Mead 'amoeba'
[Show transparency]

* Often gets stuck on 'plateau'

Some parameters so wrong computer can't find direction.

- No substitute for thinking.
  Which parameters need to change to fix things? Missing features?

* Search subspaces

Often only 4-5 'stiff' directions

Ridge dimension N-5

Pick 5 'important' parameters to search for to locate ridge

Polishing minima:
Levenberg–Marquardt (discuss later, only non-linear least squares)
Monte Carlo...
Summary: The Simplex Method

- Original Sim
- high
- low
- reflection
- reflection and expansion
XII-XIII Fourier methods

A. Why are Fourier methods important?

- Humans hear frequencies
  Air $P(t) \rightarrow$ Brain $|P(w)|^2$ (actually nonlinear, time dependent wavelet?)

- Diffraction experiments measure
  Fourier components
  X-rays, neutrons, electron scattering
  $\rightarrow p(k)$ elastic $\Rightarrow$ average over
  $\rightarrow s(k,w)$ inelastic $\Rightarrow$ beam size

- Common mathematical operations
  simpler in Fourier space

  Derivative
  $\frac{df}{dx} \rightarrow -i \, k \, \hat{f}(k)$

  Integral
  $\int_0^k f(x) \, dx \rightarrow \frac{\hat{f}(k)}{-i \, k}$

  Convolution
  $(f * g)(x) = \int f(x) \, g(y-x) \, dx \rightarrow \hat{f}(k) \, \hat{g}(k)$

  Correlation function
  $C(x_0) = \int f(x) \, f(x+x_0) \, dx \rightarrow \hat{C}(k) = |\hat{f}(k)|^2$
- Fourier modes are eigen functions (irreducible representations) of the translation group. \( T_\Delta = \text{Translate function by } \Delta \)

\[
T_\Delta \left[ e^{ikx} \right] = e^{i(k(x-\Delta))} = (e^{-i\Delta})e^{ikx}
\]

\( \text{Eigenvalue} \quad \text{Eigenfunction} \)

\[ \rightarrow \text{Linear problems with translation invariance have Fourier solutions} \]

\[ \text{Linear constant coefficient ODEs} \quad -a\frac{d^2}{dx^2} + b \frac{d}{dx} c = 0 \]

\[ \text{try } e^{iwt}, \text{ solve for } w \]

\( \text{Schrödinger, wave equations solution } e^{i(k(x-wt))} \)

Physics explanation: \([H, T_\Delta] = 0\), commuting operators have simultaneous eigenvectors

Math explanation: Space of solutions can be decomposed into "irreducible representations of translation group" = Plane waves

Rotation invariance \( \rightarrow \) Spherical Harmonics \( \ell, m \)

\( SU(3) \rightarrow 8\)-fold way families of baryons & mesons

Poincaré group (Lorentz + Rotations) \( \rightarrow \) Mass & Spin
B. Aliasing & the Fourier Sampling Theorem

Fourier Transform in Time

\[ \hat{f}(\omega) = \sum_{-\infty}^{\infty} e^{i\omega t} f(t) \, dt \]

Numerical discretization

\[ \hat{f}(\omega) \approx \sum_{m=-\infty}^{\infty} e^{i\omega mh} f(mh) \cdot h \]

\[ = \sum_{-\infty}^{\infty} e^{i\omega t} f(t) [h \Delta(t)] \, dt \]

Comb function \( \Delta(t) = \frac{\sin(\pi t/2h)}{\pi t/2h} \)

\[ \Delta(t) = \sum_{m=-\infty}^{\infty} \delta(t-mh) = \frac{1}{h} \sum_{n=-\infty}^{\infty} e^{i\pi nh} \]

\[ \omega_0 = \frac{2\pi}{h} \]

Discrete Fourier transform

\[ \hat{f}(\omega) \propto \sum_{n=-\infty}^{\infty} e^{i\omega t} f(t) \sum_{n=-\infty}^{\infty} e^{i\omega nt} \, dt \]

\[ = \sum_{n=-\infty}^{\infty} e^{i\omega t + \omega nt} f(t) \, dt \]

\[ = \sum_{n=-\infty}^{\infty} \hat{f}(\omega + n\omega_0) \]

\[ = \hat{f}(\omega) + \sum_{n=1}^{\infty} \hat{f}(\omega + n\omega_0) + \hat{f}(\omega - n\omega_0) \]

Error
- Fourier power outside \((-\frac{\omega_0}{2}, \frac{\omega_0}{2})\) aliased to inside the interval. [Windowing, ...]

- If signal has no Fourier components outside \(\pm \omega_0/2\) (bandwidth limited), discrete Fourier transform is exact (Fourier sampling theorem)

- Simple picture: sample \(e^{i(\omega_0+\omega)t}\) same as \(e^{i\omega t}\)

- Bandwidth limited? 
  \[ \text{Exact Formula} \quad \hat{f}(t) = \lim_{N \to \infty} \sum_{m=-\infty}^{\infty} \frac{\sin(\frac{\omega_0}{2}(t-m\Delta))}{\pi(t-m\Delta)} \]

  \(\Delta\) determines \(f(t)\).
C. Fast Fourier Transform

How many operations are needed to compute
\[ \hat{f}_k = \sum_{n=0}^{N-1} f_n e^{2\pi i kn/N} \] for all \( N \) values of \( k \) when \( k = 0, \ldots, N-1 \).

No need for lots of cosines & sines: compute \( W = e^{2\pi i/N} \), then \( \hat{f}_k = \sum_{n=0}^{N-1} W^{nk} f_n \).

Naively, \( N^2 \) multiplications to compute \( \hat{f}_k \) for all \( N \) values of \( k \).

Can we do better? Trick: separate sums into even & odd points in the sum:

\[ \hat{f}_k = \sum_{m=0}^{N/2-1} e^{2\pi i k (2m)/N} f_{2m} + e^{2\pi i k (2m+1)/N} W^{-k} f_{2m+1} \]

\[ = \sum_{m=0}^{N/2-1} e^{2\pi i k m/(N/2)} (f_{2m} + W^{-k} f_{2m+1}) \]

\[ = \hat{f}_k + W^{-k} \hat{f}_k \]

Compute two transforms \( N/2 \) long, then replicate to length \( N \).
Pointwise multiply second by vector \( W^{-k} \).

Time \([N]\) = 2 Time \([N/2]\) + \( CN \)
If \( N \) is a power of two (say \( 2^m \))

\[
\text{Time}[N] = 2^m \text{Time}[1] + MCN
\]

\[
= N + CN \log_2 N = N \log N
\]

Much faster than \( N^2 \).

Almost as fast as dot products, discrete derivatives, ...

Not a power of two?

- Fancy methods for

\[
2^m 3^r 5^s \ldots \to \text{worse for lengths with big prime factors}
\]

- Crucial if you know period & can't control it

\[\text{Post-doc: period} = \text{Fibonacci}[n];\]

\[
\text{Fib}_8 = 2584 = 2^3 \cdot 17 \cdot 19 \quad \text{good compared to}
\]

\[
\text{Fib}_7 = 1597 \quad \text{prime}
\]

\[
\text{Fib}_9 = 4181 = 37 \times 113
\]

- Fancy methods for primes too.

- Change period / pad with zeros / 000
Dr. Fourier Headaches

FFTs save computer time, at the expense of human time. Be prepared to fuss with details.

(1) Conventions.

- Some engineers use $j = \sqrt{-1}$ instead of $i$.
- Sometimes $i^2 \rightarrow -i^2$ are swapped.

Physics & \[ \hat{y}(\omega) = \int_{-\infty}^{\infty} y(t) e^{-i\omega t} dt \]
but \[ \check{y}(k) = \int_{-\infty}^{\infty} y(x) e^{-i k x} dx \]
so that $e^{i k \cdot x - i \omega t}$ propagates forward in time & space.

- Some (Numerical recipes) use $\hat{y}$ and $\check{y}$.
- Some (Physics) use $\omega = \frac{2\pi}{T}$ and $k = \frac{2\pi}{L}$.
- Various factors of $T, V, N, 2\pi$ for transform & inverse transform.
- Power spectrum conventions even more varied.
(2) Types
- Fourier Series (periodic in real space)
  - Fourier Transform (all real space, all Fourier space)
  - Fourier Series (periodic in real space, discrete in Fourier space)
  - Lattice Functions (Discrete in real space, periodic in Fourier space)
    - Crystal (phonons, wave functions, quasi-particles)
    - Brillouin zone
- FFT (Discrete and periodic in real and Fourier space)

(3) Unpacking
- FFT doesn't give \( \hat{u}_n = \frac{2\pi n}{N} \), build by hand
- FFT gives negative \( \omega \)'s at end

2D FFT reassembled into zero-centered region

Real-valued functions special FFTs, twice as fast, no small imaginary parts
Example: Derivatives and Integrals

\[ \frac{d^2 f}{dx^2} = (ik)^2 \hat{f}(k) \]
\[ = -k^2 \hat{f}(k) \]
\[ \left( \frac{d^2 f}{dx^2} \right)_n = -k_n^2 \hat{f}(k_n) \]

What to choose for \( k_n \)?

- Small positive \( k \)
  \[ k_n = \frac{2\pi n}{\Delta x} \]

- Small negative \( k \)
  \[ k_n = \frac{2\pi (N-n)}{\Delta x} \] at end of sequence

- Cross at \( N/2 \). Should \( k_{N/2} \) be positive or negative? Either \( k_n \) sense

\[ \frac{df}{dx} = ik \hat{f}(k) \]: Different choice. (4 grad-student weeks, my group) to avoid complex derivatives.

\[ k_{N/2} = 0 \implies \text{Required so that} \]

\[ \frac{d^2 f}{dx^2} = \frac{\hat{f}(k) + \hat{f}(-k)}{2i} = -ik \hat{f}(k) \]

\[ \frac{\hat{f}(k)}{\hat{f}(-k)} \]

\[ \text{Real} \left( \frac{d^2 f}{dx^2} \right) \]
\( \hat{f}(-k) = \hat{f}(k)^* \quad \text{for real } f(x) \)

\[
\left[ ik \frac{d \hat{f}(k)}{dk} \right] = -i k \frac{\hat{f}(k)}{dx} \quad \text{for most } k
\]

\[\frac{d\hat{f}}{dk} \quad \text{also FT of real } \frac{df}{dx} \]

\( \text{At } N/2, \quad \hat{f}(-k) = \hat{f}(k) = \hat{\tilde{f}}(k_{N/2}) \)

FET has \( N/2 \) same

\[ \Rightarrow \hat{f}_{N/2} \quad \text{real} \]

So must have \( ik_{N/2} \hat{f}(k_{N/2}) = 0 \Rightarrow \text{pick } k_{N/2} = 0 \)

\[ \hat{\tilde{f}} = \frac{\hat{f}(k)}{ik} \quad \text{Pick } k_0 = k_{N/2} = 0 \quad \int \]

Note! If \( \hat{f} \) periodic, \( \hat{\tilde{f}} \) periodic only if \( \hat{f}(0) \) mean of \( f(x) = 0 \Rightarrow \text{must calculate } k=0 \) separately.

\[ \text{Mean zero } \Rightarrow \text{avoid } \frac{0}{0} \Rightarrow \text{calculate set to } \frac{0}{0} \]

\( N/2 \) term zero

\[ \text{Mean, } N/2 \text{ term non-zero } \Rightarrow \text{Must calculate them separately anyhow.} \]
XVII. Ordinary Differential Equations (ODE)

Initial value problem:
\[ \frac{dy}{dx} = f(x, y) \]

Solve for \( y(x) \)
starting from \( y_0 \) at \( x_0 \).

\[ \begin{aligned}
\frac{dy}{dx} &= f_i(x, y_0, y_1, \ldots, y_N) & i = 0, \ldots, N-1 \\
\frac{d^2y}{dx^2} &\rightarrow \frac{d^3y}{dx^3} = \mathcal{F}(y) & \mathcal{F}(y) = (z)
\end{aligned} \]

- Many methods, shared interfaces
- Easy to swap
- Try several, find what's best for your problem

- Two-point boundary value problems, partial differential equations → PDE
A. General Purpose Methods (Non-conservative)

1. Euler method

\[ \tilde{\gamma}(x+h) = \tilde{\gamma}(x) + h f(x, \tilde{\gamma}(x)) + O(h^2) \]

- Simple
- Not very accurate
- \( h^2 \) error per timestep

First order method

[\( n^{th} \) order is accurate to \( O(h^n) \)]

Not very stable

Often used in PDEs
- Minimizes memory storage
- Tiny timestep set by stability, plenty accurate
2. Midpoint method (second-order Runge-Kutta)

\[
\begin{align*}
\vec{k}_1 &= h \vec{f}(x, \vec{y}(x)) \\
\vec{k}_2 &= h \vec{f}(x + \frac{h}{2}, \vec{y}(x) + \frac{\vec{k}_1}{2}) \\
\vec{y}(x+h) &= \vec{y}(x) + \vec{k}_2 + O(h^3)
\end{align*}
\]

- Example of higher order methods
- Used for extrapolation to limit (below)
3. 4th-order Runge-Kutta

\[ k_1 = h f(x, y(x)) \]
\[ k_2 = h f(x + \frac{h}{2}, y + \frac{k_1}{2}) \]
\[ k_3 = h f(x + \frac{h}{2}, y + \frac{k_2}{2}) \text{ Better midpoint} \]
\[ k_4 = h f(x + h, y + k_3) \text{ First endpoint} \]

\[ y(x+h) = y(x) + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5) \]

Very traditional, not too messy

Not fancy, but not embarrassing

(Dopri 853 recommended; eighth order, 12 function evaluations)
4. Bulirsch-Stoer

- Richardson extrapolation to $h \to 0$.
- Uses modified midpoint method for extrapolation to $H = n \cdot h$

$$Z_0 = y(x)$$

$$Z_i = Z_{i-1} + 2h \cdot \frac{y(x+ih, z_{i-1})}{2} + h \cdot f(x+H, z_i)$$

$$y(x+H) = \frac{1}{2} \left( Z_n + Z_{n-1} \right) + h \cdot f(x+H, z_n) + O(h^2)$$

- Even points connected by flat odd points
- One derivative per step (2 for regular midpoint)

- Error only contains even powers of $h$

- Simpson's rule:
  $$\gamma(x+H) = \frac{4}{3} y_{5}(x+H) - \frac{1}{3} y_{3/5}(x+H) + O(h^5)$$
  (1.5 $f$'s per step as good as 4th order Runge-Kutta, 4 $f$'s per step)

- Extrapolate to $h \to 0$. 
Warning:

Non smooth problems won't extrapolate

- Singularities during evolution
- Approximate right-hand-sides
  \[
  \frac{dy}{dt} = \begin{cases} \text{Approximate function} \\ \text{numerical integral} \\ \text{interpolated sine(x)} \end{cases}
  \]

⇒ Bulirsch-Stoer will go crazy
⇒ Runge-Kutta will give ok answers, just not as fast
B. Errors + Adaptive Stepsize

1. How big should step-size $h$ be? (Burden-Still or size of $H$)
   - Not too small.
     - Expensive
     - Rounding error: not usually the bottleneck
   
   - Maybe not size of desired output
     Movie: analysis: want $y(m^2)$ spaced $\Delta$
     $h << \Delta$ often for accuracy
     If $h \geq \Delta$, use dense output between sampled points
     (polynomial interpolation to same accuracy as solution)

   - Often changes with time/position

* Depends on accuracy desired.
Global accuracy vs. time-step error

If I use an $n^{th}$ order Runge-Kutta with step-size $h$, to a final position $t = Nh$, what will my total error be? [in $n^{th}$ order error is $O(h^n)$]

"Cumulative" error $E \approx Nh^{n+1} = h^n \cdot H$ — that's why it's called $n^{th}$ order.

But is this a good estimate?

- OK for periodic motion
- Errors smaller for convergence to fixed point
  \( (H \to \infty, \text{error} \to 0) \) \( \chi(t) \)

- Errors grow exponentially for chaotic motion

\[ \text{Often infeasible to get } \]
\[ \text{correct trajectory - make sure} \]
\[ \text{motion is qualitatively accurate} \]
\[ \text{(attractor converged, not trajectories)} \]

⇒ Focus on time-step error
3. Adaptive time steps

Amazingly useful trick! Use the computer to change the time step.

- Take two different methods (hopefully sharing function evaluations)
- Monitor difference between two

  - If bigger than required tolerance, use error to estimate repeated step size \( h' \)

  - If smaller than required, estimate new step size \( h_{n+1} \)

Example: 4th order Runge Kutta, total step \( 2h \)

One step of \( 2h \)
\[ y(x+2h) = y(x+2h) + (2h)^5 \phi \]

Two steps of \( h \)
\[ y(x+2h) = y(x+2h) + 2(h^5) \phi \]

\[ \Delta = |y_1 - y_2| \]
\[ \text{max} + |\Delta| \leq 2(2\alpha) + 1 \cdot y_1 \frac{\Delta}{2\alpha} \]

Attributions:
If \( h \) produced an error \( \Delta \), and error \( \Delta_h h^n \), and desired error is \( \Delta_0 \), choose

\[
\text{new } h' = h \left( \frac{\Delta_0}{\Delta} \right)^{\frac{1}{n+1}}.
\]

However:

- \( \Delta \) can be near zero by accident. Use \( \max(\Delta n, \Delta n-1) \).

- If stability is bottleneck, don't want step-size to keep going below stability boundary.

\[
\rightarrow \text{control theory, } h_{n+1} = h_n \frac{\Delta_0^k \beta}{\Delta_n^k / \Delta_{n-1}^k}, \quad \ldots
\]

3. Qualitative errors: Conservation laws

[See homework]
17.5 Stiff differential equations

How large a time step \( h \) can we take?

**Instabilities**

Simple case: \( y' = -cy \quad (c > 0) \)

\[ y(t) = e^{-ct} \]

**Big \( c \):** when \( y < (\text{absolute error cutoff}) \)
want big steps \( h \)

---

**Problem:** All the methods go nuts so far go nuts if \( h > \frac{1}{c} \) \{Time of experiment \Rightarrow Natural time scale\}

**Example:** Euler step \([\text{explicit, or forward Euler}]\)

\[ y_{n+1} = y_n + h y_n' = y_n (1 - ch) \]

Goes unstable if multiplier \(|1+ch| > 1, h > \frac{1}{c}\).
Even worse, if multiple scales of time inside equations

\[ u' = 0.998u + 1.998v \]
\[ v' = -0.999u - 1.999v \]

\[ u(0) = 1 \quad (u=2y-z \ \text{... solve}) \]
\[ v(0) = 0 \quad (v=-y+z) \]

\[ u = 2e^{-x} - e^{-1000x} \]
\[ v = -e^{-x} + e^{-1000x} \]

Step size \( << 1/1000 \), boring answer except near \( t=0 \). stiffness

When does this happen?

- Separation of Scales
  - Superconducting Vortex Cores,
  - \( \frac{\gamma}{k_F} >> \xi >> \lambda \)
  - Magnetic Electron wavelengths \( \text{Superconducting Length} >> \text{Coherence Length} \)

- Singular Perturbations / Boundary Layers

\[ \frac{\partial^2 y}{\partial t^2} + \frac{\partial y}{\partial t} = 0 \]

Constant tiny terms - 1st order ODE: \( y(0) \) defines trajectory with tiny terms \( y'(0), y(0) \) boundary conditions
Solution: Use stiff integrator

- Implicit (or backward) Euler
\[ y_{n+1} = y_n + h \frac{y'}{y_{n+1}} \]

For \( y' = -cy \),
\[ y_{n+1} = y_n - he_y \]
so solve
\[ y_{n+1} = y_n \left( \frac{1}{1 + hc} \right) \]
The solution is stable if \( \frac{1}{1 + hc} < 1 \) guaranteed.

- Works for vector problems
\[ \vec{y}' = -C \cdot \vec{y} \]
\[ \vec{y}_{n+1} = (1 + Ch)^{-1} \vec{y}_n \]
so long as all eigenvalues of \( C \) are \( \text{Rel}(x) > 0 \).
Note: Matrix inversion at each time step
\[ y_{n+1} = f(y) \]

- Nonlinear problem: don't know \( y_{n+1} \) without knowing \( y_{n+1} \)? Semi-implicit Euler
\[ y_{n+1} \approx y_n + h \left[ f(y_n) + \frac{d}{dy} \left| y_n \left( y_{n+1} - y_n \right) \right. \right] \]
\[ y_{n+1} \approx y_n \left( 1 - h \frac{d}{dy} \right)^{-1} f(y_n) \]... solve
Stiff versions of

* Runge-Kutta $\Rightarrow$ Rosenbrock
* Bulirsch-Stoer
* Predictor-Corrector

MORE

Galerkin, relaxational methods
18. Two-point boundary value problem

\[ \begin{align*}
Y_0 & = Y_f \\
1 & = x_0 = x_f
\end{align*} \]

Know \( Y_0 \), not \( \frac{dY}{dt}_0 = V_y \).

Want \( Y(x) = Y_f \).

**Shooting Method**: Change angle of cannon until see where ball hits target.

Often used for eigenvalues.

Find Root on \( f(V_y) = \left\{ \begin{array}{l}
\text{Solve } \text{ODE int. } \{Y_0 \} [X] - Y_f \\
\text{Solve for position at } x
\end{array} \right\} \)

(a) \text{ ask for more precision for inner ODE int than requested for findroot}

\( f \) low precision may waste much effort
(b) Multivariable shooting can be tricky (Hand tune to near root?)

(c) Many problems complicated boundary conditions (Relaxation)

(d) Often very fussy

- Try integrating backward, or meet at middle
- Try relaxation methods

Relaxation Method

\[
\begin{align*}
\text{Solve } & \quad Y_0 = Y(0) \\
& \quad \frac{Y_{k+1} - 2Y_k + Y_{k-1}}{2h} = -\frac{g}{v_x^2} \\
& \quad Y_N = Y(x)
\end{align*}
\]
17.7 *Stochastic Reaction Networks*

Chemical reactions

\[ M + M \xrightarrow{k_b} D \]

\[ D \xrightarrow{k_u} M + M \]

Large numbers:

\[
\frac{d[M]}{dt} = -2k_b[M]^2 + 2k_u[D] \\
\frac{d[D]}{dt} = k_b[M]^2 - k_u[D]
\]

Small numbers - tiny volume inside cell

- \[ M \approx D \approx \]\ few molecules?
Stochastic cells: simple dimerization reaction

- simple dimerization reaction
  - homodimerization: $M + M \leftrightarrow D$
  - (as distinct from heterodimerization: $A + B \leftrightarrow AB$)
- introduce Petri net representation
  - places (circles): molecular species
  - transitions (rectangles): chemical reactions, parameterized by rate constants
  - arcs (directed segments): stoichiometric weights
- compare stochastic and deterministic simulations
  - deterministic
    - $\frac{dy}{dt} = f(y; k_b, k_u); y = (M, D)$
  - stochastic
    - Gillespie algorithm

Petri net for $M + M \leftrightarrow D$

Stochastic vs. deterministic simulation
Gillespie algorithm

- Gillespie’s “Direct Method”, a.k.a. continuous time Monte Carlo, or the Bortz-Kalos-Lebowitz algorithm
- A stochastic method for simulating reaction dynamics
  - Pick at random a reaction to occur next, and a time at which it will occur (consistent with reaction rates)

Petri net for $M+M \leftrightarrow D$

\[ M+M \rightarrow D: k_b [M] ([M]-1) \]
\[ D \rightarrow M+M: k_u [D] \]
\[ \text{Total rate} = \Gamma = k_b [M] ([M]-1) + k_u [D] \]

Next reaction drawn uniformly from weighted rates
Next reaction time $t_{\text{wait}}$ drawn from probability distribution
\[ \rho(t) = \Gamma \exp(-\Gamma t) \]
The Repressilator

- Repressilator
  - Repressor Oscillator
    - engineered synthetic system encoded on a plasmid (introduced into E. coli)
    - oscillatory mRNA/protein dynamics from mutually repressing proteins
    - TetR inhibits λ cI inhibits LacI inhibits TetR (rock-paper-scissors)
  - paper describes both experimental system and mathematical models
    - ODE-based model
    - stochastic, reaction-based model
The Repressilator reaction network

TetR repression by LacI: modeling via Petri nets

- **Binding** (A,B,C, $k_b$)
  - $A + B \rightarrow C$: rate $= k_b [A][B]

- **Unbinding** (C,A,B, $k_u$)
  - $C \rightarrow A + B$: rate $= K_u [C]

- **Catalyzed Synthesis** (C,P, $\gamma_m$)
  - $C \rightarrow C + P$: rate $= \gamma_m [C]$ (transcription)

- **Catalyzed Synthesis** (C,P, $\gamma_p$)
  - $C \rightarrow C + P$: rate $= \gamma_p [C]$ (translation)

- **Degradation** (C, $k_d$)
  - $C \rightarrow \emptyset$: rate $= k_d [C]

**Coarse-grained Continuum Model**

\[
\frac{dm_i}{dt} = -m_i + \frac{\alpha_i}{(1 + p_i^3)} + \alpha_c_i \\
\frac{dp_i}{dt} = -\beta(p_i - m_i)
\]

\[
\begin{align*}
(i &= \text{laci.tetR.cl)} \\
(j &= \text{cl.laci.tetR)}
\end{align*}
\]
Noise in the Repressilator

- **shot noise**
  - fluctuations due to fact that chemical numbers are discrete and potentially small
- **telegraph noise**
  - fluctuations due to fact that some states (e.g., promoter bound by protein) are either on or off
- can scale parameters in model to accentuate or diminish different types of noise
Biomolecular reaction networks: gene regulation & the Repressilator

Phys 682/ CIS 629: Computational Methods for Nonlinear Systems
Processing information to coordinate activity

regulatory & signaling networks as information processing systems

coordinating the processing of matter and energy
Stochastic/Deterministic Repressilators: the use of inheritance

- Inheritance allows for the definition of families of related classes, distinguished from one another by degrees of specialization
  - base class / superclass: more generic; derived class / subclass: more specialized
- Inheritance also allows for code reuse (common behavior can be defined in the superclass)

```python
class Repressilator:
    # code to define chemicals & reactions
    def __init__(self, ...):  # initialize a Repressilator
        def AddChemical(self, chemical):  # add a chemical
            def AddReaction(self, reaction):  # add a reaction

class StochasticRepressilator (Repressilator):
    def ComputeReactionRates(self):
        # compute instantaneous rates for Gillespie alg.
    def Step(self, dtmax):
        # implement Gillespie alg. for time up to dtmax
    def Run(self, tmax, delta_t):
        # run Gillespie steps for time up to tmax

class DeterministicRepressilator (Repressilator):
    def dcdt(self, c, t):
        # return right-hand-side for ODE integration
    def Run(self, tmax, dt):
        # integrate ODE for time up to tmax
```
20. Partial differential equations

• Solving for function $u$ of more than one independent variable

  $\ddot{u}(x,t)$, given initial $\dot{u}(x,0)$

  $\ddot{u}(x)$, given boundary values at $x=\pm L$, $(0, \pm L)$

• Zoo of different kinds of problems

  - Linear
  - Smooth (Diffusion, Elasticity)
  - Nonlinear
  - Fractal (Turbulence)
  - Weird boundaries (Airplanes)

• Zoo of different methods

  - Finite Elements
  - Airplanes, bridges
  - Finite Volume
  - Fluid flow in curved pipes
  - Finite Difference
  - Spectral
  - Multigrid
  - Upwind/Godunov/…”
Partial Differential Equations
Boundary value problems

Solving elastic theory for helicopter spiral bevel gear (Ingraffea)
Boundary: fixed loading outside (or zero), fixed displacement inside
Finite elements $\rightarrow$ polyhedra, (non)linear systems of equations
Flexible, efficient, special elements for crack tips
Weak solutions; shape functions; preconditioning;
Partial Differential Equations
Initial value problems: shocks!

Sonic booms, traffic jams, plasticity (Limkumnerd, Chen, Choi)
Singularity formation in finite time
Finite difference method: regular grid, time-step
Stability, fidelity more crucial than accuracy (accuracy tricky)
Upwind, Lax-Friedrich, Godunov, …
20.1 Stability of initial value problems

Suppose I'm studying the evolution of some small spatial feature, of size $\Delta x$.

Often, to decrease the fuzziness $\Delta x$, you must also decrease the time step $\Delta t$ (even though nothing much happens that fast.)

To see this, we use linear stability analysis.

- Start with a boring solution (say $y_0 \equiv 0$).
- Linearize the equations about $y_0$.
- Add a small perturbation $\delta u$ (translation invariant? $\delta u = e^{i \mathbf{k} \cdot \mathbf{x}}$).
- See if it decays or grows.
Example: \( \frac{\partial u}{\partial t} = -\nu \frac{\partial u}{\partial x} \) \] [solution \( u = f(x-\nu t) \)]

\[ u_0 = 0 \quad \nu \]

Already linear \( \nu \)

\[ x_j = x_0 + j \Delta x \]

\[ t_n = t_0 + n \Delta t \]

Try simplest:\n
\[ \frac{u_j^{n+1} - u_j^n}{\Delta t} = -\nu \frac{u_j^{n+1} - u_j^{n-1}}{2\Delta x} \]

Growth factor \( \xi \) per time step

\[ u_j^n = \xi^n e^{i k\Delta x j} \]

\[ e^{i k \Delta x (j+1)} = e^{i k \Delta x j} \]

\[ (\xi^n - i ) = -\nu \frac{\xi^n e^{i k \Delta x j} - \xi^n e^{i k \Delta x (j+1)}}{2\Delta x} \]

\[ = -\nu \frac{\xi^n e^{i k \Delta x} - \xi^n e^{i k \Delta x}}{2\Delta x} \]

\[ \xi^n = 1 - i \frac{\nu \Delta t}{\Delta x} \sin k \Delta x \]

\[ |\xi|^2 = 1 + \frac{\nu^2 \Delta t^2}{\Delta x^2} \sin^2 k \Delta x \]

But \( |\xi| > 1 \) for all \( k \neq 0 \)

Unconditionally unstable.
OK, PDEs are tricky. We need to fiddle to get a method that works, and then see how $\Delta t$ depends on $\Delta x$.

**Lax method:** Blur $u^n$ sideways in time derivative

$$u_j^{n+1} = \frac{u_j^n \text{ blurred} + u_{j+1}^n}{2}$$

$$\frac{u_j^n - u_{j-1}^n}{\Delta t} = -\nu \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}$$

Solve...

$$\xi = \cos k \Delta x - i \frac{\nu \Delta t}{\Delta x} \sin k \Delta x$$

$$|\xi|^2 = \cos^2 k \Delta x + \frac{\nu^2 \Delta t^2}{\Delta x^2} \sin^2 k \Delta x$$

Stable if $\frac{\nu \Delta t}{\Delta x} < 1$ (Courant-Friedrichs-Lewy) or Courant condition

Better spatial resolution demands smaller time-step.
- Intuition: Propagation velocity

\[ \Delta x \approx v \Delta t \]

Computer in formation

\[ \Delta x < v \Delta t \]

Physical information

\[ \Delta x < v \Delta t \]

Physically important
Numerically invisible

Computer knows enough physical info to move forward

Algorithm can't know enough to propagate

- Use higher order spatial gradients

\[ \frac{\partial^4 u}{\partial x^4} = \ldots \frac{\partial^4 u}{\partial x^4} \]

have much worse time-step scaling

\[ \Delta t \approx \frac{\Delta x^4}{d} \]

- Implicit methods
Shock formation and upwind schemes

Burger’s equation (inviscid)
\[ \frac{\partial u}{\partial t} = -u \frac{\partial u}{\partial x} \]

Like wave equation \[ \frac{\partial u}{\partial t} = -v \frac{\partial u}{\partial x} \] except velocity depends on field \( u \) \[ v = u \]

Shock formation

Upwind Scheme
\[ \frac{u^{n+1}_i - u^n_i}{\Delta t} = -v^n \left\{ \left| \frac{u^n_{i+1} - u^n_{i}}{\Delta x} \right| - \left| \frac{u^n_{i} - u^n_{i-1}}{\Delta x} \right| \right\} \]

Mention: ENO, WENO

Artificial viscosity

Wave Solutions

Velocity Viscosity
Example PDE: Cardiac Dynamics

FitzHugh–Nagumo model for heart tissue

\[ \frac{\partial V}{\partial t} = \Delta^2 V + \frac{1}{\varepsilon} (V - V^3/3 - W) \]

\[ \frac{\partial W}{\partial t} = \varepsilon (V - \gamma W + \rho) \]

\[ \Delta^2 V = \sum\frac{\partial^2 V}{\partial x^2} + \sum\frac{\partial^2 V}{\partial y^2} \]

\[ \Delta^2 V = \frac{V_{i+1,j} - 2V_{i,j} + V_{i-1,j}}{\Delta x^2} + \frac{V_{i,j+1} - 2V_{i,j} + V_{i,j-1}}{\Delta y^2} \]

\[ \Delta^2 V \approx \frac{1}{\Delta x^2} \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \]

Better (same order, but more spherical)

\[ \Delta^2 V \approx \frac{1}{\Delta x^2} \begin{pmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{pmatrix} \]
Ghost Arrays

Edges: \( \frac{\partial V}{\partial x} - \frac{\partial W}{\partial x} = 0 \)

Zero normal derivatives (Neumann boundary condition)

\[ V(-1_{NN}) = V_{01} \quad V_{NN} = V_{N-1N} \]

Periodic

\[ V_{-1N} = V_{N-1N} \quad V_{N1} = V_{01} \]

Muntzki Parallel

Euler step
*** Big Projects: Design Patterns

Typically, sometimes even simple numerical tasks become enormous endeavors during a PhD thesis.

Molecular Dynamics using Verlet:

\[ \vec{\dot{x}} = -\nabla V / m \]

\[ \vec{\dot{v}} = \vec{\dot{x}} \cdot \Delta t / m \]

Inner loop

Immediate implications:

⇒ Store \( \vec{x} \) and \( \vec{v} \) separately, as 3xN vectors.

⇒ No 'Atom' class, with \( \text{Atom. } \vec{x}, \text{atom. } \vec{v}, \text{atom. } m, \text{atom. color, atom. radius} \)

(inefficient to load in, jump over extra info together to calculate \( F(R) \), nothing much else.)

List of Atoms

\( \text{List of Atoms} \) atoms

\( \text{List of Atoms} \) atoms, \( x = 3xN \) away...
[Show 1993 paper melting copper clusters]

16727 atoms, 8000 processors (?)

Sometimes want to write out atom positions each N time steps

need to calculate which are liquid-like, which crystalline

Specific Heat: Kinetic Energy $K$, potential energy $E$
- calculate at each time step
- store $\langle K \rangle, \langle E \rangle$
- use to estimate specific heat, temperature $T = 2 \langle K \rangle / N k_B$
  (equipartition, $K = \frac{1}{2} N k_B T$

Specific heat

Stress, pressure, 000

$X^{+} = \dot{X} + \Delta t$

$\ddot{X} = \frac{F(\dot{X})}{m} \cdot \Delta t / m$

\begin{cases}
  M_{t+1} = 1 \\
  0 \\
\end{cases}

if $m \propto M = 0$:

$K = \text{Kinetic Energy} = 0.5 * m * \dot{X} \cdot \dot{X}$

$E = \text{Energy}(\dot{X})$

$T = 2 * K / (N * k_B)$

WriteToFile(..., 'ligSol', $X$)

CalculateLiqSol

ligSol = Liquid Solid
Atoms($X$)
Filename = "ligSol.m"
Melting a Copper Cluster

Negative Specific Heat: The Danes Melt a Copper Cluster. (65)

While in Denmark on sabbatical, my hosts (Karsten Jacobsen, Jens Nørskov and their group) were simulating a 17,000 atom copper cluster using effective medium theory on a massively parallel CM-2 computer. Starting from a crystal at low temperatures, they were adding heat and watching the temperature as the cluster melted.

For an infinite system, the temperature should stay fixed while the crystalline core of the cluster slowly melts. Because we had a finite droplet, this jump in energy at fixed temperature becomes a curve, which in a small region bends backward: a negative specific heat! As energy is added, the crystalline nucleus melts; the smaller nucleus has a larger curvature, lowering its melting temperature. Putting in data for real copper into this simple theory led to a beautiful correspondence with the simulation.

From a formal statistical mechanics viewpoint, this temperature drop scales as $N^{1/4}$, where $N$ is the number of atoms in the droplet; for large $N$ ours is the largest finite--size effect known in first--order transitions. A macroscopic cluster with a mole of copper should have over a millidegree drop in temperature as heat is added.

More Information

- Equilibrium Crystal Shapes
KMCMD: Picoseconds to Seconds

Hybridizing Kinetic Monte Carlo with Molecular Dynamics

Joachim Jacobsen worked with Barbara Cooper and me to combine molecular dynamics calculations and Kinetic Monte Carlo calculations into one hybrid method, which we call KMC-MD, to study how surfaces grow when they are bombarded with high-velocity atoms.

What are these methods, and why did we need to hybridize them? The need comes from the vast differences in time scales...

The collision of an atom with the surface happens very quickly: our Molecular Dynamics simulation (which solves for the motion of the atoms near the collision) ends within five picoseconds ($5 \times 10^{-12}$ seconds, or $1/200,000,000,000$ second). The surfaces are grown, however, very slowly: typically one layer of atoms is added each second. If we ran our Molecular Dynamics code until the next collision, we’d wait about a year before the second atom hits!

**Molecular Dynamics: solving for the collision**

On the right, you see a picture of the region of atoms we use to solve for this very fast collision. It’s a tiny portion of our whole surface: by slicing out the important region we save some computer time. The different colors of atoms are treated differently with our model: the black atoms are held fixed in place, the grey atoms are put in a kind of computer glue (Langevin dynamics) that soaks up the energy from the collision so it doesn’t bounce back to the surface, and the white atoms are normal.

Here you can see some of the complicated collisions that occur. On the left you see the trajectory of the atoms as atom number one hits from above. On the right you can see
Quantum Tunneling of a Large Object

Inside the atom, the weird effects of quantum mechanics rule. Electrons have no definite position or velocity; the results of experiments can only be expressed in terms of probabilities. One of the weirdest effects is quantum tunneling: a particle can escape a trap even when it does not have the energy to do so. Some forms of radioactive decay are due to the quantum tunneling of alpha particles out of the nucleus. Scanning tunneling microscopes rely on electrons tunneling from the scanning tip to the surface being imaged.

For larger objects (atoms and collections of atoms), quantum fluctuations are unimportant and a simpler theory, classical mechanics, is sufficient. For example, tunneling of atoms is unusual. Because they are so heavy and large, they tunnel through barriers only rarely or under special circumstances. There has nonetheless been growing experimental evidence over many years - involving quantum creep, dissipation, and low-temperature scaling behavior in nanodevices - that some kind of atomic tunneling is important in metals.

At root, the reason atoms don’t tunnel is that their tunneling barriers and distances are set by the much lighter electrons. The tunneling of a proton over a barrier one Rydberg high and one Bohr radius wide is suppressed by the exponential of $\sqrt{(2M_p R_y a_0^2)} = \sqrt{(M_p / m_e)} \approx 42.85$: a factor of $10^{-19}$. When atoms tunnel, one looks for a special reason.

We and colleagues at CAMP at the Technical University of Denmark calculated the quantum tunneling rates of a crystalline defect - a dislocation kink - in copper, using a classical interatomic potential. (Usually one turns to quantum mechanics to get the basic parameters for a classical calculation: here we use classical mechanics to get the parameters for a quantum calculation.) The dislocation is split into two partial dislocations (green in the figure above), separated by a stacking fault (yellow). The kinks on each partial dislocation are spread out among hundreds of atoms. Perversely, it is because the kink is so large that it
Replace with Subject-Observer design pattern

- List of Atoms. Attach (Energy Observer)
- Atoms. Attach (Liquid Solid Visualizer)

\[
\begin{align*}
\text{atoms}, \mathbf{\vec{x}}_t &= \mathbf{\vec{v}} \cdot \Delta t \\
\text{atoms}, \mathbf{\vec{v}}_{t+1} &= \mathbf{\vec{F}}(\mathbf{\vec{x}}_t) \cdot \Delta t / m \\
\text{atoms. Notify()} 
\end{align*}
\]

- For all o in observers
- o. Update(self)

\[\text{[Atoms]}\]

- Vastly more flexible, clean
- Efficient even when updating each spin flip for Ising model of hysteresis

Collaboration between objects
- Define interface: observers must have update
- Subjects should have attach

Design to be flexible for those tasks that are likely to change

[Show transparency of class interactions]
[Show HMCMD: fixed, damped, free atoms, colliding ion]

AtomsInitializer

- Build three subclusters from definition of lattice, radii

- Three atom lists? Or one atom list with three sublists?

ClusterInitializer (In ( ), )...
Hexagonal ClusterInitializer
Composite ClusterInitializer

[Show Kinki Tunneling: dislocation line, partials, kink?]

Transformer

- cut-open simulation
- deform atoms, remove overlaps
- relax...

[Show transparency: collaboration between objects]
Subject - Observer design (aka Publish - Subscribe)

When one object changes state [Subject], all dependents [Observers] are notified and updated.

Subject
- Attach (Observer)
- Detach (Observer)
- Notify -> (for all o in observers: o.Update(self))

Observer
- Update (Subject)
Potential

do double Potential Energy (atoms)

Get Force (atoms, Force)

Example of the Factory Method = Virtual Constructor

- Define interface to objects
  - makes it easy to swap one for another
- Commonly implemented

(NR - various integration schemes swappable)
Neighbor Locator

- Finds all neighbors of each atom within potential cutoff radius Rcut

Some implement
Get Neighbors (atom Number)
→ returns
[neighbors, \tilde{r}_{ij}, \tilde{r}_c - \tilde{r}_{ij}]

Note: \tilde{r}_{ij} calculated including effects of periodic boundary conditions

Others implement
Get Half Neighbors (atom#)
- Returns those abrs w j > i.

Good for pair potentials (don't do each bond twice!)
avoid otherwise.

How to change one to the other to allow interchange of potentials? Adapter design pattern.
Adapter [aka Wrapper]

Convert the interface of one object into another that clients expect, allowing classes to work together that before were incompatible.

Adapted Neighbor Locator
- Member object Old Neighbor Locator
  Get Half Neighbors (atoms)
    Full Nbrs = Old Neighbor Locator, Get Neighbors()
    [Prune...]
    [Return...]

480.680 0
Molecular Dynamics Class Diagram

Below is a sketch of our molecular dynamics classes and their associations. It does not address how the objects communicate.

A class which refers to another points to it with an arrow. (This does mean ListOfAtoms has pointers to a BoundaryCondition, a NeighborLocator, and a Constraint.) Derived classes point to their bases with hollow arrows. The numbers on either end of lines connecting classes indicate multiplicity. For instance, a ListOfAtoms has only one BoundaryCondition, but a BoundaryCondition can be associated with more than one list of atoms.
Large Projects and Design Patterns:
- Planning for Change

1. Verlet inner loop:
   - Storage: List of Positions
   - List of Velocities
   - not Atom structure

2. Observers:
   - write to file N steps
   - calculate energy, fluctuations \rightarrow specific heat
   - calculate stress, pressure, ...

   \rightarrow Subject-Observer paradigm

3. Initializers:
   - Many types, interchangeable pattern

4. Movers / Transformers

\# Design Patterns
- collaborations among computational objects

\# Boundary Conditions, \( N^2 \), Multicore
- Neighbor locator
- DM Array Composite design pattern
- Observer for atoms