

Numerical methods for physicists

1

Logistics

- Course Web page:
www.pha.jhu.edu/~neufeld/numerical
– login: Student; password: Raphson
- Lecture schedule
Th – F 9 – 10:20 a.m. in Bloomberg 278

2

Contact information

Instructor: David Neufeld
509 Bloomberg
410-516-8582
neufeld@pha.jhu.edu
Office hours: Thursdays 1:30 – 2:30 PM, or by appointment

TA: Divya Singh
346 Bloomberg
410-516-6694 / 301-975-5660
divyas@pha.jhu.edu
Office Hours: Tuesdays 2:30 – 3:30 PM, or by appointment

3

Assignments

There will be problem sets roughly every week.

Solutions will be posted on the course website soon after they are turned in.

Many (most) of the problems will be made much easier by incorporating routines from *Numerical Recipes*; a complete set [in Fortran, ANSI C, KR C (original C dialect), and C++] can be found at eta.usr/site/numerical-recipes.

Grades will be based entirely on the problem sets; there will be no exams.

4

Ground rules for Problem Sets

- Unlimited discussion with fellow students is allowed
- But, your actual writing of the solutions, and your writing of the supporting computer code, must be done entirely independently

5

Late homework policy

- Homeworks will typically be due, in class, on a Friday.
- Late homeworks will be accepted – for reduced credit – up to one week after the due date.
- Homeworks more than 7 days late will not be accepted.
- Exceptions to all the above: extensions granted – IN ADVANCE – for medical reasons or because of research-related or emergency travel

6

Ethics

- The strength of the university depends on academic and personal integrity. In this course, you must be honest and truthful. Ethical violations include cheating on exams, plagiarism, reuse of assignments, improper use of the Internet and electronic devices, unauthorized collaboration, alteration of graded assignments, forgery and falsification, lying, facilitating academic dishonesty, and unfair competition.
- Report any violations you witness to the instructor. You may consult the associate dean of students and/or the chairman of the Ethics Board beforehand. See the guide on "Academic Ethics for Undergraduates".

7

Course overview

- **Goal:** to learn how to apply numerical methods to a variety of physical problems
- **Background:**
 - Many canned routines are available (e.g. Mathematica), and you can try to use them without understanding how they work

BUT.....

8

Course overview

- You really need to know the strengths and limitations of the various methods in order to make an informed choice
- Many numerical problems have pathologies: you need to learn how to recognize them
- You may need to improve the canned routines (better precision, or faster execution)
- Some methods are not widely available as canned routines (e.g. Monte Carlo simulations)
- The title "Numerical Recipes" is perhaps poorly-chosen

9

Syllabus

Preliminaries	Chap 1
Linear Equations	Chap 2
Integration	Chap 4
Random Numbers	Chap 7
Root Finding	Chap 9
Minimization	Chap10
Eigenvalues/Eigenvectors	Chap 11
Fourier Transforms	Chap 12
Ordinary Diff. Eqs	Chap 16
Partial Diff. Eqs	Chap 19
Statistical Methods	Chaps 14+15 + notes
Neural Networks	notes

10

Lecture 1: Preliminaries (see "Recipes" Chapter 1)

Analytic techniques have been discussed in the previous semester of this course

- **Advantages of analytic methods:**
 - Behavior of a physical system described by finite set of mathematical expressions
 - Limiting behaviors easily studied: important for making a "sanity check" and in understanding the underlying physics
 - Expressions often make behavior easy to visualize

11

Numerical vs. analytic methods

- **Disadvantages of analytic methods**
 - Many problems (most real life problems) are simply not soluble by analytical methods, especially
 - for complex systems involving several interacting physical processes
 - "God does not care about our mathematical difficulties. He integrates empirically."* – A. Einstein
 - when comparison is made with real observational data (with instrumental effects, selection effects)

12

Numerical vs. analytic methods

- **Disadvantages of analytic methods**

- Stochastic processes are not well described by analytic methods
 - Average properties can often be derived, but not observables that involve correlations between degrees of freedom

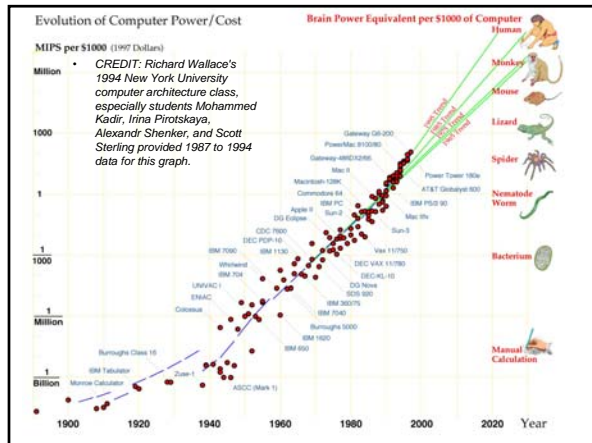
→ Has led to the development of a great many techniques (“Monte Carlo” simulations)

13

The microprocessor revolution

- Numerical methods have been revolutionized by advances in microprocessors
- The following plot appeared in
 - ROBOT: Mere Machine to Transcendent: by Hans Moravec (Oxford University Press, 1998)

14



Numerical vs. analytic methods

- **Advantages of numerical methods**

- Can solve real problems in a “straightforward manner”
- Very natural for stochastic processes

16

Numerical vs. analytic methods

- **Disadvantages of numerical methods**

- Solutions can be difficult to interpret: must often vary parameters to get a full understanding
- Limiting cases not always easy .. beware of $0 \div 0$
- Visualization must be built-in: no simple expressions to view
- Checking of results is non-trivial – bugs
- Pathological behavior: difficulties with precision, singularities, stability

17

Programming languages

- Numerical solutions always involve writing computer programs
- Languages:
 - Fortran: simple, old-fashioned, good at numerical computation, good handling of vectors and matrices
 - C: more powerful for strings/non-numerical data, weaker than Fortran for numerical calculations, more powerful control structures (relative to Fortran-77 but not Fortran-90)
 - C++: object oriented approach, poorer numerical performance relative to Fortran/C, industrial standard

18

Programming style

- Keep it simple, stupid (KISS)
- Comment extensively
 - eases debugging
 - eases reuse of the code (by others, and by you when you come back to it years later!)
- Use modules (subroutines and functions)
- Use “structured programming” approach
- Name variables appropriately and uniformly
- Avoid explicit branches (i.e. use if..then..else instead of if..goto)

19

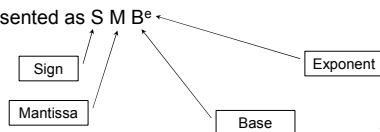
Considerations beyond the scope of this course (and the book)

- In real-life computing, for many problems, a key factor is writing computer code that runs rapidly, by
 - facilitating parallel computing (“vectorizing”)
 - controlling storage (to keep the most accessed variables in the fastest memory)
- This is really a computer science problem (that is highly hardware-dependent)

20

Errors and precision

- **Storage of numbers in computers**
 - Integers: exact, discrete
 - Used for loop indices, array indices
 - Floating point (“real”): approximate, continuous
 - Used for actual numerical calculations
 - Represented as $S M B^e$



21

Floating point numbers

- **Single precision**
 - 32 bits assigned to each number: 8 bits to e, 24 bits to M and S
 - Range: $2^{-127} < |R| < 2^{128}$ ($10^{-38} < |R| < 10^{38}$)
 - Precision: 1 part in 2^{23} (few parts in 10^8)
- **Double precision**
 - 64 bits assigned in various ways, leading to (for example)
 - Range: $10^{-76} < |R| < 10^{76}$
 - Precision \sim 1 part in 10^{16}

22

Floating point numbers

- **Quad precision**
 - 128 bits assigned
 - (available on some compilers)
- **Complex**
 - Real and imaginary parts

23

Roundoff error

- The finite precision of floating point numbers leads to roundoff error, a serious issue that must be minimized.
- Most problematic when numbers of very different size are added or subtracted
- In single precision,
 - $1.0 + 0.001 = 1.0010000$...BUT...
 - $1.0 + 0.000000001 = 1.0000000$

24

Roundoff error

- Effects can often be minimized by paying careful attention to the order of operations:

e.g. suppose we are interested in the sum of N numbers of widely-varying size, a_n . We want to add them together in increasing order of size.

$$\underbrace{1.0 + 1.0 + 1.0 + 1.0 + \dots + 1.0}_{10^9 \text{ entries}} + 1.0 \times 10^9 = 2.0 \times 10^9$$

BUT

$$1.0 \times 10^9 + \underbrace{1.0 + 1.0 + 1.0 + 1.0 + \dots + 1.0}_{10^9 \text{ entries}} = ???$$

25

Roundoff error

- Even so, you often encounter problems if you add $\sim 10^7$ single precision numbers
- Solution: go to double precision
- Downside: double precision additions (multiplications) may take 2 (4) times as long (depending upon the CPU)

26

Underflow/overflow errors

- Overflow errors: $10^{30} \times 10^{30} = \text{NaN}$
 - program will usually crash
- Underflow errors: $10^{-30} \times 10^{-30} = 0$
 - program will continue (but a good compiler should warn you what happened)
- Solutions
 - rescale, or use a logarithmic scale
 - go to double precision
 - pay attention to the order of operations

e.g. $(6.7 \times 10^{-27} \times 6.7 \times 10^{-27}) / (9.1 \times 10^{-28}) = 0$
 whereas $(6.6 \times 10^{-27} / 9.1 \times 10^{-28}) \times 6.6 \times 10^{-27} = 4.7 \dots \times 10^{-287}$

"Not a Number"

Truncation errors versus roundoff errors

- Truncation errors are the errors inherent in the numerical method adopted, even on a perfect machine (with infinite precision)

Example: numerical integration using the trapezoid rule

$$\int_{x_1}^{x_2} f(x) dx = \frac{1}{2} h \{f(x_1) + f(x_2)\} + O(h^3 f')$$

Truncation error

Stability

- Under certain circumstances, the accumulation of roundoff errors can lead to a rapidly (e.g. exponentially-growing) overall error
- This is called instability
- Example: consider the "Golden Mean", $\phi \equiv \frac{1}{2}(\sqrt{5} - 1) = 0.618\dots$, which obeys the recurrence relation $\phi^{n+1} = \phi^{n-1} - \phi^n$

29

Stability

- So couldn't you use the recurrence relation to obtain powers of ϕ by means of subtraction (very fast) rather than multiplication?

$$\begin{aligned} \text{e.g. } \phi^2 &= 1 - \phi \\ \phi^3 &= \phi - \phi^2 \\ \phi^4 &= \phi^2 - \phi^3 \\ &\dots \end{aligned}$$

30

Stability

- Problem: second solution to recurrence relation is $-\frac{1}{2}(\sqrt{5} + 1)$
 - This solution has absolute value > 1 , so powers of it grow exponentially
 - Any small component of this solution, introduced by roundoff error, grows exponentially (whereas the solution we want shrinks exponentially)
 - Pretty soon, the results of applying the recurrence relation are completely wrong

31

Summary

- Numerical methods are powerful
- Numerical methods are delicate
 - It's very easy to get the wrong answer!
 - ➔ much of art of applying numerical methods to real world problems is figuring out how to check whether your program is giving the right answer

32

Lecture 2: Systems of Linear Equations (see "Recipes" Chapter 2)

- Systems of linear equations arise in many contexts. As we'll see later, their solution is a fundamental tool in numerical methods.
- Consider a system of M linear equations in N unknowns

$$\begin{array}{rcl} a_{11}x_1 + a_{12}x_2 + \dots + a_{1N}x_N & = & b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2N}x_N & = & b_2 \\ a_{31}x_1 + a_{32}x_2 + \dots + a_{3N}x_N & = & b_3 \\ \vdots & & \vdots \\ a_{M1}x_1 + a_{M2}x_2 + \dots + a_{MN}x_N & = & b_M \end{array} \quad \begin{array}{l} a_{ij}, b_i \text{ are known constants} \\ x_i \text{ is unknown} \end{array}$$

33

Matrix notation

- In vector notation: $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$
- For a square matrix (N equations and N unknowns), the behavior is critically affected by the determinant,

$$|\mathbf{A}| = \sum_{\epsilon_{i_1 i_2 i_3 \dots i_N}} \epsilon_{i_1 i_2 i_3 \dots i_N} a_{1i_1} a_{2i_2} a_{3i_3} \dots a_{Ni_N}$$

where the sum is computed with $i_1, i_2, i_3, \dots, i_N$ each taking all possible values over the range 1 to N, and

$\epsilon = -1$ if $i_1, i_2, i_3, \dots, i_N$ is an odd permutation of 1, 2, 3, ..., N (e.g. as in ϵ_{2134})

$\epsilon = +1$ if $i_1, i_2, i_3, \dots, i_N$ is an even permutation of 1, 2, 3, ..., N (e.g. as in ϵ_{2143})

$\epsilon = 0$ otherwise (i.e. any two indices are the same)

There are N^N terms in the sum, of which $N!$ are non-zero

34

Singular versus non-singular matrices

- If $M = N$, there is a unique solution for \mathbf{x} , namely $\mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}$, if and only if $|\mathbf{A}| \neq 0$.

Matrices with $|\mathbf{A}| \neq 0$ are said to be non-singular:

Matrices with $|\mathbf{A}| = 0$ are singular (and they have no inverse)

- Example: consider a diagonal square matrix,

$$\mathbf{A} = \begin{pmatrix} a_{11} & & & \\ & a_{22} & & \\ & & a_{33} & \\ & & & \dots \\ & & & & a_{NN} \end{pmatrix}$$

35

Singular versus non-singular matrices

$$\mathbf{A}^{-1} = \begin{pmatrix} 1/a_{11} & & & \\ & 1/a_{22} & & \\ & & 1/a_{33} & \\ & & & \dots \\ & & & & 1/a_{NN} \end{pmatrix} \quad \text{exists iff all the } a_{ii} \text{ are non-zero, so that a unique solution } \mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b} \text{ exists}$$

But if one or more a_{ii} are zero, $|\mathbf{A}| = \prod a_{ii} = 0$, the matrix is singular, and \mathbf{A}^{-1} does not exist

36

Singular versus non-singular matrices

- Matrices are singular if two or more rows are not linearly independent
 - We then have fewer than N linearly independent equations to determine N unknowns
- Numerically, roundoff errors introduce noise that can have two effects:
 - A non-singular problem can become approximately singular
 - the solution blows up (yielding huge, crazy solutions)
 - A stable, non-singular problem can produce the wrong answer
 - always need to check solution
 - For well conditioned (far from singular) problems, single precision works OK for $N < 30$ and double precision for $N < 300$

37

Non-square matrices ($N \neq M$)

• Two cases

$M < N$ – there is insufficient information to define a unique solution, but we can find the $(N - M)$ dimensional subspace that the N dimensional vector \mathbf{x} inhabits (singular value decomposition SVD, to be considered later).

$M > N$ – the solution is overconstrained (unless $M - N$ equations can be written as linear combinations of the remaining N). There is no exact solution, but there is one that minimizes the summed squared error

$$E = |\mathbf{A}\mathbf{x} - \mathbf{b}|^2 = (\mathbf{A}\mathbf{x} - \mathbf{b})^T \cdot (\mathbf{A}\mathbf{x} - \mathbf{b})$$

38

Minimization of the error when the solution is overconstrained

- The summed squared error, E, can be written $(\mathbf{A}\mathbf{x} - \mathbf{b})^T \cdot (\mathbf{A}\mathbf{x} - \mathbf{b}) = \mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{A} \mathbf{x} - \mathbf{x}^T \mathbf{A}^T \mathbf{b} + \mathbf{b}^T \mathbf{b}$

$1 \times N$ $N \times M$ $M \times N$ $N \times 1$ All terms are scalar

- Minimum error, E, occurs when

$$0 = \partial E / \partial x_j = (\mathbf{A}^T \mathbf{A})_{jj} x_j + (\mathbf{A}^T \mathbf{A})_{jj} x_j - (\mathbf{b}^T \mathbf{A})_j - (\mathbf{A}^T \mathbf{b})_j$$

$$= 2 (\mathbf{A}^T \mathbf{A})_{jj} x_j - 2 (\mathbf{A}^T \mathbf{b})_j \quad \mathbf{A}^T \mathbf{A} \text{ is a symmetric } N \times N \text{ matrix}$$

The solution is therefore obtained from equation

$$(\mathbf{A}^T \mathbf{A}) \mathbf{x} = \mathbf{A}^T \mathbf{b} \quad (N \text{ equations and } N \text{ unknowns})$$

and the corresponding value of E is $\mathbf{b}^T (\mathbf{b} - \mathbf{A} \cdot \mathbf{x})$

39

Solution of the N x N problem LU decomposition

- Write \mathbf{A} as a product of two triangular matrices (non unique: we discuss how to do this later)

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{U} = \begin{pmatrix} L_{11} & 0 & 0 & \dots & 0 \\ L_{21} & L_{22} & 0 & \dots & 0 \\ L_{31} & L_{32} & L_{33} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ L_{N1} & L_{N2} & \dots & L_{NN} & \dots \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & \dots & U_{1N} \\ 0 & U_{22} & \dots & U_{2N} \\ 0 & 0 & U_{33} & \dots & U_{3N} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & U_{NN} \end{pmatrix}$$

40

Once \mathbf{L} and \mathbf{U} are known, \mathbf{x} can easily be found

- The problem $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ can be written $\mathbf{L} \cdot \mathbf{y} = \mathbf{b}$, with $\mathbf{y} \equiv \mathbf{U} \cdot \mathbf{x}$

We solve first for \mathbf{y} by “forward-substitution”

$$L_{11} y_1 = b_1 \quad \rightarrow y_1 = L_{11}^{-1} b_1$$

$$L_{21} y_1 + L_{22} y_2 = b_2 \quad \rightarrow y_2 = L_{22}^{-1} [b_2 - L_{21} y_1]$$

...

$$L_{i1} y_1 + L_{i2} y_2 + \dots + L_{ii} y_i = b_i \quad \text{The unknowns ("not-yet-knowns") appear in red - only one in each eqn}$$

$$\rightarrow y_i = L_{ii}^{-1} [b_i - L_{i1} y_1 - L_{i2} y_2 - \dots - L_{i,i-1} y_{i-1}]$$

KEY POINT: Each component of \mathbf{y} can be written easily in terms of components already computed

41

Solution of the N x N problem LU decomposition

- Once we have \mathbf{y} , we solve $\mathbf{U} \cdot \mathbf{x} = \mathbf{y}$ just as easily by “backsubstitution” (essentially the same method, but starting with the bottom row: $U_{NN} x_N = y_N$)

Again, whenever we come to a particular component of \mathbf{x} , we get an equation that involves only components that have already been computed.

42

So how do we find the LU decomposition of matrix **A**?

A has N^2 elements, each of which can be written

$$A_{ij} = L_{i1}U_{1j} + L_{i2}U_{2j} + \dots + L_{ik}U_{jk}$$

Need only go up to $k = \min(i,j)$, since $L_{ik} = 0$ for $k > i$ and $U_{jk} = 0$ for $k > j$

L and U each have $\frac{1}{2}N(N+1)$ non-zero elements, for a total of $N^2 + N$ unknowns

But there are only N^2 constraints, so the problem is underdetermined \rightarrow we can choose N of the elements

43

So how do we find the LU decomposition of matrix **A**

Convention: choose $L_{ii} = 1$ for $i = 1, N$

Can now solve all the others in such a way that each element can be written in terms of elements already computed

$$\begin{aligned} A_{11} &= L_{11}U_{11} \rightarrow U_{11} = A_{11}/L_{11} = A_{11} \\ A_{1j} &= L_{11}U_{1j} \rightarrow U_{1j} = A_{1j}/L_{11} = A_{1j} \\ A_{i1} &= L_{i1}U_{11} \rightarrow L_{i1} = A_{i1}/U_{11} = A_{i1}/A_{11} \end{aligned}$$

44

So how do we find the LU decomposition of matrix **A**?

We continue

$$A_{2j} = L_{21}U_{1j} + L_{22}U_{2j} \rightarrow U_{2j} = A_{2j} - L_{21}U_{1j}$$

$$A_{i2} = L_{i1}U_{12} + L_{i2}U_{22} \rightarrow L_{i2} = U_{22}^{-1} (A_{i2} - U_{12}L_{i1})$$

$$\text{In general: } U_{ij} = A_{ij} - \sum_{k=1}^{i-1} L_{ik} U_{kj} \quad \text{go upward in } i$$

$$L_{ij} = U_{jj}^{-1} (A_{ij} - \sum_{k=1}^{j-1} L_{ik} U_{kj}) \quad \text{go upward in } j$$

45

How do we minimize roundoff error?

- Very important trick: reorder rows of a before starting so that column 1 elements decrease with increasing i
- This minimizes roundoff error in the early stages and keeps its growth to a minimum

46

Computational cost of solving linear system

- To find any U_{ij} requires $2(i-1)$ operations (1 addition plus 1 multiplication per term in the sum)

$$\text{Total computation of U requires } \sum_{j=1}^N \sum_{i=1}^j 2(i-1) \sim \frac{1}{3} N^3 \text{ operations}$$

- Similarly, to find any L_{ij} requires $2(j-1) + 1$ operations \rightarrow total computation of L also requires $\sim \frac{1}{3} N^3$ operations

Extra division

- The final substitutions require $\sim N^2$ operations (negligible additional amount for large N)

47

Summary of solution to **A.x = b** using LU decomposition

- 1) Find **L.U = A** ($\sim \frac{2}{3} N^3$ operations)
- 2) Solve **L.y = b** for **y** ($\sim \frac{1}{2} N^2$ operations)
- 3) Solve **U.x = y** for **x** ($\sim \frac{1}{2} N^2$ operations)

48

Iterative improvement

(correcting the solution for residual roundoff error)

- Let \mathbf{x} be the exact solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ and let $\mathbf{x} + \delta\mathbf{x}$ be the result of our numerical calculation
- When we actually substitute $\mathbf{x} + \delta\mathbf{x}$ into the original equation, we get something that isn't exactly \mathbf{b} ; call the result $\mathbf{b} + \delta\mathbf{b}$

$$\mathbf{A}(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \mathbf{A}\delta\mathbf{x} = \mathbf{b} + \delta\mathbf{b}$$

- Thus we can estimate the error $\delta\mathbf{x}$ by solving $\mathbf{A}\delta\mathbf{x} = \delta\mathbf{b}$. This can be done very quickly since we already have the LU decomposition of \mathbf{A} !

49

Matrix inversion and determinants

- Once we have LU decomposed \mathbf{A} , it is straightforward to obtain \mathbf{A}^{-1} , which is just the solution to

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$$

We can obtain this column by column by solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ with

$$\mathbf{b} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \text{ etc}$$

- The determinant of \mathbf{A} is simply $|\mathbf{A}| = |\mathbf{L}||\mathbf{U}| = (\prod L_{ii}) (\prod U_{ii}) = (\prod U_{ii})$

50

Lecture 3: Sparse matrices

(Numerical Recipes, § 2.7)

- The solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ can go much faster if the matrix is "sparse", i.e. has few non-zero elements.
- Example: consider the numerical solution to $d^2x/dt^2 + f(t)x = g(t)$

Driven 1-D harmonic oscillator with time-varying spring constant

51

Sparse matrices

- Let's represent x on a uniform grid of discrete times, $t_i = i \Delta t$, ...and define $x_i \equiv x(t_i)$, $f_i \equiv f(t_i)$, and $g_i \equiv g(t_i)$,
- The second derivative can be approximated $d^2x/dt^2 \sim (x_{i+1} - 2x_i + x_{i-1}) / \Delta t^2$

and the equation can be written as a linear system $\mathbf{A}\mathbf{x} = \mathbf{g}$

52

Sparse matrices

where \mathbf{A} only has elements on – or right next to – the diagonal (and is called a "tridiagonal" matrix)

$$\begin{pmatrix} \times & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & \times & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \times & \times \end{pmatrix}$$

The LU decomposition of \mathbf{A} (and the forward and backsubstitution steps) only involve $O(N)$ computations! 53

Singular systems

- When linear systems fail, many things can be going on

Consider the singular system

$$\begin{aligned} x_1 + x_2 &= b_1 \\ 2x_1 + 2x_2 &= b_2 \end{aligned}$$

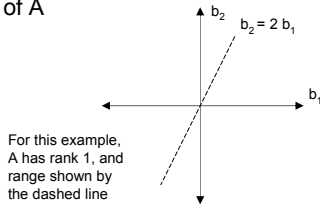
- In general, the equation has no solution
- But if $b_2 = 2b_1$, it has the (non-unique) solution

$$x_2 = b_1 - x_1$$

54

Singular systems

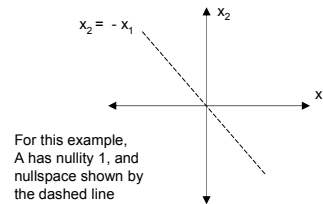
- The subspace within which \mathbf{b} yields a solution to $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ is called the range of \mathbf{A}
- The dimensionality of this range is called the rank of \mathbf{A}



55

Singular systems

- If \mathbf{A} is singular, and \mathbf{b} lies within the range of \mathbf{A} , the solution to $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ is always non-unique. In particular, the solution to $\mathbf{A} \cdot \mathbf{x} = \mathbf{0}$ is called the nullspace of \mathbf{A} and its dimensionality is called the nullity of \mathbf{A}



56

Singular systems

- The nullity and rank of an $N \times N$ matrix are related by rank + nullity = N
- If \mathbf{A} is non-singular:
 - rank = $N \rightarrow$ the range of \mathbf{A} covers the entire $N \times N$ space and there is a unique solution for any \mathbf{b}
 - nullity = 0 \rightarrow the nullspace of \mathbf{A} is a single point, $\mathbf{x} = \mathbf{0}$, which is the only solution to $\mathbf{A} \cdot \mathbf{x} = \mathbf{0}$

57

Singular systems

- The nullity and rank of an $N \times N$ matrix are related by rank + nullity = N
- If \mathbf{A} is singular:
 - rank < $N \rightarrow$ the range of \mathbf{A} covers only a subspace within the $N \times N$ space. There is only a solution to $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ when \mathbf{b} lies within this subspace (and then the solution is non-unique)
 - nullity > 0 \rightarrow the nullspace of \mathbf{A} consists of more than a single point (i.e a line, a plane, a 3-space....)

58

Singular value decomposition (Numerical Recipes, §2.7)

- SVD (Singular value decomposition) is a powerful technique for diagnosing and solving singular systems: essentially, it tells you the range and nullspace of \mathbf{A}
- It can also be used to understand (and sometimes fix) nearly-singular problems

59

Singular value decomposition

- The singular value decomposition of \mathbf{A} is written

$$\mathbf{A} = \mathbf{U} \mathbf{W} \mathbf{V}^T$$

$[\text{M} \times \text{N}] \quad [\text{M} \times \text{N}] \quad [\text{N} \times \text{N}] \quad [\text{N} \times \text{N}]$

where \mathbf{W} is diagonal and \mathbf{U} and \mathbf{V} are orthonormal
i.e. $\mathbf{U}^T \mathbf{U} = \mathbf{V}^T \mathbf{V} = \mathbf{I}$

This can always be done, and the decomposition is nearly unique (except for some transpositions)

- important theorem that we will not prove
- we will also not discuss the computational method for obtaining the SVD; the algorithm used in Numerical Recipes is very robust

60

Singular value decomposition

- The matrix W is diagonal

$$\begin{pmatrix} w_1 & & & & \\ & w_2 & & & \\ & & w_3 & & \\ & & & \dots & \\ & & & & w_{N-1} \\ & & & & & w_N \end{pmatrix} = \text{diag}(w_i)$$

The w_i are called the singular values of matrix A .
All the w_i are non-negative: if $M < N$, the last $N - M$ singular values are zero

61

Singular value decomposition of a square matrix

- The SVD is $A = U W V^T$, with U , V^T and W all square $N \times N$ matrices
- The inverse of A (if it exists) is therefore

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

with $W^{-1} = \text{diag}(1/w_i)$

Clearly, A^{-1} exists iff W^{-1} exists, and W^{-1} exists iff all the w_i are non-zero

62

Singular value decomposition of a singular matrix

- When A is singular, the SVD still exists, $A = U W V^T$ but the diagonal matrix W has one or more non-zero diagonal elements, w_i .
 - First define z_k as the column vector with all elements zero except for the k th, which is equal to 1
 - Suppose the k th singular value is zero, $w_k = 0$. Then $W z_k = 0$
Thus $A x = 0$ whenever $V^T x = z_k$
This occurs if $x = V z_k$
- But $V z_k$ is simply the k th column of $V \rightarrow$ if $w_k = 0$, the k th column of V lies in the nullspace of A . This defines the nullspace of A

63

2006: Lecture 4 started here

Singular value decomposition of a singular matrix

- What about the matrix U
- The vector $A \cdot x = U W V^T x$ is always a linear combination of the columns of U
 $\sum q_k u_k$
where $u_k = U \cdot z_k$ is the k th column of U and q_k is the k th element of the vector $W V^T x$
If w_k is zero, then q_k is zero. Thus $A x$ can only be a linear combination of those u_k for which w_k is non-zero: this defines the range of A

64

Singular value decomposition: significance of U and V

- Suppose the number of elements of W that are zero is $p > 0$ (i.e. A is singular)
- Those columns U that correspond to non-zero elements of W form an orthonormal basis for the range of A
 - the rank of A is $N - p$
- And those columns V that correspond to zero elements of W form an orthonormal basis for the nullspace of A
 - the nullity of A is p

65

Solution to $A x = b$

- If we can determine any particular solution to $A x = b$, call it x_0 , we know that the general solution is

$$x = x_0 + \sum r_k v_k,$$

where v_k is the k th row of V and where the sum is taken over all k for which $w_k = 0$

The r_k are coefficients that can take any value

66

Finding a single solution to $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ when \mathbf{A} is singular

- Consider the following matrix $\mathbf{P} = \mathbf{V} \mathbf{Q} \mathbf{U}^T$ where \mathbf{Q} is the diagonal matrix defined such that

$$Q_{ii} = 1/w_i \quad \text{whenever } w_i \neq 0$$

$$Q_{ii} = 0 \quad \text{when } w_i = 0$$
 This is called the pseudo inverse of \mathbf{A} , for reasons that will become clear. Suppose first that \mathbf{A} is non-singular, then all the w_i are non-zero and \mathbf{Q} is simply \mathbf{W}^{-1} . In this case,

$$\mathbf{P} = \mathbf{V} \mathbf{Q} \mathbf{U}^T = \mathbf{V} \mathbf{W}^{-1} \mathbf{U}^T = \mathbf{A}^{-1}$$
 and the (unique) solution is $\mathbf{x} = \mathbf{P} \mathbf{b}$

67

Amazing result

- Even if \mathbf{A} is singular, $\mathbf{x}_0 = \mathbf{P} \mathbf{b}$ is a solution (now non-unique, of course) to $\mathbf{A} \mathbf{x}_0 = \mathbf{b}$
- Let's check:

$$\mathbf{A} \mathbf{x}_0 = \mathbf{A} \mathbf{P} \mathbf{b} = (\mathbf{U} \mathbf{W} \mathbf{V}^T) (\mathbf{V} \mathbf{Q} \mathbf{U}^T) \mathbf{b}$$

$$= \mathbf{U} \mathbf{W} \mathbf{Q} \mathbf{U}^T \mathbf{b}$$
 The product $\mathbf{W} \mathbf{Q}$ is the *identity matrix*, except the diagonal components $(\mathbf{W} \mathbf{Q})_{kk}$ set equal to zero whenever $w_k = 0$

68

Amazing result

But these particular components don't matter if \mathbf{b} lies in the range of \mathbf{A}

Why? Because if $w_k = 0$, then the k th column of \mathbf{U} (which we are calling \mathbf{u}_k) lies outside the range of \mathbf{A} . Thus, if \mathbf{b} lies *within* the range of \mathbf{A} , then $\mathbf{u}_k \cdot \mathbf{b} = 0$ (because \mathbf{U} is orthonormal). The k th component of the vector $\mathbf{U}^T \mathbf{b}$ is therefore zero, and the k th component of $\mathbf{W} \mathbf{Q}$ doesn't matter. Thus we could replace all the zero components of $\mathbf{W} \mathbf{Q}$ by 1's, and we conclude that

$$\mathbf{A} \mathbf{x}_0 = \mathbf{U} \mathbf{W} \mathbf{Q} \mathbf{U}^T \mathbf{b} = \mathbf{U} \mathbf{U}^T \mathbf{b} = \mathbf{b}$$

as advertised

69

2007: Lecture 4 started here

Even more amazing result

- $\mathbf{x}_0 = \mathbf{P} \mathbf{b}$ is the *smallest* solution to $\mathbf{A} \mathbf{x} = \mathbf{b}$ (i.e. the one that minimizes the length $|\mathbf{x}_0|$)
- Consider another solution, $(\mathbf{x}_0 + \mathbf{x}')$, where \mathbf{x}' , of course, must lie in the nullspace of \mathbf{A}
- But, $\mathbf{x}_0 = \mathbf{V} \mathbf{Q} \mathbf{U}^T \mathbf{b}$ must be orthogonal to the nullspace:
 - \mathbf{x}_0 is a linear combination of those columns of \mathbf{V} for which w_k is non-zero (since $Q_{kk} = 0$ when $w_k = 0$)
 - \mathbf{x}' is a linear combination of those columns of \mathbf{V} for which w_k is zero
- Hence $|\mathbf{x}_0 + \mathbf{x}'|$ is always larger than $|\mathbf{x}_0|$
- $\rightarrow \mathbf{x}_0 = \mathbf{P} \mathbf{b}$ is the *shortest* vector that satisfies $\mathbf{A} \mathbf{x} = \mathbf{b}$

70

Least squares solution when \mathbf{b} lies outside the range of \mathbf{A}

- This is related to the case of the $M \times N$ matrix considered in Lecture 1 (for the case $M > N$)

The solution that comes closest to satisfying $\mathbf{A} \mathbf{x} = \mathbf{b}$ (i.e. which minimizes $|\mathbf{A} \mathbf{x} - \mathbf{b}|$) is again $\mathbf{x} = \mathbf{P} \mathbf{b}$

71

Nearly singular matrices

- Even if \mathbf{A} is non-singular, numerical problems can arise as a result of roundoff error when solving $\mathbf{A} \mathbf{x} = \mathbf{b}$, particularly when the matrix \mathbf{A} is "nearly" singular.
- Here, none of singular values is actually zero, but some are very small relative to others
- Define condition number = largest w_i / smallest w_i . When the condition number is very large, LU decomposition fails

72

Nearly singular matrices

- SVD can often help here. We obtain the SVD of the nearly singular matrix, then zero out the small w_i values, then find the least squares solution to $\mathbf{A} \mathbf{x} = \mathbf{b}$ (i.e. write $\mathbf{x} = \mathbf{P} \mathbf{b}$)
- The error caused by replacing the smallest w_i values with zero is often not as bad as the roundoff errors that result when the condition number is very large \rightarrow it sometimes pays to deliberately make a nearly singular matrix singular and then use SVD rather than LU decomposition

73

Numerical integration

(Numerical recipes, Chapter 4)

- Analytical differentiation is easy, but analytical integration is "hard": more of an art than a science
 - Led to the development of numerical techniques, even back in the 18th Century
 - Even simple classical methods are still useful

74

Numerical integration

- Numerical integration is always equivalent to the solution of a differential equation:
If $y = \int f(x) dx$ then $dy/dx = f(x)$

There are several modern algorithms for solving differential equations (that are particularly useful when $f(x)$ is not smooth)

These involve adaptive step size and will be discussed later

75

Basic integration techniques

- These harken back the Riemann definition of the integral

$$\int_a^b f(x) dx = \lim_{\substack{\Delta x \rightarrow 0 \\ N \rightarrow \infty}} \sum_{i=1}^N \Delta x_i f(x_i)$$

- First consider methods that use a constant step size ($\Delta x_i = h$ for all i)

76

Trapezoidal rule

- Break integral into $N - 1$ equally spaced intervals

$$I = h \left(\frac{1}{2}f_1 + f_2 + f_3 + \dots + f_{N-1} + \frac{1}{2}f_N \right) + h^2 (f_1' - f_N') / 12 + \dots$$

with stepsize $h = (x_b - x_a)/(N - 1)$

- Error is proportional to $h^2 \propto 1/N^2$
 - Need to cover regions where f'' is large with high density of points

77

Simpson's rule

- Slightly better performer for fairly *smooth* functions

- Again, break integral into $N - 1$ equally spaced intervals, but now use

$$I = h \left(f_1 + 4f_2 + 2f_3 + 4f_4 + 2f_5 + \dots + 2f_{N-2} + 4f_{N-1} + f_N \right) / 3 + h^4 (f_1'' - f_N'') / 180 + \dots$$

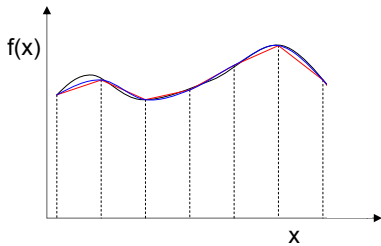
with N odd

- Now error is proportional to f''''/N^4

78

Geometric interpretation

- Black – integrand; red – trapezoid; blue – Simpson's



79

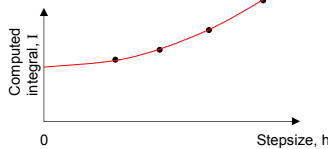
Control of truncation errors

- The choice of N must reflect the obvious tradeoff between
 - accuracy ($\propto 1/N^2$ or $1/N^4$) and
 - computational expense ($\propto N$)
- We keep increasing N until the answer stops changing (or changes only by an acceptably small amount)

80

Romberg integration

- We can even plot I vs h



- If the resultant function is smooth (good polynomial fit), we can extrapolate to $h = 0$

81

Complications

- Integrable singularity at the lower limit
 - Suppose the integrand has a singularity at $x = a$ (lower limit of integration), which diverges as $(x - a)^{-\gamma}$ with $0 \leq \gamma < 1$
 - Then use the change of variable $t = (x - a)^{1-\gamma}$, which yields

$$\int_a^b f(x) dx = \frac{1}{1-\gamma} \int_0^{(b-a)^{1-\gamma}} f(t^{1/(1-\gamma)} + a) dt$$

82

Complications

- Integrable singularity at the upper limit
 - Suppose the integrand has a singularity at $x = b$ (lower limit of integration), which diverges as $(b - x)^{-\gamma}$ with $0 \leq \gamma < 1$
 - Then use the change of variable $t = (b - x)^{1-\gamma}$, which yields

$$\int_a^b f(x) dx = \frac{1}{1-\gamma} \int_0^{(b-a)^{1-\gamma}} f(b - t^{1/(1-\gamma)}) dt$$

83

Complications

- Integrable singularity in the middle of the range at $x = c$, where $a < c < b$
- Simply write

$$\int_a^b f(x) dx = \int_a^c f(x) dx + \int_c^b f(x) dx$$

and use both formulae

84

Complications

- Bottom line: hard to devise a general numerical integrator
 - We always evaluate the function at a finite number of points, so we rarely detect an integrable singularity
 - Need to identify integrable singularities by analytical methods

85

Complications

- Upper or lower limit of $\pm\infty$
 - Use change of variables to map ∞ to a finite number (e.g. zero)

e.g. for
$$I = \int_1^{\infty} \frac{1}{x^2} g(x) dx$$

use $x = 1/t \rightarrow dx = - dt/t^2 = - x^2 dt$

to obtain
$$I = \int_0^1 g(1/t) dt$$

86

Irregular step sizes (see Numerical Recipes, §4.5, for details)

- So far, we have only used methods that compute the integral as

$$I = \sum w_i f(x_i)$$
 where w_i are the weights (e.g. $1/2, 1, 1, 1, \dots, 1, 1/2$ for the trapezoid rule) and the x_i are evenly-spaced
- For smooth functions, better accuracy can be achieved if we permit the x_i to be unevenly spaced
- For the right choices of w_i and x_i , we can arrange for the sum to be exact for $f(x)$ equal to any polynomial of degree $2N - 1$ or less! (Gaussian quadrature)

87

Multidimensional integrals (see Numerical Recipes, §4.6)

- So far, we've only considered 1-D integrals.
- Methods for multidimensional integrals:

1) for highly symmetric cases, can sometimes be rewritten as 1-D integral

Example: when integrating a spherically-symmetric 3-D function of x , y , and z , we can use

$$\iiint f(x,y,z) dx dy dz = \int f(r) 4\pi r^2 dr$$

88

Multidimensional integrals

2) Can write as N nested 1-D integrals

$$\int d^N x f(\mathbf{x}) = \int dx_1 \int dx_2 \dots \int dx_N f(x_1, x_2, \dots, x_N)$$

But..

- we need to know about function pathologies
- the boundary has to be simple (which it often isn't)
- we need N^a evaluations of the function

89

Multidimensional integrals

If the function is smooth, but the boundary is complex, then Monte Carlo methods can work quite well

→ Use multiple function evaluations at random locations within the boundary

Next topic: random number generation

90

Monte Carlo Integration

- Basic idea:

$$\int f(\mathbf{x}) dV = \bar{f} V$$

where \bar{f} is the average value of f within the volume V

91

Monte Carlo Integration

- How do we estimate \bar{f} ?
- Throw down points at random within the volume, and compute the average of f evaluated for those points
- If the boundary is complex, choose a hypercube that encloses it and discard points that lie outside the boundary

92

Monte Carlo Integration

If there are N points within the boundary:

Best estimate of \bar{f} is $\langle f \rangle \equiv (1/N) \sum f(x_i)$

Standard error on that estimate is $\sqrt{(\langle f^2 \rangle - \langle f \rangle^2)/N}$

93

Monte Carlo Integration

- So, Monte Carlo integration yields an estimate of the integral

$$I = (V/N) \sum f(x_i)$$

with a typical error $\delta I = V \sqrt{(\langle f^2 \rangle - \langle f \rangle^2)/N}$

Unfortunately, the $N^{-1/2}$ dependence is not very favorable

94

Monte Carlo Integration

- The fractional error $\delta I/I$ is of order $\sqrt{(\langle f^2 \rangle / \langle f \rangle^2 - 1)/N}$

The quantity $(\langle f^2 \rangle / \langle f \rangle^2 - 1)$ can be large if $f(\mathbf{x})$ is strongly peaked

For example, if f is very high in a small fraction of the volume, we are oversampling (i.e. wasting points) on regions where f is small.

95

Monte Carlo Integration

- Accuracy is improved if we can find a change of variables $dV' = g(\mathbf{x}) dV$ such that $g(\mathbf{x})$ has a strong dependence similar to that of $f(\mathbf{x})$

- Then $\int f(\mathbf{x}) dV = \int [f(\mathbf{x})/g(\mathbf{x})] dV'$

\uparrow \uparrow
 Strongly peaked Not strongly peaked

96

Lecture 5: Random numbers and Monte Carlo (Numerical Recipes, Chapter 7)

- Motivations for generating random numbers
 - To sample a function in a statistically controlled manner (i.e. for Monte Carlo integration)
 - To simulate truly random processes (noise, radioactive decay ...)

97

Random number generators

- Obvious problem: computers are deterministic.
 - Computer generated random numbers aren't truly random: given a known starting point, any sequence can be predicted
 - Best we can achieve is *pseudo-random* sequences with certain statistical properties

98

Hardware devices can produce truly random numbers



Rely on thermal noise to produce several $\times 10^5$ random numbers / second – cryptographic applications

99

Random number generators

- Typical generator: creates a sequence of numbers, r_k , distributed uniformly over the interval $[0, 1]$
 - Start with a seed value: each seed creates a different sequence of integers, I_k , over the interval $[0, M]$
 - These integers are used to create floating-point numbers, $r_k = \text{float}(I_k) / \text{float}(M)$ over the interval $[0, 1]$
 - Typical generator is portable: yields same result on any machine (important for error-checking)
 - The sequence will inevitably repeat with a period $P \leq M$

100

Random number generators

- We want to minimize correlation functions:

Define $A_L = \langle (r_k - 0.5)(r_{k+L} - 0.5) \rangle$
where $\langle \rangle$ denotes an average over all k

Truly random numbers would have $A_L = 0$ for all L

Clearly, real random number generators have $A_L > 0$ whenever L is a multiple of P

... and many have some correlations even when $L < P$

101

Random number generators

- Linear congruential generators:

$I_{j+1} = (a I_j + c) \text{ mod } M$
with M , a and c (carefully chosen) fixed integers
("Minimal Standard" routine: $M = 2^{31} - 1$, $a = 7^5$, $c = 0$)

Problems:

Clearly, the period P is at best M (and – for poorly chosen a and c – can be much less than M)

Correlations for small L :

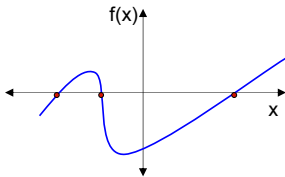
e.g. in above example, $r_k < 10^{-6}$ always followed by $r_{k+1} < 0.0168$

von Neumann: "Anyone who uses arithmetic methods to produce random numbers is in a state of sin."

102

Numerical root finding (Numerical Recipes, Chapter 7)

- A basic problem in mathematics: solve an equation $g(x) = y$
- Convenient convention: rewrite this as $f(x) = 0$
- The solutions are called the roots of f



109

Numerical root finding

- We can also have a multidimensional problem in which coupled sets of equations must be solved

$$f_1(x_1, x_2, \dots) = 0$$

$$f_2(x_1, x_2, \dots) = 0 \quad \rightarrow \quad \mathbf{f}(\mathbf{x}) = \mathbf{0}$$

(Just spent two lectures doing this for the linear case $\mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}$. *MUCH* harder for the general multi-D non-linear case; will discuss this more later, but first let's look at the 1-D non-linear case, $f(x) = 0$)

110

Bracketing

- How do you know if a root exists?
If there are no singularities, then if you can find an interval $[a, b]$ such that $f(a)$ and $f(b)$ have opposite signs, there must be at least one root between a and b

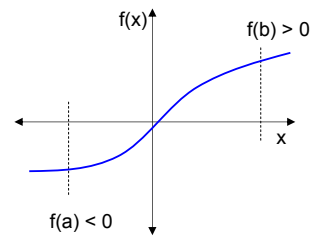
Establishing such an interval is called "bracketing"

Root finding without a bracket is tricky

111

Bracketing

- If there are no singularities, there must be a root between a and b



112

Root finding algorithms

- All root finding algorithms are iterative
 - need a good guess (or small bracket) as a starting point

Like numerical integration, we need to know something about our problem to create an optimal solution: making a graph never hurts

113

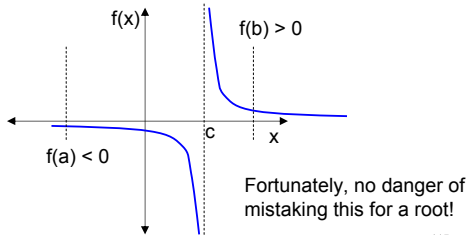
Root finding algorithms

- In the 1-D case, there is a tradeoff between speed and robustness
- Slower methods maintain a bracket at all times (bisection, "false position") – successive iterations simply contract the bracket
- Faster methods do not maintain a bracket and can shoot unstably away from the root (Newton-Raphson, "secant")
- Hybrid methods seek to get the best of both worlds (Brent, Ridder, "safe Newton-Raphson")

114

Bracketing

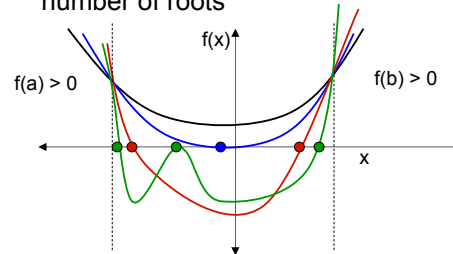
- A bracket is usually sufficient, although a singularity is still possible: e.g. $f(x) = (x - c)^{-1}$



115

Bracketing

- The absence of a bracket can imply any number of roots



116

Finding an initial bracket

- Recipes* describes and provides two bracket-finding routines:
- ZBRAC: expands a seed interval looking for a bracket
- ZBRAK: subdivides a seed interval looking for brackets (good for multiple roots)

117

Pathologies

- Some functions have roots, but it is impossible to find brackets
e.g. *Recipes* equation (3.0.1)

$$0 = f(x) \equiv 3x^2 + (1/\pi^4) \ln [(x - \pi)^2] + 1$$

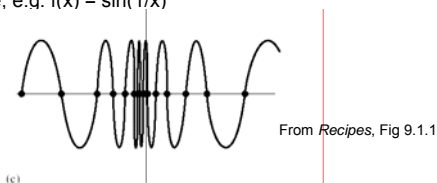
Has a weak singularity at $x = \pi$, two roots very near $x = \pi$, but is negative only in the interval $x = \pi \pm 10^{-667}$

You will probably never find a bracket even in quad precision

118

Pathologies

- Some functions have infinitely many roots over a finite range, e.g. $f(x) = \sin(1/x)$



- MORAL: understand your function before jumping into a numerical solution

119

Root finding algorithms: bisection

- Once you have found a bracket, it can always be narrowed using the bisection method:
 - Divide $[a, b]$ into $[a, \frac{1}{2}(a+b)]$ and $[\frac{1}{2}(a+b), b]$
 - Keep the valid bracket and repeat until the interval reaches a preset size
 - Totally foolproof – will also find singularities if they exist

120

Performance of bisection

- In bisection, the size of the bracket after n iterations is $\varepsilon_n = \frac{1}{2} \varepsilon_{n-1}$
- This is called linear convergence, because $\varepsilon_n \propto (\varepsilon_{n-1})^m$ with $m = 1$
- Faster methods have $m > 1$ and have “supralinear” convergence

121

Performance of bisection

- Note that linear convergence isn't at all bad: we still have $\varepsilon_n = 2^{-n} \varepsilon_0$
- Number of iterations needed to achieve precision ε is $\log_2(\varepsilon_0/\varepsilon)$
- Need to quit anyway when ε approaches machine accuracy (for which n might typically be ~ 40 in double precision)

122

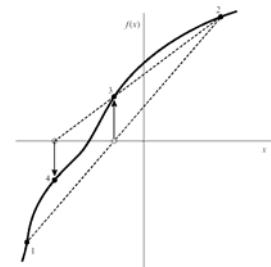
Secant and false position methods

- Bisection only makes use of information about the sign of the function. We can do better if we consider also the magnitude
- The secant and false position methods both use linear interpolation to decide how to narrow the bracket
 - False position always maintains a valid bracket
 - Secant doesn't necessarily, and is therefore less robust (but faster)

123

Geometric representation

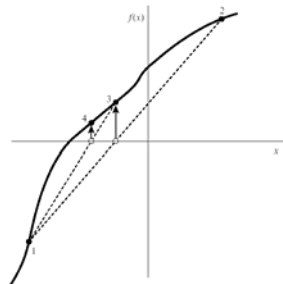
- Secant method: picture from *Recipes*, showing how an initial interval $[1,2]$ is narrowed to $[1,3]$ and then to $[4,3]$
- Convergence is supralinear, with $m = 1.618$



Extrapolate or interpolate from two most recently evaluated points 124

Geometric representation

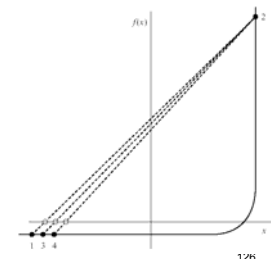
- False position method: picture from *Recipes*, showing how an initial bracket $[1,2]$ is narrowed to $[1,3]$ and then to $[1,4]$ as necessary to maintain a valid bracket
- Convergence is somewhat supralinear, with $1 < m < 1.618$ depending upon the exact function



Interpolate between the two most recently evaluated points that bracket the root 125

Pathologies

- Neither secant nor false position works very well in all circumstances: example using false position from *Recipes* Fig 9.2.3 $[1,2] \rightarrow [3,2] \rightarrow [4,2] \dots$
- In this case, bisection would be much better



126

Brent's method

(Van Wijngaarden-Dekker-Brent method developed in 1960s)

- Hybrid method which is generally the method of choice for 1-D functions*
- Improvements over false secant method
 - Using quadratic interpolation using *three* previously calculated points
 - Switches to bisection when that proves faster
- Faster convergence for well-behaved functions
- Always does at least as well as bisection

*unless the derivative is known: see Newton-Raphson

127

2007 Lecture 6 starts here

Newton-Raphson

- When the derivative of $f(x)$ is known, and when $f(x)$ is well behaved, the celebrated (and ancient) Newton-Raphson method gives the fastest convergence of all ("quadratic", i.e. $m = 2$, such that $\epsilon_n = \epsilon_{n-1}^2$)

- Relies on the Taylor expansion

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

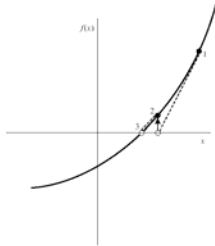
If *ith* iteration, x_i , is close to the root, then for the next iteration, try $x_{i+1} = x_i + \delta$ with $\delta = -f(x_i) / f'(x_i)$

128

Geometric representation

- Newton-Raphson method: picture from *Recipes*

- Convergence is quadratic, with $m=2$



Extrapolate the local derivative to find the next estimate of the root

129

Newton-Raphson

- Convergence is rapid, and the method is very useful for "polishing" a root (i.e. refining an estimate that is nearly correct)

Clearly, a few iterations usually yields an accurate result in the limit of small δ , because terms of order $\frac{1}{2} \delta^2 f''(x)$ or higher are much smaller than $\delta f'(x)$

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

Exception: when $f'(x)$ is very small (or zero)

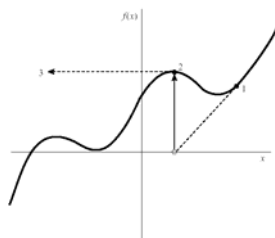
130

Newton-Raphson

- Cases where $f'(x)$ is small (i.e. where there is a local extremum) can send the solution shooting off into outer space

... unless the method is modified to keep the solution within a known bracket

(as in routine "RTSAFE")



131

Summary of 1-D root finding

- Preferred methods:
 - If f' is known analytically, "safe Newton-Raphson" is the preferred method
 - If f' is not known analytically, Brent's method is the method of choice
 - Could use "safe Newton-Raphson" with the derivative computed numerically, but the latter takes up as much time as is saved by the use of N-R
 - Newton-Raphson is also useful for rapid "polishing" (very close to the solution) and for multidimensional root finding.

132

Roots of polynomial functions

- General polynomial equation

$$0 = P(x) \equiv \sum a_k x^k$$

has an analytic solution for cases up to and including the quartic

No general analytic solution for quintics and higher (“the equation that couldn’t be solved”)

133

Roots of polynomial functions

- General features:

Polynomial of order N will have N roots, which can be real or complex and may or may not be distinct

If the a_k are all real, the roots may be real or complex, but any complex roots will occur in pairs of complex conjugates, $a \pm bi$

Polynomials can be ill-conditioned: a small change in the coefficients can cause a large change in the roots

134

Roots of polynomial functions

- General features:

Degenerate roots, or nearly equal roots, present the biggest numerical problems

Example: $P(x) \equiv (x - c)^2 = 0$

- Brackets don’t exist since $P(x)$ is never negative
- Derivative $f'(x) = 0$ at the solution $x = c$
 - Newton-Raphson is potentially unstable

135

Deflation

- Every time a root, x_k , is found, the order of the polynomial can be decreased by one, and the equation becomes

$$0 = P(x) = (x - x_k) Q(x)$$

where $Q(x)$ can be obtained by “synthetic division” (see §5.3 in *Recipes*)

- This procedure, known as “deflation”, allows the roots to be obtained one-by-one

136

Laguerre’s method for obtaining a single root

- Mathematical background

$$P(x) = (x-x_1)(x-x_2) \dots (x-x_n)$$

$$\ln |P(x)| = \ln |x-x_1| + \ln |x-x_2| + \dots + \ln |x-x_n|$$

$$G \equiv d \ln |P(x)| / dx = (x-x_1)^{-1} + (x-x_2)^{-1} + \dots + (x-x_n)^{-1}$$

$$H \equiv -d \ln |P(x)| / dx^2 = (x-x_1)^{-2} + (x-x_2)^{-2} + \dots + (x-x_n)^{-2}$$

137

Laguerre’s method for obtaining a single root

- Now make a “drastic set of assumptions”

We suppose x_1 is located at distance a from our current guess, x , and that all the other roots lie at a distance b

$$G = (x-x_1)^{-1} + (x-x_2)^{-1} + \dots + (x-x_n)^{-1} = a^{-1} + (n-1) b^{-1}$$

$$H = (x-x_1)^{-2} + (x-x_2)^{-2} + \dots + (x-x_n)^{-2} = a^{-2} + (n-1) b^{-2}$$

We can then eliminate b from the above equations to obtain a quadratic equation for a^{-1}

138

Laguerre's method for obtaining a single root

We then obtain

$$a = \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}}$$

This, of course, is not the exact value of $(x - x_i)$, because of the approximation made in obtaining it, but it is useful for obtaining the next estimate of x_i , which we take as $(x - a)$

139

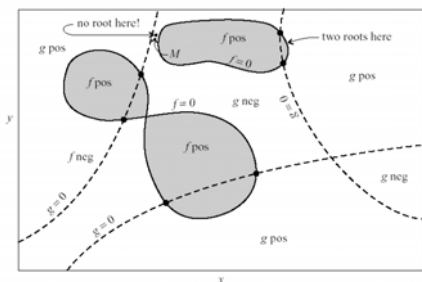
Overview

- Find roots one at a time, using Laguerre's method iteratively until a is sufficiently small
- After each root is found, reduce the order of the polynomial by deflation.

140

Multidimensional non-linear systems of equations

- In general, a horrible problem, for reasons shown schematically in *Recipes*, Fig 9.6.1



141

Multidimensional non-linear systems of equations

- The only general algorithm is Newton-Raphson

Generalization to multidimensional case

$$f(x + \delta) = f(x) + \delta f'(x) + (1/2) \delta^2 f''(x) + \dots$$

becomes

$$f(x + \delta) = f(x) + J(x) \delta + O(\delta^2)$$

where J is the Jacobian matrix, with $J_{ik} = \partial f_i / \partial x_k$

142

Multidimensional non-linear systems of equations

The next iteration on \mathbf{x} is therefore

$$\mathbf{x}' = \mathbf{x} + \delta = \mathbf{x} - \mathbf{J}^{-1} \mathbf{f}(\mathbf{x})$$

where $\delta = -\mathbf{J}^{-1} \mathbf{f}(\mathbf{x})$ is obtained by the solution of the linear set of equations $\mathbf{J} \delta = -\mathbf{f}$

(which we know how to do! ☺)

As in 1-D, this only works with a smooth function or a good first guess

143

Newton-Raphson with a modified step size

- Can improve the robustness of multidimensional NR by considering the quantity:

$$Q \equiv \frac{1}{2} |\mathbf{f}|^2 = \frac{1}{2} \sum f_i^2$$

which tends to zero at the root

Any good step should make Q smaller

144

Newton-Raphson with a modified step size

- Consider now the gradient of Q

$$\partial Q / \partial x_j = \partial (\frac{1}{2} |\mathbf{f}|^2) / \partial x_j = \sum f_i \partial f_i / \partial x_j$$

$$\rightarrow \nabla Q = \mathbf{f}^T \mathbf{J}$$

Our step $\delta = -\mathbf{J}^{-1} \mathbf{f}$ is in the *right direction* to reduce Q, because

$$\nabla Q \cdot \delta = -\mathbf{f}^T (\mathbf{J} \mathbf{J}^{-1}) \mathbf{f} = -2Q < 0$$

145

Newton-Raphson with a modified step size

So if adding δ to \mathbf{x} doesn't decrease Q, a sufficiently small step in *the same direction* must

Strategy: if the solution starts to overshoot – as indicated by monitoring of $Q \equiv \frac{1}{2} |\mathbf{f}|^2$ – use a smaller step $\lambda \delta$ in the same direction (where λ is a positive scalar < 1)

See *Recipes* §9.7 for the details of how to choose λ

This is implemented in the function NEWT, which will successfully find a root for almost any reasonable initial guess.

146

Example application: interstellar chemistry

- More than one hundred different molecules have been detected in the interstellar gas
- They are formed and destroyed by a complex network of reactions involving bimolecular reactions and unimolecular processes such as photodissociation

147

Example application: interstellar chemistry

- Consider the reaction $A + B \rightarrow C + D$:
This destroys A and B (and creates C and D) at a rate $k_r n(A)n(B)$ per unit volume, where $n(X)$ is the density of species X (number per unit volume) and k_r is the rate coefficient (units: $\text{cm}^3 \text{s}^{-1}$) for the reaction in question
- Molecule A might also be destroyed by photodissociation, $A + h\nu \rightarrow E + F$
This destroys A (and creates E and F) at a rate $\zeta_p n(A)$ per unit volume, where $n(X)$ is the density of species X (number per unit volume) and ζ_p is the photodissociation rate (units: s^{-1}) for this process

148

Rate equations for interstellar chemistry

- The density of the i th molecule therefore obeys the rate equation:

$$\frac{\partial n_i}{\partial t} = \sum_j \sum_k k_{ijk} n_j n_k + \sum_k \zeta_{ik} n_k - n_i \sum_m \sum_k k_{mik} n_k - n_i \sum_m \zeta_{mi}$$

Sum of rate coefficients for all reactions of j and k that produce i

Total rate at which k undergoes unimolecular processes to produce i

149

Equilibrium solution

- In equilibrium, the molecular densities will obey $\mathbf{f}(\mathbf{n}) = \mathbf{0}$

where n_i is the density of molecule number i , and $f_i \equiv \partial n_i / \partial t$ is a quadratic function of the n_i

A multidimensional root-finding problem!
...and one for which the Jacobian is easily computed (quadratic function)

$$\partial f_i / \partial n_k = \sum_j k_{ijk} n_j + \zeta_{ik} - n_i \sum_m k_{mik} \quad (i \neq k)$$

$$\partial f_i / \partial n_i = -\sum_m k_{mik} n_k - \sum_m \zeta_{mi} \quad (\text{diagonal elements})$$

Newton's method generally works well

150

Complication

- The equilibrium equations, as written, are singular, since they don't uniquely specify the total density of molecules
We need to apply some constraints of the form

$$\sum n_i N_{ic} = n(E_c)$$

N_{ic} = number of atoms of element C contained in molecule i
 $n(E_c)$ = (fixed) density of C nuclei

151

Non-equilibrium solution

- In astrochemistry, the timescale for reaching equilibrium can sometimes be long compared to the ages of molecular clouds

Therefore, it is also interesting to integrate the equations

$$\partial n_i / \partial t = \sum_j \sum_k k_{ijk} n_j n_k + \sum_k \zeta_{ik} n_k - n_i \sum_m \sum_k k_{mik} n_k - n_i \sum_m \zeta_{mi}$$

(a coupled set of ODE's, to be considered later)

In the limit of large t, this tends to the equilibrium solution

→ an alternative method even if we are uninterested in the time evolution

152

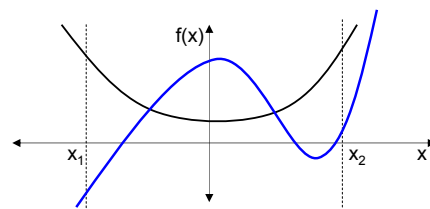
Lecture 7: Minimization or maximization of functions (*Recipes* Chapter 10)

- Actively studied subject for several reasons:
 - Commonly encountered problem: e.g. Hamilton's and Lagrange's principles, economics problems, statistical fitting of data (χ^2 or maximum likelihood)....
 - For the most interesting cases (multivariate, non-linear functions), there is no "best technique"
 - There are many competing methods each with some advantages and disadvantages

153

Minimization of 1-D functions

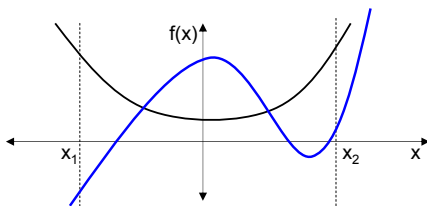
- We will search for a minimum of a function $f(x)$ on some interval $[x_1, x_2]$



154

Minimization of 1-D functions

- Notes:
 - No loss of generality in focusing on minimum: for maximum consider the function $-f(x)$
 - Global minimum need not have $f'(x) = 0$



155

How accurately can the minimum be found?

- Suppose a minimum of $f(x)$ occurs at $x = b$ (in the case where $f'(b) = 0$)

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

Define $\delta f = f(x) - f(b)$ as the smallest difference in FP numbers that we can distinguish:

$$\text{Then } \delta f = \varepsilon f(b)$$

with $\varepsilon \sim 10^{-8}$ in single precision or $\sim 10^{-16}$ in double precision

156

How accurately can the minimum be found?

- Then

$$\delta f = \varepsilon f(b) = \frac{1}{2} f''(b) (x-b)^2$$

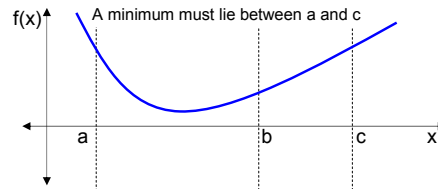
$$\text{and } |x-b| = \sqrt{2 \varepsilon f(b) / f''(b)} \sim b \varepsilon^{1/2}$$

→ we typically cannot resolve minima with fractional accuracies better than $\sim 10^{-4}$ in single precision

157

Bracketing for minimization

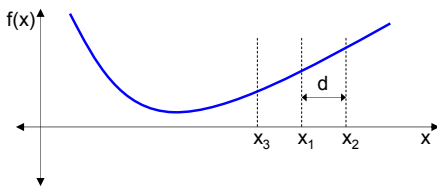
- As in the case of root finding, the best 1-D techniques make use of bracketing. In this context, a "bracket" is defined by three points, $a < b < c$, for which $f(b) < f(a)$ AND $f(b) < f(c)$



158

Establishing a bracket

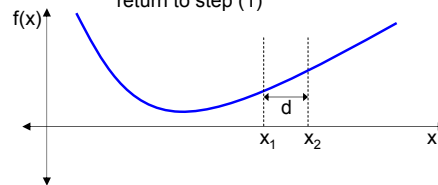
- Choose two points, $x_1 < x_2$, separated by $d = |x_2 - x_1|$
- If $f(x_1) < f(x_2)$, choose $x_3 = x_1 - d$
else, choose $x_3 = x_2 + d$



159

Establishing a bracket

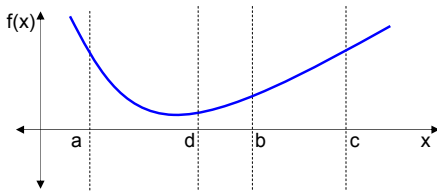
- Evaluate $f(x_3)$
If $f(x_3) > \min [f(x_2), f(x_1)]$, we are done
else, set $x_2 = x$ ($\min [f(x_2), f(x_1)]$), $x_1 = x_3$
return to step (1)



Can accelerate this by allowing the step size to grow 160

Golden section search

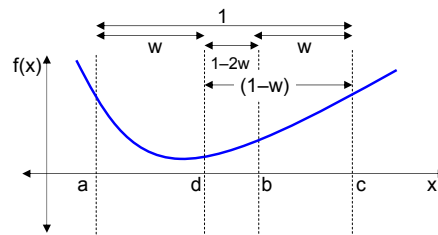
- We now want to squeeze the bracket:
 - put in a new point d
 - New bracket is either $\{a, d\}$ or $\{d, b\}$



161

Golden section search

- Choose the distances shown below such that the ratios are preserved $(1-w):w = w:(1-2w)$
Leads to self-similarity



162

Golden section search

- The required value of w is the solution to $w^2 - 3w + 1 = 0$
 $\rightarrow w = \frac{1}{2} (3 - \sqrt{5}) = 0.38197\dots = 1 - \phi$
 (need solution with $w < 1$)
- This procedure converges *linearly*, with bracket size after N iterations given by $(x_2 - x_1) 0.618^N$

cf. bisection for root finding yields bracket size $(x_2 - x_1) 0.5^N$ after N iterations

163

Faster methods

- As with bisection, in the Golden section method we only ask about whether certain quantities (e.g. $f(d) - f(c)$) are positive or negative)
- We can accelerate convergence by using more information about the values of various quantities

164

Brent's method

(a.k.a. inverse parabolic interpolation)

- In Brent's method, we expand about the true minimum, x^*

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

If $R(x)$ were zero, we would have three unknowns: x^* , $f(x^*)$, and $f''(x^*)$

three data points: $f(a)$, $f(b)$, $f(c)$
 (from the function values on our three bracket points)

165

Brent's method

- The solution is

$$x_4 = x_3 - \frac{(x_3 - x_1)^2 [f_3 - f_2] - (x_3 - x_2)^2 [f_3 - f_1]}{2 (x_3 - x_1) [f_3 - f_2] - (x_3 - x_2) [f_3 - f_1]}$$

- If x_4 is reasonable – i.e. lies in the interval $[x_1, x_2]$ and yields $f_4 < f_3$ (previous smallest value) use it to form a new bracket
- Otherwise, revert to Golden section

166

Brent's method

For the case $R(x) \sim \frac{1}{6} f'''(x^*) (x - x^*)^3$,

we find that $|x_4 - x^*| \sim [2f'''(x^*) / 2f''(x^*)]^{1/2} |x_3 - x^*|^{3/2}$

- \rightarrow supralinear convergence ($m=1.5$) when it works (or $m = 1$ when it reverts to Golden section)

Hybrid method: combines robustness (valid bracket always maintained) with increased speed when possible

167

Use of derivative information

- When $f'(x)$ is known explicitly, this information can be used to further improve performance
 - Recipes* has a hybrid routine that uses the secant method to find the root of $f'(x)$ with the Golden section method to ensure that a bracket is maintained

168

Multi-D minimization (*Numerical Recipes*, §10.4 – 10.7)

- As with root finding, things get a lot harder when f is a function of several variables
 - no analog to a “bracket”
- Overview of techniques
 - Function evaluations only → downhill simplex method
 - Function evaluation to estimate the optimum direction of motion → Powell’s method
 - Function evaluations and explicit gradient calculation → Conjugate Gradient Method

169

Downhill simplex

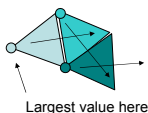
- A simplex is a hyperpolygon of $N + 1$ vertices in an N -dimensional space
 - $N = 2$: triangle
 - $N = 3$: tetrahedron
- If one vertex is at the origin of the coordinate system, the others are given by N vectors which span the N dimensional space:

$$V_i = P_i - P_0 \quad (i = 1, N), \text{ where } P_i \text{ is the } i\text{th vertex}$$

170

Downhill simplex

- Downhill simplex involves moving a simplex downhill to find the minimum of a function
- Basic move: reflection in the face opposite the vertex for which f is largest



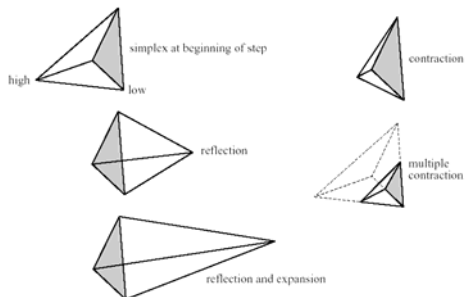
171

Downhill simplex

- Additional moves:
 - Stretch to accelerate motion in a particular direction
 - Contraction, if reflection overshoots the minimum
- Press et al. name their routine AMOEBA

172

3–D representation (from Recipes)



173

Direction set methods

- Basic tool of all such methods is a 1-D minimization (Golden section, Brent’s method)
- Choose a starting position \mathbf{p} , and a direction $\hat{\mathbf{n}}$, and minimize $f(\mathbf{p} + \lambda \hat{\mathbf{n}})$
- Now use $\mathbf{p} + \lambda \hat{\mathbf{n}}$ as the new starting position, choose a different direction, and minimize along that direction.....
- Methods differ as to how the directions are chosen

174

Direction set methods

- Simplest method: take N orthogonal unit vectors in turn, \hat{e}_i
- Slow convergence, unless the unit vectors are well oriented with respect to the valley.

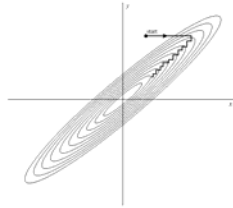


Figure 10.5.1: Recursive reorientation along coordinate directions in a long narrow "valley" allows an optimal search. Unlike the valley in steepest descent, the optimal coordinate method, taking many tiny steps to get to the minimum, converges and reorienting the principal axes.

Recipes, Fig 10.5.1

175

Direction set methods

- Better methods update the directions as the method proceeds, so as to
 - choose favorable directions that proceed far along narrow valleys
 - choose “non interfering” directions, such that the next direction doesn’t undo the minimization achieved by previous steps

176

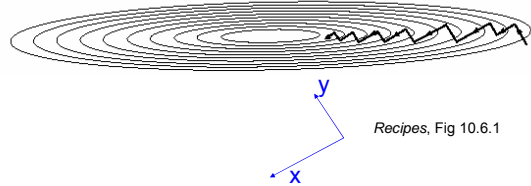
Steepest descent

- If you know the derivatives of f (i.e. you know ∇f), you might think that you would do best to choose $\hat{n} = -\nabla f / |\nabla f|$
- This is the method of steepest descent
- BUT, this means you always choose a new direction that is *orthogonal* to the previous direction
i.e. $\hat{n}_{i+1} \cdot \hat{n}_i = 0$

177

Steepest descent

- The performance isn’t that good, because we can only ever take a right angle turn



Recipes, Fig 10.6.1

178

Steepest descent: 2-D example

- Suppose step k occurred along the y -axis, and led to position \mathbf{p}_{k+1} , at which $\partial f / \partial y = 0$.
- Next step is along the x -axis: that step leads to a position \mathbf{p}_{k+2} , where $\partial f / \partial x = 0$
- But if $\partial^2 f / \partial y \partial x$ is non-zero, $\partial f / \partial y$ will no longer be zero.
- We really want to move along some direction other than the x -axis, such that $\partial f / \partial y$ remains zero.

179

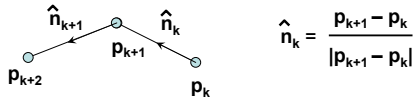
Lecture 8: Conjugate directions

- Thus the optimum direction is not along $-\nabla f$, but rather in a direction that preserves the minimization achieved in the previous step (and, in multi-dimensions, all previous steps)

This is called the conjugate direction

180

Conjugate directions



By the definition of a minimum, we will always have
 $\hat{n}_k \cdot \nabla f(\mathbf{p}_{k+1}) = 0$
 $\hat{n}_{k+1} \cdot \nabla f(\mathbf{p}_{k+2}) = 0$

In addition, we would also like $\hat{n}_k \cdot \nabla f(\mathbf{p}_{k+2}) = 0$

181

Conjugate directions

To arrange this, let's consider the Taylor expansion:

$$f(\mathbf{x} + \delta\mathbf{x}) = f(\mathbf{x}) + \delta\mathbf{x} \cdot \nabla f(\mathbf{x}) + \frac{1}{2} \delta\mathbf{x} \cdot \mathbf{A} \cdot \delta\mathbf{x} + \dots$$

where $A_{ij} = \partial^2 f / \partial x_i \partial x_j$ is the Hessian of f

Taking the gradient of the Taylor expansion, we obtain

$$\nabla f(\mathbf{x} + \delta\mathbf{x}) = \nabla f(\mathbf{x}) + \mathbf{A} \cdot \delta\mathbf{x} + \dots$$

182

Conjugate directions

Then, our wish to make $\mathbf{n}_k \cdot \nabla f(\mathbf{p}_{k+2}) = 0$ requires $0 \sim \mathbf{n}_k \cdot (\nabla f(\mathbf{p}_{k+1}) + \mathbf{A} \cdot \mathbf{n}_{k+1})$

$\rightarrow \hat{n}_k \cdot \mathbf{A} \cdot \hat{n}_{k+1} = 0$
 (definition: \hat{n}_k and \hat{n}_{k+1} are "conjugate")

Zero, because \mathbf{p}_{k+1} was obtained by minimizing f along the \mathbf{n}_k direction

Clearly, this is different from steepest descent (for which $\hat{n}_k \cdot \hat{n}_{k+1} = 0$) in the case where the Hessian has off-diagonal elements (e.g. $\partial^2 f / \partial y \partial x$ non zero in the 2-D case)

183

Conjugate directions

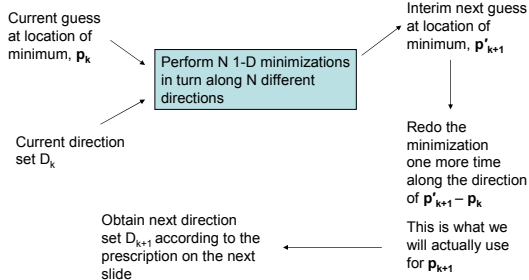
- The optimal procedure is to minimize f in turn along N directions, such that the k th direction \hat{n}_k is conjugate to all previous directions
- How do we find such directions?
 - It would be simple if we knew the Hessian, but second derivatives are hard to compute numerically with good accuracy (and need $O(N^2)$ operations, of course)
 - Powell's method: requires no knowledge of derivatives
 - Conjugate gradient method: requires knowledge of first derivative only

184

Powell's method

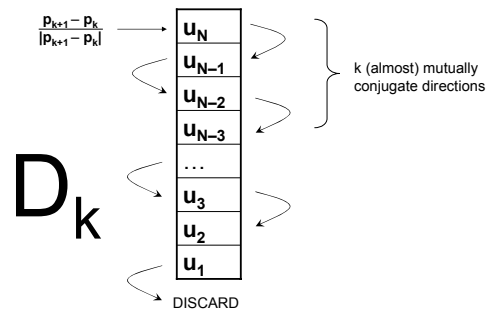
(doesn't require us to know any derivatives)

Procedure:



185

Powell's method



186

Powell's method

After N passages through the blue box (requiring $N(N+1)$ 1-D minimizations!) the entire direction set is mutually conjugate, and the procedure yields quadratic convergence

Problem:

Unfortunately, the direction set tends towards becoming linearly dependent

Solutions:

(1) after each cycle of $N(N+1)$ 1-D minimizations: reset the direction set to an orthogonal one; or
 (2) Throw away the direction that gave the largest decrease in f instead of \mathbf{u}_1 (lose quadratic convergence)

Solution (2) is implemented in subroutine POWELL

187

Conjugate gradient method

- First step: 1-D minimization along the direction of $\mathbf{h}_0 = -\nabla f(\mathbf{p}_0)$, call it \mathbf{h}_0
 \rightarrow move to \mathbf{p}_1
- Next direction is along:
 $\mathbf{h}_1 = -\nabla f(\mathbf{p}_1) + \mathbf{h}_0 \lambda$
 (close to $-\nabla f(\mathbf{p}_1)$, but mixes in a little bit of the previous step)

188

Conjugate gradient method

- What λ makes this conjugate to \mathbf{h}_0 ?

We require

$$0 = \mathbf{h}_1 \cdot \mathbf{A} \cdot \mathbf{h}_0 = (-\nabla f(\mathbf{p}_1) + \lambda \mathbf{h}_0) \cdot \mathbf{A} \cdot \mathbf{h}_0$$

But $\mathbf{A} \cdot \mathbf{h}_0$ is parallel to $\nabla f(\mathbf{p}_1) - \nabla f(\mathbf{p}_0)$ (Taylor expansion)

So this requirement becomes

$$0 = (-\nabla f(\mathbf{p}_1) + \lambda \mathbf{h}_0) \cdot (\nabla f(\mathbf{p}_1) - \nabla f(\mathbf{p}_0)) \\ = -|\nabla f(\mathbf{p}_1)|^2 + \lambda \mathbf{h}_0 \cdot \nabla f(\mathbf{p}_1) - \lambda \mathbf{h}_0 \cdot \nabla f(\mathbf{p}_0) + \nabla f(\mathbf{p}_1) \cdot \nabla f(\mathbf{p}_0)$$

Zero, because \mathbf{p}_1 is a minimum

189

Conjugate gradient method

- What λ makes this conjugate to \mathbf{h}_0 ?

- Hence the required value of λ is

$$|\nabla f(\mathbf{p}_1)|^2 / |\nabla f(\mathbf{p}_0)|^2$$

$$\text{i.e. } \mathbf{h}_1 = -\nabla f(\mathbf{p}_1) + \mathbf{h}_0 |\nabla f(\mathbf{p}_1)|^2 / |\nabla f(\mathbf{p}_0)|^2$$

190

Conjugate gradient method

- Can be proven by induction that the i th direction is along

$$\mathbf{h}_i = -\nabla f(\mathbf{p}_i) + \mathbf{h}_{i-1} |\nabla f(\mathbf{p}_i)|^2 / |\nabla f(\mathbf{p}_{i-1})|^2$$

The gradients at any two successive positions are orthogonal:

$$\nabla f(\mathbf{p}_i) \cdot \nabla f(\mathbf{p}_{i-1}) = 0$$

- Requires only N 1-D minimizations
 (of course, we need to be able to write down ∇f)

191

Variable metric techniques (brief summary)

- Can look for the solution of $\nabla f(\mathbf{x}) = \mathbf{0}$ making use of the Taylor expansion
 $\nabla f(\mathbf{x} + \delta \mathbf{x}) = \nabla f(\mathbf{x}) + \mathbf{A} \cdot \delta \mathbf{x} + \dots$

Here, the i th estimate of the minimum is

$$\mathbf{x}_i = \mathbf{x}_{i-1} - \mathbf{A}^{-1} \nabla f(\mathbf{x}_{i-1})$$

192

Variable metric techniques

- Problems:
 - Need to calculate 2nd derivatives
 - Far from the minimum, this may not converge
 - $\mathbf{x}_i - \mathbf{x}_{i-1}$ may not even be a descent direction
- Various techniques have been developed to deal with the 2nd of these.

193

Lecture 9: Linear Programming

- A common optimization problem involves finding the maximum of a linear function of N variables

$$Z = \sum_{i=1}^N a_i x_i \quad (\text{the "objective function"})$$

where the x_i are all non-negative

194

Linear Programming

...subject to a series of M constraints

$$\sum_{i=1}^N c_{ij} x_i \begin{cases} \leq \\ = \\ \geq \end{cases} b_j \quad j = 1, 2, 3, \dots, M$$

where the b_j are all non-negative

195

Linear Programming

- The constraints are each represented by N-1 dimensional hyperplanes within the N dim space
 - these form a convex hyperpolygon in N dimensions bounding a region of volume V^N
 - if V^N is zero (all volume eliminated), there is no solution
 - if V^N is > 0 , then the gradient $\nabla Z = \sum_{i=1}^N a_i \mathbf{e}_i$ leads us to a boundary plane
 - Unless ∇Z is perpendicular to the boundary plane, we can follow that plane to an edge and that edge to a vertex → a single optimal vector

196

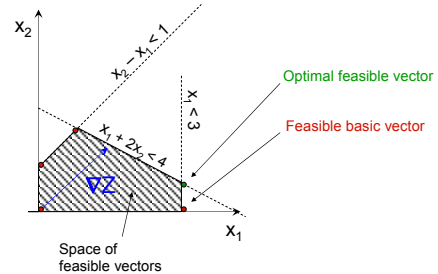
Feasible vectors

- The space of vectors satisfying the constraints is called the space of "feasible vectors", and the vertex that maximizes Z is the "optimal feasible vector"
- Vertices of the hyperpolygon are places that satisfy N constraints as equalities: these are called basic feasible vectors. (Here, the "constraints" include the non-negativity requirement on the x_i).

197

Feasible vectors

- 2-D example: $Z = x_1 + x_2$



198

The fundamental theorem of linear programming

“If an optimal feasible vector exists, then there is a feasible basic vector that is optimal”

i.e. the optimal vector is at one of the vertices
Our goal is to find which vertex (i.e. which N of the constraints does it satisfy)

199

Feasible basic vectors

The total number of locations at which N constraints are satisfied simultaneously can be very large:

It can be as large as the number of ways of choose N items out of N+M, which comes to $(N+M)! / (N! M!)$

In the 2-D example, N=2, M=3, and so $(N+M)! / (N! M!) = 5! / (2! 3!) = 10$

of which 5 lie in the feasible space and one is missing (since $x_1=0$ and $x_1=3$ cannot be satisfied simultaneously)

200

Restricted normal form

- The constraints can always be written such that
 - All constraints are written as equalities
 - Number of constraints $M \leq N$
 - Each constraint equation has a variable that appears in that equation alone (How to do this to be discussed later)
- This permits us to obtain a solution via the “simplex method”

201

The simplex method (illustrated by example)

- Example (N=4, M=2)

$$\text{Maximize } z = -4x_1 - 25x_2 + 4x_3 + x_4$$

$$\text{subject to } \begin{aligned} x_1 + 6x_2 - x_3 &= 2 \\ -3x_2 + 4x_3 + x_4 &= 8 \end{aligned}$$

Move **variables** that appear in only one constraint equation to the left hand side and call them **left hand variables**

202

Solution via the simplex method

This yields
$$\begin{aligned} x_1 &= 2 - 6x_2 + x_3 \\ x_4 &= 8 + 3x_2 - 4x_3 \end{aligned}$$

which we substitute into the objective function to obtain the latter in terms of **right hand variables**

$$z = 2x_2 - 4x_3$$

There are **M left hand variables** and **N - M right hand variables**

203

Solution via the simplex method

Represent these equations by a “tableau”:

$$\begin{aligned} z &= 2x_2 - 4x_3 \\ x_1 &= 2 - 6x_2 + x_3 \\ x_4 &= 8 + 3x_2 - 4x_3 \end{aligned}$$

is written

	const	x_2	x_3
z	0	+2	-4
x_1	+2	-6	+1
x_4	+8	+3	-2

204

Solution via the simplex method

Setting the **RH variables** equal to zero, we can immediately obtain a basic feasible vector (but not, in general, the optimal one) $x = (+2, 0, 0, +8)$

Increasing x_3 from zero will clearly decrease z , so we conclude that the optimal feasible vector must have $x_3 = 0$

	const	x_2	x_3
z	0	+2	-4
x_1	+2	-6	+1
x_4	+8	+3	-2

205

Solution via the simplex method

We now “pivot” about the negative entry that limits how large x_2 can become, and switch the “handedness” of x_1 and x_2 .

$$x_1 = 2 - 6x_2 + x_3 \rightarrow x_2 = 1/3 - x_1/6 + x_3/6$$

$$x_4 = 8 + 3x_2 - 4x_3 \rightarrow x_4 = 8 + 3(1/3 - x_1/6 + x_3/6) - 4x_3 = 9 - x_1/2 - 7x_3/2$$

$$z = 2x_2 - 4x_3 \rightarrow z = 2(1/3 - x_1/6 + x_3/6) - 4x_3 = 2/3 - x_1/3 - 11x_3/3$$

206

Solution via the simplex method

Our new **LH variables** are x_1 and x_3 and our new **RH variables** are x_4 and x_2

And our new tableau is

	const	x_1	x_3
z	+2/3	-1/3	-11/3
x_2	+1/3	-1/6	+1/6
x_4	+9	-1/2	-7/2

207

Solution via the simplex method

Setting the **RH variables** to zero, we find that the feasible basic vector we can immediately write is now

$$x = (0, +1/3, 0, +9), \text{ for which } z = 2/3$$

Because all the entries in the z -row are now negative, this is the *optimum* feasible vector, because increasing either RH variable will only decrease z

	const	x_1	x_3
z	+2/3	-1/3	-11/3
x_2	+1/3	-1/6	+1/6
x_4	+9	-1/2	-7/2

208

Generalization to N variables and M constraints

	const					
z	>0	<0	<0	>0	<0	<0
	<0			<0		

← “z-row”

N - M

209

Keep pivoting until the “z row” is all negative

	const					
z	<0	<0	<0	<0	<0	<0
	<0					

Zero components of optimal feasible vector

Maximum value of z

-z-row

Non-zero components of optimal feasible vector their values

N - M

210

Generalization to N variables and M constraints

- The maximum number of required pivots is $\min(N - M, M)$
- The process can (and should) fail if there are no negative entries in a column headed by a positive pivot
 - implies there is no maximum allowed value of z (i.e. no boundary prevents us from going to infinity along that basis vector)

211

Obtaining the constraints in restricted normal form

- What happens if
 - Some constraints are inequalities
 - Not every constraint equation has a variable that appears in that equation alone
 - Number of constraints $M > N$
- General solution: increase the number of variables!

212

Treatment of inequalities

Suppose we want

$$\sum_{i=1}^N c_{ij} x_i \leq b_j \quad \text{for one particular } j$$

Introduce new non-negative variable, y_j , (called a "slack variable"), and consider the constraint

$$\sum_{i=1}^N c_{ij} x_i + y_j = b_j$$

213

Treatment of inequalities

(Equivalently, for

$$\sum_{i=1}^N c_{ij} x_i \geq b_j \quad \text{for one or more } j$$

we use the constraint

$$\sum_{i=1}^N c_{ij} x_i - y_j = b_j \quad)$$

214

Treatment of inequalities

- Convert all inequalities according to this prescription
 - This increases the dimensionality of the problem from N to N + K, where K is the number of inequalities
 - Solve by the previous method
 - Ignore the solution for the y_j

215

Getting the constraints into restricted normal form

Arranging for every constraint equation to have a variable that appears in that equation alone

Change the constraint equations into "zero form":

$$\text{Define } z_j \equiv \sum_{i=1}^N c_{ij} x_i - b_j$$

The original M constraints become $z_j = 0$

216

Getting the constraints into restricted normal form

First use $Z' = - \sum_{j=1}^M z_j$ as the "auxiliary" objective function

And adopt the definitions of the z_j as the M constraints. Initially, the z_j are all LH variables and the x_i are all RH variables. This is a problem in restricted normal form

The solution which maximizes Z' has all the z_j equal to zero* \rightarrow after the Simplex method is done, all the z_j must become RH variables, and M out of N the x_i will be LH variables.

*if such a solution exists: if not, then the constraints cannot be satisfied simultaneously

217

Getting the constraints into restricted normal form

But now each of the RH variables appears in exactly 1 constraint equation. We zero out all the z_j , and have a problem involving the x_i in restricted normal form.

This we now solve via the Simplex method, using the original objective function

$$Z = \sum_{i=1}^N a_i x_i$$

218

What if the number of constraints exceeds the number of variables?

If $M > N$, we can't possibly have a restricted normal form (there aren't enough variables for each constraint to have a variable that appears in that constraint alone).

But the method described above adds M more variables, so the number of variables always exceeds the number of constraints.

219

Lecture 10: Eigenvectors and eigenvalues (*Numerical Recipes*, Chapter 11)

The eigenvalue problem,

$$\mathbf{A} \mathbf{x} = \lambda \mathbf{x},$$

occurs in many, many contexts:

classical mechanics, quantum mechanics, optics.....

220

Eigenvectors and eigenvalues (*Numerical Recipes*, Chapter 11)

The textbook solution is obtained from

$$(\mathbf{A} - \lambda \mathbf{I}) \mathbf{x} = 0, \text{ which can only hold for } \mathbf{x} \neq \mathbf{0} \text{ if}$$

$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0.$$

This is a polynomial equation for λ of order N

221

Eigenvectors and eigenvalues (*Numerical Recipes*, Chapter 11)

This method of solution is:

- (1) very slow
- (2) inaccurate, in the presence of roundoff error
- (3) yields only eigenvalues, not eigenvectors

- (4) of no practical value
(except, perhaps in the 2 x 2 case where the solution to the resultant quadratic can be written in closed form)

222

Review of some basic definitions in linear algebra

A matrix is said to be:

Symmetric if $\mathbf{A} = \mathbf{A}^T$ (its *transpose*) i.e. if $a_{ij} = a_{ji}$

Hermitian if $\mathbf{A} = \mathbf{A}^\dagger \equiv (\mathbf{A}^T)^*$ i.e. if $a_{ij} = a_{ji}^*$
(its *Hermitian conjugate*)

Orthonormal if $\mathbf{U}^{-1} = \mathbf{U}^T$

Unitary if $\mathbf{U}^{-1} = \mathbf{U}^\dagger$

223

Review of some basic definitions in linear algebra

All 4 types are *normal*, meaning that they obey the relations

$$\mathbf{A} \mathbf{A}^\dagger = \mathbf{A}^\dagger \mathbf{A}$$

$$\mathbf{U} \mathbf{U}^\dagger = \mathbf{U}^\dagger \mathbf{U}$$

224

Eigenvectors and eigenvalues

Hermitian matrices are of particular interest because they have:

real eigenvalues

orthonormal eigenvectors ($x_i^\dagger x_j = \delta_{ij}$)
(unless some eigenvalues are degenerate, in which case we can always design an orthonormal set)

225

Diagonalization of Hermitian matrices

Hermitian matrices can be diagonalized according to

Unitary matrix whose column
are the eigenvectors

$$\mathbf{A} = \mathbf{U} \mathbf{D} \mathbf{U}^{-1}$$

Diagonal matrix consisting
of the eigenvalues

226

Diagonalization of Hermitian matrices

Clearly, if $\mathbf{A} = \mathbf{U} \mathbf{D} \mathbf{U}^{-1}$, then $\mathbf{A} \mathbf{U} = \mathbf{U} \mathbf{D}$
.....which is simply the eigenproblem
(written out N times):

The columns of \mathbf{U} are the eigenvectors (mutually orthogonal) and the elements of \mathbf{D} (non zero only on the diagonal) are the corresponding eigenvalues.

227

Non-Hermitian matrices

We are often interested in non-symmetric real matrices:

- the eigenvalues are real or come in complex conjugate pairs
- the eigenvectors are not orthonormal in general
- the eigenvectors may not even span an N dimensional space

228

Diagonalization of non-Hermitian matrices

For a non-Hermitian matrix, we can identify two (different) types of eigenvectors

Right hand eigenvectors are column vectors which obey:

$$\mathbf{A} \mathbf{x}_R = \lambda \mathbf{x}_R$$

Left hand eigenvectors are row vectors which obey:

$$\mathbf{x}_L \mathbf{A} = \lambda \mathbf{x}_L$$

The eigenvalues are the same (the roots of $\det(\mathbf{A} - \lambda \mathbf{I}) = 0$ in both cases) but, in general, the eigenvectors are not

229

Diagonalization of non-Hermitian matrices

Note: for a Hermitian matrix, the two types of eigenvectors are equivalent, since

$$\begin{aligned} \mathbf{A} \mathbf{x}_R &= \lambda \mathbf{x}_R && \begin{array}{l} \text{since } \mathbf{A} = \mathbf{A}^\dagger \\ \text{since } \lambda \text{ is real} \end{array} \\ \Rightarrow \mathbf{x}_R^\dagger \mathbf{A}^\dagger &= \lambda^* \mathbf{x}_R^\dagger && \Rightarrow \mathbf{x}_R^\dagger \mathbf{A} = \lambda \mathbf{x}_R^\dagger \end{aligned}$$

so if any vector \mathbf{x}_R is a right hand eigenvector, then \mathbf{x}_R^\dagger is a left hand eigenvector

230

Diagonalization of non-Hermitian matrices

- Let \mathbf{D} be the diagonal matrix whose elements are the eigenvalues, $\mathbf{D} = \text{diag}(\lambda_i)$
- Let \mathbf{X}_R be the matrix whose columns are the right hand eigenvectors
- Let \mathbf{X}_L be the matrix whose rows are the left hand eigenvectors
- Then the eigenvalue equations are

$$\mathbf{A} \mathbf{X}_R = \mathbf{X}_R \mathbf{D} \quad \text{and} \quad \mathbf{X}_L \mathbf{A} = \mathbf{D} \mathbf{X}_L$$

231

Diagonalization of non-Hermitian matrices

- The eigenvalue equations

$$\mathbf{A} \mathbf{X}_R = \mathbf{X}_R \mathbf{D} \quad \text{and} \quad \mathbf{X}_L \mathbf{A} = \mathbf{D} \mathbf{X}_L$$

$$\text{imply } \mathbf{X}_L \mathbf{A} \mathbf{X}_R = \mathbf{X}_L \mathbf{X}_R \mathbf{D} \quad \text{and} \quad \mathbf{X}_L \mathbf{A} \mathbf{X}_R = \mathbf{D} \mathbf{X}_L \mathbf{X}_R$$

$$\text{Hence, } \mathbf{X}_L \mathbf{A} \mathbf{X}_R = \mathbf{X}_L \mathbf{X}_R \mathbf{D} = \mathbf{D} \mathbf{X}_L \mathbf{X}_R$$

$$\Rightarrow \mathbf{X}_L \mathbf{X}_R \text{ commutes with } \mathbf{D}$$

$$\Rightarrow \mathbf{X}_L \mathbf{X}_R \text{ is also diagonal}$$

$$\Rightarrow \text{rows of } \mathbf{X}_L \text{ are orthogonal to columns of } \mathbf{X}_R$$

232

Diagonalization of non-Hermitian matrices

It follows that any matrix can be diagonalized according to

$$\mathbf{D} = \mathbf{X}_L \mathbf{A} \mathbf{X}_R = \mathbf{X}_R^{-1} \mathbf{A} \mathbf{X}_R, \text{ or equivalently}$$

$$\mathbf{D} = \mathbf{X}_L \mathbf{A} \mathbf{X}_L^{-1}$$

The special feature of a Hermitian matrix is that the \mathbf{X}_L and \mathbf{X}_R are unitary

233

Solving the eigenvalue equation

The problem is entirely equivalent to figuring out how to diagonalize of \mathbf{A} according to

$$\mathbf{A} = \mathbf{X}_R \mathbf{D} \mathbf{X}_R^{-1}$$

All methods proceed by making a series of similarity transformation

$$\mathbf{A} = \mathbf{P}_1^{-1} \mathbf{M}_1 \mathbf{P}_1 = \mathbf{P}_1^{-1} \mathbf{P}_2^{-1} \mathbf{M}_2 \mathbf{P}_2 \mathbf{P}_1 = \dots = \mathbf{X}_R \mathbf{D} \mathbf{X}_R^{-1}$$

where $\mathbf{M}_1, \mathbf{M}_2 \dots$ are successively more nearly diagonal

234

Solving the eigenvalue equation

There are many routines available:

EISPACK + Linpack – LAPACK (free)
NAG, IMSL (expensive)

Before using a totally general technique, consider what you want:

eigenvalues alone, or eigenvectors as well?

..and how special your case is

real symmetric & tridiagonal, real symmetric, real non-symmetric, complex Hermitian, complex non-Hermitian

235

Solution for a real symmetric matrix: the Householder method (*Recipes*, §11.2)

The Householder method makes use of a similarity transform based upon

$$P = I - 2 w w^T,$$

where w is a column vector for which $w^T w = 1$ and $w w^T$ is an $N \times N$ symmetric matrix

i.e. $(w w^T)_{ij} = (w w^T)_{ji} = w_i w_j$

236

The Householder method

The Householder matrix, P , clearly has the following properties

P is symmetric (being the difference between 2 symmetric matrices, I and $2w w^T$)

P is orthogonal:

$$\begin{aligned} P^T P &= (I - 2 w w^T)^T (I - 2 w w^T) \\ &= (I - 4 w w^T + 4 \underbrace{w w^T w w^T}_{= I}) = I \end{aligned}$$

237

The Householder method

Let a_1 be the first column of A

$$\text{Let } w = \frac{a_1 - |a_1| e_1}{|a_1 - |a_1| e_1|}$$

where $e_1 = (1, 0, 0, 0, \dots, 0)^T$

238

The Householder method

Let's consider the action of $P = I - 2 w w^T$ upon the first column of A

$$P a_1 = \left\{ I - 2 \frac{(a_1 - |a_1| e_1)(a_1 - |a_1| e_1)^T}{|a_1 - |a_1| e_1|^2} \right\} a_1$$

$$= a_1 - 2 \frac{(a_1 - |a_1| e_1)(|a_1|^2 - |a_1| e_1^T a_1)}{|a_1|^2 - 2|a_1| e_1^T a_1 + |a_1|^2}$$

$$= |a_1| e_1$$

→ P zeroes out all but the first element of a_1

239

The Householder method

Suppose A is symmetric

$$A = \begin{pmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \end{pmatrix} \rightarrow P A = \begin{pmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{pmatrix}$$

$$\rightarrow M_1 = P A P^{-1} = P A P^T = \begin{pmatrix} x & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{pmatrix}$$

240

The Householder method

Suppose A is symmetric

$$A = \begin{pmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \end{pmatrix} \rightarrow PA = \begin{pmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{pmatrix}$$

$$\rightarrow M_1 = PAP^{-1} = PA P^T = \begin{pmatrix} x & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{pmatrix}$$

241

The Householder method

Next step, choose new Householder matrix of the form

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots \end{pmatrix} \text{ to operate on } M_1 = \begin{pmatrix} x & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{pmatrix}$$

$$\rightarrow M_2 = P M_1 P^{-1} = \begin{pmatrix} x & 0 & 0 & 0 & 0 & 0 \\ x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \end{pmatrix}$$

242

The Householder method

After N Householder transformations, the matrix is tridiagonal

$$M_N = \begin{pmatrix} x & x & 0 & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{pmatrix} = P_N \dots P_2 P_1 A P_1^{-1} P_2^{-1} \dots P_N^{-1}$$

243

The Householder method: computational cost

You might think that each Householder transformation would require $O(N^3)$ operations (2 matrix multiplications), so the whole series would cost $\sim O(N^4)$ operations

However, we can rewrite the product PAP^{-1} as follows

$$\begin{aligned} PAP^{-1} &= PAP = (I - 2ww^T)A(I - 2ww^T) \\ &= (I - 2ww^T)(A - 2Aw w^T) \\ &= (A - 2ww^T A - 2Aw w^T + 4ww^T A w w^T) \end{aligned}$$

244

The Householder method: computational cost

Now define an $N \times 1$ vector $v = Aw$

$$\begin{aligned} P^{-1}AP &= (A - 2ww^T A - 2Aw w^T + 4ww^T A w w^T) \\ &= (A - 2ww^T A - 2v w^T + 4w \underbrace{w^T v}_{\text{scalar}} w^T) \end{aligned}$$

This calculation involves an N^2 operation to compute v , and a series of N^2 operations to compute, subtract and add various matrices

The reduction of the matrix to tridiagonal form therefore costs only $O(N^3)$ operations, not $O(N^4)$

245

Diagonalization of a symmetric tridiagonal matrix (*Recipes*, §11.3)

- The diagonalization of a symmetric tridiagonal matrix can be accomplished by a series of $\sim N$ 2×2 rotations
- Consider the rotation

$$R = \begin{pmatrix} c & s & 0 & 0 & 0 & 0 & 0 \\ -s & c & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

(rotation through angle $\theta = \cos^{-1} C = \sin^{-1} S$ in the $x_1 - x_2$ plane)

246

Diagonalization of a symmetric tridiagonal matrix

- Applying this similarity transform to the tridiagonal matrix, we obtain

$$R^T \begin{pmatrix} t & u & 0 & 0 & 0 & 0 & 0 \\ u & v & x & 0 & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 & 0 \\ 0 & 0 & x & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & 0 & x & x \end{pmatrix} R = \begin{pmatrix} x & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 & 0 \\ 0 & 0 & x & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & 0 & x & x \end{pmatrix}$$

247

Diagonalization of a symmetric tridiagonal matrix

- To make the off-diagonal elements $(R^T M_N R)_{12} = (R^T M_N R)_{21}$ zero, we require

$$tSC + u(C^2 - S^2) - vSC = 0$$

$$\rightarrow \tan \theta = \frac{(t - v) \pm \sqrt{(t - v)^2 + 4u^2}}{2u}$$

248

Lecture 11: Diagonalization of complex Hermitian matrices (Recipes, §11.4)

- Decompose Hermitian matrix A into real and imaginary parts

$$A = B + iC$$

where B is symmetric & real and C is antisymmetric & real

- Decompose the eigenvectors in the same way

$$x = u + iv$$

249

Diagonalization of complex Hermitian matrices

- The eigenvalue equation becomes

$$(B + iC)(u + iv) = \lambda(u + iv)$$

$$\rightarrow Bu - Cv = \lambda u \quad (\text{real part})$$

$$Cu + Bv = \lambda v \quad (\text{imaginary part})$$

250

Diagonalization of complex Hermitian matrices

- This can be written

$$\begin{pmatrix} B & -C \\ C & B \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \end{pmatrix}$$

↑
2N x 2N
real symmetric
matrix

251

Diagonalization of complex Hermitian matrices

- Notes

1) If $\begin{pmatrix} u \\ v \end{pmatrix}$ is an eigenvector, so is $\begin{pmatrix} -v \\ u \end{pmatrix}$

→ N real eigenvalues, 2N doubly (at least) degenerate eigenvectors

2) Computation cost is 8x symmetric real case

→ faster to use technique that uses complex arithmetic on the N x N matrix

252

Diagonalization of non-symmetric matrices (*Recipes*, §11.5 – 11.6)

- Non-symmetric matrices present the most difficult case
- Effect of Householder transformations is to bring the matrix to “Hessenberg” form

$$P_N^{-1} \dots P_2^{-1} P_1^{-1} A P_1 P_2 \dots P_N = \begin{pmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{pmatrix}$$

253

Diagonalization of non-symmetric matrices

- Clearly, we can use Householder transformations on the left side of **A** to write any as matrix as the product of an orthogonal matrix and a triangular matrix

$$T = P_N \dots P_2 P_1 A$$

$$\rightarrow A = Q T$$

$$\text{where } Q^{-1} = Q^T = P_N \dots P_2 P_1$$

254

Diagonalization of non-symmetric matrices

- Amazing theorem:
If we iteratively apply a similarity transformation as follows
 $A_{s+1} = Q_s^{-1} A_s Q_s = T_s Q_s$
where Q_s^{-1} is the sequence of transformations needed to get A_s into triangular form
 $A_s = Q_s T_s \rightarrow T_s = Q_s^{-1} A_s$
...then the matrices A_s become more and more nearly diagonal

255

The QT algorithm

- Notes on the so-called QT algorithm
 - 1) The iterative procedure is $A_{s+1} - A_s = T_s Q_s - Q_s T_s$
 - 2) **T** is lower triangular, the eigenvalues appear in increasing order along the diagonal
 - 3) Convergence:
after s iterations, off-diagonal elements $(A_s)_{ij} \sim (\lambda_i/\lambda_j)^s$

256

The QT algorithm

- Notes on the so-called QT algorithm
 - 4) Computational cost

For general matrix: $O(N^3)$ per iteration
For Hessenberg matrix: $O(N^2)$ per iteration
For tridiagonal matrix: $O(N)$ per iteration

→ strategy: use Householder to get to Hessenberg form; then use QT

257

Eigenvalue shifting

Convergence can be slow for nearly degenerate eigenvalues

Trick: shift eigenvalues using this revised algorithm:

$$A_s - \kappa I = Q_s T_s$$

$$A_{s+1} = T_s Q_s + \kappa I$$

258

Eigenvalue shifting

Eigenvalues of $\mathbf{A}_s - \kappa \mathbf{I}$ follow from $\det(\mathbf{A}_s - \kappa \mathbf{I} - \lambda' \mathbf{I}) = 0 \rightarrow \lambda' = \lambda + \kappa$
 \rightarrow all eigenvalues shifted by κ

The revised algorithm converges according to $(\mathbf{A}_{s+1})_{ij} \sim (|\lambda_i - \kappa|/|\lambda_j - \kappa|) (\mathbf{A}_s)_{ij}$
 If κ is close to λ_i , this converges rapidly

Several algorithms shift eigenvalues to achieve better convergence; must remember to correct eigenvalues at the end

259

Roundoff error

Thus far, we have not considered the issue of roundoff error

Certain steps can be taken to minimize roundoff error

1) Prior to using the Householder method to diagonalize a symmetric real matrix, permute the matrix to make the smaller elements in the top left-hand corner.

2) Prior to using the QT algorithm to diagonalize a non-symmetric matrix, use similarity transformation to "balance" the matrix such that $\langle A_i^2 \rangle \sim \langle A_j^2 \rangle$

260

Iterative improvement of the solution to the symmetric eigenvalue problem

- Like the solution to $\mathbf{A} \mathbf{x} = \mathbf{b}$, the solution to $\mathbf{A} \mathbf{x} = \lambda \mathbf{x}$ (with \mathbf{A} real symmetric) can be improved by iterative improvement
- Suppose the "true" set of eigenvectors and eigenvalues is \mathbf{x}_i^* , λ_i^*

...and that we have obtained an approximate set \mathbf{x}_i , λ_i

261

Iterative improvement

- Let us suppose our approximate set of eigenvectors is related to the true set by the equation $\mathbf{x}_j = \sum_{i=1}^N c_{ij} \mathbf{x}_i^*$ where c_{ij} for $i \neq j$ is small relative to the c_{ji}
- Let's look for a better j th eigenvector $\mathbf{x}'_j = \sum_{i=1}^N c'_{ij} \mathbf{x}_i^*$

262

Iterative improvement

Claim: $(\mathbf{A} - \lambda_j \mathbf{I}) \mathbf{x}'_j = \mathbf{x}_j$ yields a better estimate of the j th eigenvector

This prescription yields

$$(\mathbf{A} - \lambda_j \mathbf{I}) \sum_{i=1}^N c'_{ij} \mathbf{x}_i^* = \sum_{i=1}^N c_{ij} \mathbf{x}_i^*$$

$$\rightarrow \sum_{i=1}^N c'_{ij} (\lambda_i^* - \lambda_j) \mathbf{x}_i^* = \sum_{i=1}^N c_{ij} \mathbf{x}_i^*$$

$$\rightarrow c'_{ij} (\lambda_i^* - \lambda_j) = c_{ij}$$

Since the \mathbf{x}_i^* are orthogonal

263

Iterative improvement

This procedure yields

$$c'_{ij} = c_{ij} (\lambda_i^* - \lambda_j)^{-1}$$

If the eigenvalues are close to the correct values, this **amplifies** the $i=j$ components of c'_{ij} , meaning that the new set of eigenvectors, \mathbf{x}'_j is closer to the true set, \mathbf{x}_i^*

264

Iterative improvement

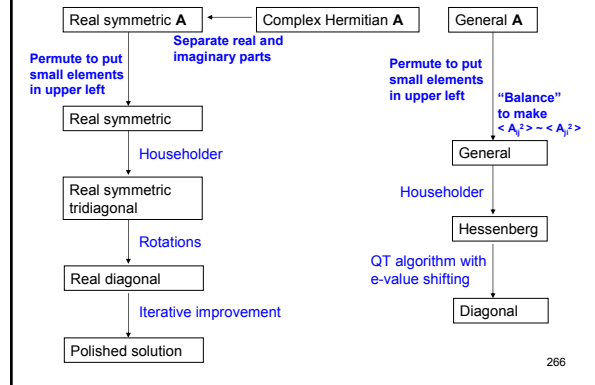
The next estimate of the eigenvalues is then obtained from

$$\lambda_j' = \mathbf{x}_j'^T \mathbf{A} \mathbf{x}_j'$$

This iteration converges quickly, and cannot be carried out too many times before $(\mathbf{A} - \lambda_j \mathbf{I})$ becomes singular (or extremely poorly conditioned)

265

Summary



266

Lecture 12: Fourier transforms (Numerical Recipes, Chapter 12)

- Fourier transforms are important in a huge variety of physics applications: optics, quantum mechanics, classical mechanics ...

Definitions:

Let $h(t)$ be a (possibly complex) function of time defined for $-\infty < t < \infty$

The Fourier transform is $H(f) = \int_{-\infty}^{\infty} h(t) e^{i2\pi ft} dt$
and

the inverse transform is $h(t) = \int_{-\infty}^{\infty} H(f) e^{-i2\pi ft} df$

267

Spectral power density

The *power* associated with the function $h(t)$ is defined by

$$P_{\text{TOT}} = \int_{-\infty}^{\infty} |h(t)|^2 dt = \int_{-\infty}^{\infty} dt h(t) h^*(t)$$

$$\begin{aligned} \text{and is equal to } & \int_{-\infty}^{\infty} dt \int_{-\infty}^{\infty} df H(f) e^{-i2\pi ft} \int_{-\infty}^{\infty} df' H^*(f') e^{i2\pi f't} \\ & = \int_{-\infty}^{\infty} df \int_{-\infty}^{\infty} df' H(f) H^*(f') \underbrace{\int_{-\infty}^{\infty} dt e^{-i2\pi(f-f)t}}_{\delta(f-f)} \\ & = \int_{-\infty}^{\infty} |H(f)|^2 df \end{aligned}$$

268

Spectral power density

The relation $P_{\text{TOT}} = \int_{-\infty}^{\infty} |H(f)|^2 df$

allows us define the spectral power density by

$$P(f) = |H(f)|^2 + |H(-f)|^2 \text{ for } f \geq 0$$

such that $P_{\text{TOT}} = \int_0^{\infty} P(f) df$

269

Spectral power density

Physical example: $h(t)$ = electric field in an EM wave

$$\text{Total flux} = (c/4\pi) |h(t)|^2$$

$$\begin{aligned} \text{Monochromatic flux at frequency } f \\ = P(f) = (c/2\pi) |H(f)|^2 \end{aligned}$$

Note: Whenever $h(t)$ is real, $H(f) = H^*(-f)$ and $P(f) = 2 |H(f)|^2$

270

Measurements of h(t)

In real measurements of h(t), we never measure h(t) over an infinite time period. Nor do we measure with an infinite density of points. We tend to SAMPLE h(t) on a discrete set of times

i.e. we obtain $h_k = h(k\Delta t)$

where $k = 0, 1, 2, 3, \dots$
and Δt is the "sampling interval"

271

Nyquist critical frequency

There is a special frequency associated with the sampling frequency called the Nyquist critical frequency:

$$f_c = \frac{1}{2} \Delta t^{-1}$$

This is the maximum frequency that we can study from our measurements

272

Sampling theorem

If $H(f) = 0$ for $|f| > f_c = \frac{1}{2} \Delta t^{-1}$

then h(t) is completely determined by the discrete samples:

$$h(t) = \Delta t \sum_{k=-N}^N h_k \frac{\sin [2\pi f_c(t - k\Delta t)]}{[\pi(t - k\Delta t)]}$$

Need at least 2 samples per cycle of a sine wave at the Nyquist frequency to determine its amplitude

273

Sampling theorem

However, if $H(f) \neq 0$ for $|f| > f_c$

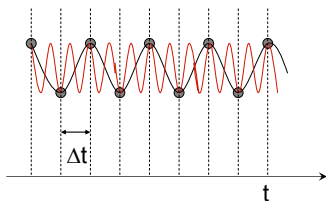
then we cannot compute a correct H(f) or P(f) even for $f < f_c$

Higher frequencies will "feed down" into the measured range: this is called ALIASING

274

Aliasing

Example: it is impossible to distinguish a signal at $f = f_c$ from one at $f = 3f_c$



275

Aliasing

- Consider the superposition of two waves at frequencies f_1 and $f_2 = f_1 + m/\Delta t$, with m any integer

$$h(t) = a_1 \exp(i2\pi f_1 t) + a_2 \exp(i2\pi f_2 t)$$

$$h_k = h(k\Delta t) = a_1 \exp(i2\pi f_1 k\Delta t) + a_2 \exp(i2\pi [f_1 k\Delta t + km])$$

$$= [a_1 + a_2 \exp(2\pi i m k)] \exp(2\pi i f_1 k\Delta t)$$

$$= [a_1 + a_2] \exp(2\pi i f_1 k\Delta t)$$

276

The discrete Fourier transform

- Real measurements always sample a function at a finite number of points N over a finite interval T
 $h_k = h(k\Delta t)$, with $k = 0, 1, 2, 3, \dots, N-1$
 (We'll assume N to be even)
- With N data points, we can only determine $H(f)$ at N frequencies in the interval $-f_c < f < f_c$

 \rightarrow suggests we should consider only the finite set of frequencies, $f_n = n / N\Delta t$
 with $n = -\frac{1}{2}N, -\frac{1}{2}N+1, \dots, -1, 0, 1, \dots, +\frac{1}{2}N$

277

The discrete Fourier transform

The Fourier transform at these frequencies is

$$H(f_n) = \int_{-\infty}^{\infty} h(t) \exp(i2\pi f_n t) dt$$

$$\sim \sum_{k=0}^{N-1} h_k \exp(i2\pi f_n k\Delta t) \Delta t$$

$$= \sum_{k=0}^{N-1} h_k \exp(i2\pi nk/N) \Delta t \equiv H_n \Delta t$$

278

The discrete Fourier transform

$$H_n = \sum_{k=0}^{N-1} h_k \exp(i2\pi nk/N)$$

Notes:

- H_n is periodic in n with a period N
 (since $\exp i2\pi k = 1$ for any integer k)
 $\rightarrow H_n = H_{N+n}$ or $H_{-n} = H_{N-n}$

This allows us to renumber with n from 0 to $N-1$ instead of from $-N/2$ to $+N/2$

279

The discrete Fourier transform

The inverse transform is

$$h_k = \frac{1}{N} \sum_{n=0}^{N-1} H_n \exp(-i2\pi nk/N)$$

where we have made use of the fact that

$$\sum_{n=0}^{N-1} \exp(-i2\pi n([k - k']/N)) = N \delta_{k,k'}$$

280

The discrete Fourier transform

The total power is defined as

$$P_{\text{TOT}} = \sum_{k=0}^{N-1} |h_k|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |H_n|^2$$

281

The discrete Fourier transform

- The calculation of a discrete FT is essentially the multiplication of a vector by a matrix

$$H_n = \sum_{k=0}^{N-1} h_k \exp(i2\pi nk/N) = \sum_{k=0}^{N-1} W^{nk} h_k$$

where $W = e^{2\pi i/N}$

\rightarrow requires $\sim N^2$ operations

282

The Fast Fourier Transform

- The direct calculation costing $\sim N^2$ operations is probably the easiest approach for $N < 1000$
- It is not the fastest way, however.
- It was realized in the 1960's that there are much more efficient ways of the doing the calculation: we describe next a one such method

283

Danielson-Lanczos Lemma

$$\begin{aligned}
 \text{Write } H_n &= \sum_{k=0}^{N-1} h_k \exp(i2\pi nk/N) \\
 &= \sum_{k=0}^{[N/2]-1} h_{2k} \exp[i2\pi(2k)n/N] + \sum_{k=0}^{[N/2]-1} h_{2k+1} \exp[i2\pi(2k+1)n/N] \\
 &= \sum_{k=0}^{[N/2]-1} h_{2k} \exp[i2\pi kn/(N/2)] + W^n \sum_{k=0}^{[N/2]-1} h_{2k+1} \exp[i2\pi kn/(N/2)] \\
 &= H_n^e + W^n H_n^o
 \end{aligned}$$

$W = \exp(2\pi i/N)$ as before

284

The Fast Fourier Transform

- Key point: H_n^o and H_n^e are sampled half as densely as H_n
 - There are only $N/2$ sample points and the Nyquist frequency is only one-half that for H_n
 - H_n^o and H_n^e therefore repeat after the first $N/2$ n -values \rightarrow only need to calculate them for the first $N/2 - 1$ terms
 - The we gain a factor of 2 in computation speed
- Of course, we can do this again, and further subdivide H_n^o and H_n^e

285

The Fast Fourier Transform

$$\begin{aligned}
 \text{Write } H_n^e &= \sum_{k=0}^{[N/2]-1} h_{2k} \exp[i2\pi kn/(N/2)] \\
 &= \sum_{k=0}^{[N/4]-1} h_{4k} \exp[i2\pi(2k)n/(N/2)] + \sum_{k=0}^{[N/4]-1} h_{4k+1} \exp[i2\pi(2k+1)n/(N/2)] \\
 &= \sum_{k=0}^{[N/4]-1} h_{4k} \exp[i2\pi kn/(N/4)] + W^{2n} \sum_{k=0}^{[N/4]-1} h_{4k+1} \exp[i2\pi kn/(N/4)] \\
 &= H_n^{ee} + W^{2n} H_n^{eo}
 \end{aligned}$$

$W = \exp(2\pi i/N)$ as before

286

The Fast Fourier Transform

- We have now obtained a factor of \sim four improvement in speed
- $H_n = H_n^{ee} + W^{2n} H_n^{eo} + W^n H_n^{oe} + W^{3n} H_n^{oo}$
- Four items, each with $N/4$ samples,
 - \rightarrow Computational cost $\sim 4 \times (N/4)^2 \sim N^2/4$

287

The Fast Fourier Transform

$$H_n = H_n^{ee} + W^{2n} H_n^{eo} + W^n H_n^{oe} + W^{3n} H_n^{oo}$$

\uparrow
 Involves $h_0, h_4, h_8 \dots$
 \uparrow
 Involves $h_2, h_6, h_{10} \dots$
 \uparrow
 Involves $h_1, h_5, h_9 \dots$
 \uparrow
 Involves $h_3, h_7, h_{11} \dots$

288

The Fast Fourier Transform

Term	Rewrite as	k-values	in binary	Bit-reversed
H_n^{ee}	H_n^{00}	0,4,8 ...	0000, 0100, 1000, 1100 ...	0000, 0010, 0001, 0011
H_n^{eo}	H_n^{01}	2,6,10,...	0010, 0110, 1010, 1110 ...	0100, 0110, 0101, 0111
H_n^{oe}	H_n^{10}	1,5,9,...	0001, 0101, 1001, 1101 ...	1000, 1010, 1001, 1011
H_n^{oo}	H_n^{11}	3,7,11,...	0011, 0111, 1011, 1111 ...	1100, 1110, 1101, 1111 ...

1 for odd, 0 for even

289

The Fast Fourier Transform

- This suggests a very clever scheme
 - Start with $N = 2^m$
 - Do the subdivision m times, until each term contains a *single* sample
- e.g. $H_n^{eeoeoeoeoe} = H_n^{001010010} = h_k$
with $k = 010010100_2 = 148$ in this example

290

The Fast Fourier Transform

- Resort terms by bit reversed k values
- Combine adjacent pairs into two point functions
- Combine adjacent pairs of pairs into 4 point functions
- Combine 4 point functions into 8 point functions
- Each step involves $\sim N$ operations, and there are $\sim \log_2 N$ steps
- Total cost $\sim N \log_2 N$

291

The Fast Fourier Transform

- This can be a huge improvement over an $O(N^2)$ algorithm
 - For $N = 10^4 \rightarrow 750$ times faster
 - For $N = 10^6 \rightarrow 50,000$ times faster
- For $h(t)$ real, a further factor ~ 2 improvement results because $H^*(f) = H(-f)$

292

Lecture 13: Applications of Fourier transforms (Recipes, Chapter 13)

There are many applications of FT, some of which involve the convolution theorem (Recipes §13.1):

The convolution of $h(t)$ and $r(t)$ is defined by

$$s(t) = h * r = \int_{-\infty}^{\infty} dt' h(t') r(t - t') = \int_{-\infty}^{\infty} dt' h(t') r(t - t')$$

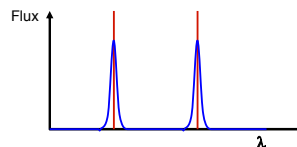
It is trivial to show that

$$S(f) = \int_{-\infty}^{\infty} [h * r](t) \exp(2\pi i f t) dt = H(f) R(f)$$

293

Instrumental response function

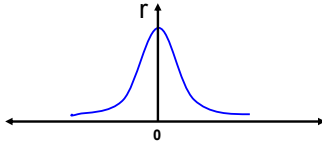
- A classic example of a convolution results from the finite resolution of a measuring instrument.
- Example: a spectrograph has finite wavelength resolution, which smears the observed spectrum: even a delta function is smeared out to a feature of finite width



294

Instrumental response function

- The instrumental response is characterized by a resolution function r , which describes how an intrinsically very narrow function will be broadened
- The function r is usually peaked at zero



295

Convolution of a signal with a resolution function

- Suppose we have discretely sampled our resolution function $r(t)$ and our signal $h(t)$ to obtain r_k and h_k
- Suppose h is periodic and has been sampled at N points: $k = 0, N-1$
- r is usually sampled (almost) symmetrically about 0:
 $r_k = r(k\Delta t)$ with $k = -\frac{1}{2}N+1, \dots, 0, \dots, \frac{1}{2}N$

296

Convolution of a signal with a resolution function

- The convolution can then be written

$$(h*r)_k = \sum_{j=1-N/2}^{N/2} r_j h_{k-j}$$

To avoid aliasing, r_j must also be periodic with period $N \rightarrow r_j = r_{j+N}$

so we can shift the indexing from

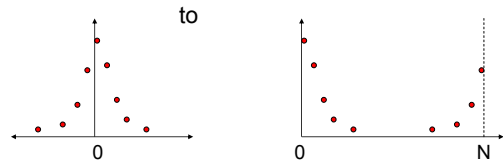
$$k = -\frac{1}{2}N+1, \dots, 0, \dots, \frac{1}{2}N$$

to $k = 0$ to $N-1$

297

Convolution of a signal with a resolution function

- The reordering changes the r_k from



and the convolution becomes

$$(h*r)_k = \sum_{j=0}^{N-1} r_j h_{k-j} \quad 298$$

Convolution of a signal with a resolution function

The convolution theorem tells us that the discrete Fourier transform of

$$s_k = (h*r)_k \text{ is } S_n = H_n R_n$$

suggesting that we can compute the convolution by obtaining the discrete FTs of h and r , multiplying them, and taking the inverse Fourier transform of the product

299

Computational expense of convolution

Thanks to the Fast Fourier Transform (FFT), this can save a lot of computational expense

Direct calculation of $(h*r)_k = \sum_{j=0}^{N-1} r_j h_{k-j}$ takes $2N^2$ operations

300

Computational expense of convolution

- Calculation with FFT method takes

$N \log_2 N$ operations for H_n
 $N \log_2 N$ operations for R_n
 N operations to multiply H_n and R_n
 $N \log_2 N$ operations to take the inverse FT of S_n
 for a total of $N(1+3 \log_2 N)$ operations.....

301

Computational expense of convolution

... which can yield a huge savings

e.g. for $N = 10^5$: $2N^2 = 2 \times 10^{10}$
 $N(1+3 \log_2 N) = 5 \times 10^6$

(although note that if the resolution function is narrowly peaked, r_k may have only M non zero values, where $M \ll N$, and the direct calculation only costs $2NM$)

302

Deconvolution

- A key application, of course, is the deconvolution:
 i.e. given the measurements s_k , and knowing the resolution function r_k , we estimate the signal h_k
 A direct solution of $s_k = \sum_{j=0}^{k-1} r_j h_{k-j}$, given s and h , requires $O(N^3)$ operations
 (or $O(N^2M)$ in the limit where r_k has only M non-zero elements)

303

Deconvolution

- Using FFT, the procedure entails

Compute S_n : $N \log_2 N$ operations
 Compute R_n : $N \log_2 N$ operations
 Compute $H_n = S_n/R_n$: N operations
 Inverse FT of H_n : $N \log_2 N$ operations

again for a total of $N(1+3 \log_2 N)$ operations.....

304

Problems with deconvolution

Two key issues arise:

1) R_n (and S_n) may be zero (or very small) at some frequencies so that H_n is undefined (smearing leads to unavoidable loss of information at high frequencies)

2) Noise can enter and spoil the precision

305

Lecture 14: Effect of noise

- Instead of measuring a perfect convolved signal, $s(t) = h*r$, we actually measure $c(t) = s(t) + n(t)$, where $n(t)$ is the noise at time t

How do we extract a good estimate of h in the presence of noise?

306

Filtering

(Recipes §13.2)

- We can attempt to correct for the presence of noise by multiplying $C(f)$ by a filter function, $\Phi(f)$, which we will take as real:
i.e. we write our best estimate of $H(f)$

$$\tilde{H}(f) = C(f)\Phi(f) / R(f)$$

[instead of $H(f) = S(f) / R(f)$]

Q: what filter function yields the best estimate of $H(f)$?

307

Filtering

- The optimal filter function is chosen to minimize the mean square deviation

$$\begin{aligned} P_{MSD} &= \int_{-\infty}^{\infty} |h(t) - \tilde{h}(t)|^2 dt = \int_{-\infty}^{\infty} |H(f) - \tilde{H}(f)|^2 df \\ &= \int_{-\infty}^{\infty} \left| \frac{S(f)}{R(f)} - \frac{\Phi(f)[S(f) + N(f)]}{R(f)} \right|^2 df \end{aligned}$$

308

Filtering

This becomes

$$P_{MSD} = \int_{-\infty}^{\infty} \frac{[1 - \Phi(f)]^2 |S(f)|^2 + |\Phi(f)|^2 |N(f)|^2}{|R(f)|^2} df$$

plus terms involving $N^*(f)S(f)$, which average to zero if the noise is uncorrelated with the signal

309

Wiener filtering

- The mean square deviation is minimized for

$$\begin{aligned} 0 &= dP_{MSD} / d\Phi = \int_{-\infty}^{\infty} \frac{-2[1 - \Phi(f)] |S(f)|^2 + 2\Phi(f) |N(f)|^2}{|R(f)|^2} df \\ \Rightarrow \Phi(f) &= \frac{|S(f)|^2}{|S(f)|^2 + |N(f)|^2} \end{aligned}$$

310

Wiener filtering

- Of course, $S(f)$ is not directly measurable
- All we measure is $c(t) = s(t) + n(t)$, for which the Fourier transform is $C(f)$
- If the signal and the noise are uncorrelated, the power spectrum of $c(t)$ is

$$|C(f)|^2 = |S(f) + N(f)|^2 = |S(f)|^2 + |N(f)|^2$$

311

Power spectrum of $c(t)$

- The power spectrum of the real data then looks like this:

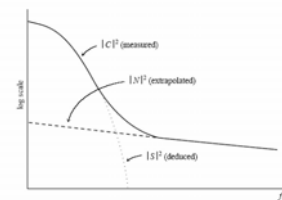


Figure 13.1. Optimal (Wiener) filtering. The power spectrum of signal plus noise shows a signal peak added to a noise floor. The floor is extrapolated back across the signal region as a "noise model". Interpolating gives the "signal model". The models need not be accurate for the method to be useful. A simple algebraic combination of the models gives the optimal filter (see text).

312

Lecture 16: Ordinary differential equations
(*Recipes*, Chapter 16)

- One of the biggest topics in numerical analysis for physicists
 - Much of physics involves solving ODEs, often with non linear coefficients
 - There are many techniques
 - They have different strengths
 - We will give an overview so that you can choose the most appropriate method

313

First-order coupled ODEs

- Note that *any* nth order ODE can be recast as a coupled set of first order ODEs:

$$f_n(x, y, y', y'' \dots) \frac{d^n y}{dx^n} + f_{n-1}(x, y, y', y'' \dots) \frac{d^{n-1} y}{dx^{n-1}} + \dots$$

$$\dots + f_1(x, y, y', y'' \dots) \frac{dy}{dx} + f_0(x, y, y', y'' \dots) y = 0$$

314

First-order coupled ODEs

- Define (for example) $y_n \equiv \frac{d^n y}{dx^n}$

to obtain $\frac{dy_0}{dx} = y_1$

$$\frac{dy_1}{dx} = y_2$$

.....

$$\frac{dy_{n-2}}{dx} = y_{n-1}$$

$$\frac{dy_{n-1}}{dx} = -f_{n-2}y_{n-2} - f_{n-3}y_{n-3} - \dots - f_0 y_0$$

315

First-order coupled ODEs

- There are (infinitely) many different ways of doing this, and a clever choice can be sometimes be used to eliminate singularities and other “bad behavior”
- In order to obtain a solution, we always require n boundary conditions

316

Boundary conditions

- Initial value problem: all conditions specified at the same “initial” x-value (call it x_0):
 $y_i(x_0) = y_{i0}$
- Two-point boundary value problems: conditions specified at different x-values (e.g. $y_i(x_1) = y_{i1}$, $y_i(x_2) = y_{i2}$, $y_j(x_3) = y_{j3}$ )
Will discuss these later

317

Euler’s method

- The set of n coupled first order ODEs is a (generally) straightforward extension of the n=1 case, so let’s consider the 1-D case for simplicity

$$dy/dx = f(x,y)$$

Suppose we know y at some location x:

We can then estimate y at position x + h using $y(x+h) = y(x) + hf(x,y)$

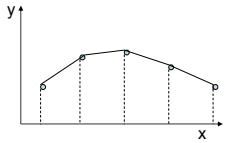
318

Euler's method

- This leads to a simple stepping scheme called Euler's method:

$$\text{If } x_{n+1} = x_n + h$$

$$\text{then } y_{n+1} = y_n + h f(x_n, y_n)$$



319

Euler's method

- This method is of little practical interest because it
 - requires very small values of h
 - is very slow
 - is fairly inaccurate
 - can easily lead to instabilities

320

Euler's method

- Example of instability

$$\text{Consider } dy/dx = -\alpha y \rightarrow y = ae^{-\alpha x}$$

$$\text{Euler's method: } y_{n+1} = y_n + h(-\alpha y_n) = y_n(1 - \alpha h)$$

becomes unstable when $h > 1/\alpha$ (changes sign at each step)

grows without bound when $h > 2/\alpha$

Return to this later

321

Euler's method

- Euler's method clearly isn't very accurate
We use only the first term in the Taylor expansion:

$$y(x+h) = y(x) + hy'(x) + h^2y''(x)/2 + h^3y'''(x)/6 + \dots$$

Error is of order h^2

322

Runga-Kutta

- We can improve the accuracy by estimating $y''(x)$ as follows:

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

$$= \frac{f(x+\frac{1}{2}h, [y(x) + \frac{1}{2}hf(x, y)]) - f(x, y)}{\frac{1}{2}h}$$

323

Runga-Kutta

- Substituting into the Taylor expansion, this yields:

$$y(x+h) = y(x) + hf(x+\frac{1}{2}h, y(x+\frac{1}{2}h)) + O(h^3)$$

In effect, we estimate the derivative at the midpoint of the interval – not the beginning

324

Runga-Kutta

- In effect, we gain a factor of h in the accuracy at the expense of an additional evaluation of f

– 2nd order Runga Kutta

- We can continue doing this, estimating

$$y'''(x) = \frac{(f_h - f_{1/2h})/1/2h - (f_{1/2h} - f_0)/1/2h}{1/2h}$$

...etc

325

4th order Runga-Kutta

- Classic and widely used extension is accurate to $O(h^4)$, requires 4 function evaluations per step, and can be written elegantly in the form

$$\Delta y_1 = hf(x, y)$$

$$\Delta y_2 = hf(x + 1/2h, y + 1/2\Delta y_1)$$

$$\Delta y_3 = hf(x + 1/2h, y + 1/2\Delta y_2)$$

$$\Delta y_4 = hf(x+h, y+\Delta y_3)$$

$$y(x+h) = y(x) + \Delta y_1/6 + \Delta y_2/3 + \Delta y_3/3 + \Delta y_4/6$$

326

4th order Runga-Kutta

- 4th order Runga-Kutta is
 - VERY robust (works well even when y isn't smooth)
 - fairly efficient (faster than Euler)
 - fairly accurate
- Q: how do we determine the required step size h to yield the desired accuracy?

327

Accuracy

- First, have to decide how we will define accuracy: various possibilities

$$\Delta \equiv |y^{\text{calc}}(x+h) - y^{\text{real}}(x+h)|$$

error in a single step

$$\Delta \equiv |y^{\text{calc}}(x+h) - y^{\text{real}}(x+h)| / |y^{\text{real}}(x+h)|$$

fractional error in a single step

$$\Delta \equiv |y^{\text{calc}}(L) - y^{\text{real}}(L)|$$

total error at endpoint (if we don't care about the trajectory)

328

Accuracy

- Of course, whatever definition we use, we still have the problem of not knowing y^{real}
- Q: so how can we estimate Δ ?
- A: use two different methods and assume that $\Delta \sim$ the difference

329

Runga-Kutta with adaptive step size

- Fehlberg found a formula involving 6 function evaluations that could be used to give both a 4th and a 5th order formula:
- Compare the two to obtain an estimate of the truncation error
- Adjust h to achieve desired Δ
 - adaptive step size which varies with x

330

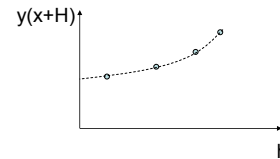
Bulirsch-Stoer

- If we have a smoothly-varying function, and we only want to know y at the endpoint, $x+H$, Runga-Kutta-Fehlberg is not the fastest method
- B-S goes from $y(x)$ to $y(x+H)$ (where H is a big step) via a Runga-Kutta like technique in which H is divided into successively smaller $h = H/n$

331

Bulirsch-Stoer

- Then plot $y(x+H)$ vs h and extrapolate to $h=0$



(Recall Romberg Integration)

332

Bulirsch-Stoer

- *Recipes* §16.4 describes the technical details:
 - How to choose the sequence of n
 - How to propagate the function
 - How to extrapolate to $h = 0$

333

Lecture 15: Unstable ODEs

- So far, we have discussed speed and accuracy, but not stability
 - Some problems (e.g. $dy/dx = -\alpha y$) apparently require very small steps to remain stable
 - A modification to Euler's method helps preserve stability
- Take $y_{n+1} = y_n + h(-\alpha y_{n+1})$
instead of $y_n + h(-\alpha y_n)$

334

Unstable ODEs

- This modification then yields $y_{n+1} = y_n / (1 + \alpha h)$ which is stable for all h
- The general approach of using $y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$ in place of $y_n + hf(x_n, y_n)$ is called an *implicit* Euler scheme

335

Generalization to N-eqns

- So far, we have only considered the case of a single equation
- Start by considering the generalization of $y' = f(x, y) = -\alpha y$ to N dimensions
- N coupled equations:
 $\mathbf{y}' = f(x, \mathbf{y}) = -\mathbf{C} \mathbf{y}$
where \mathbf{C} is a positive definite $N \times N$ matrix

336

Generalization to N-eqns

- Euler's method then gives

$$\mathbf{y}_{n+1} = \mathbf{y}_n - h \mathbf{C} \mathbf{y}_n = (\mathbf{I} - h \mathbf{C}) \mathbf{y}_n$$

which becomes unstable when $h > 1/C_{\max}$,
where C_{\max} is the largest eigenvalue

Problematic when $C_{\max} \gg C_{\min}$ ("stiff equations")
since a huge number of steps may be required
to cover the region of interest

337

Generalization to N-eqns

- Given N equations, the modification to Euler's method becomes

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(x+h, \mathbf{y}_{n+1}) = \mathbf{y}_n - h \mathbf{C} \mathbf{y}_{n+1}$$

$$\rightarrow \mathbf{y}_{n+1} = (\mathbf{I} + h \mathbf{C})^{-1} \mathbf{y}_n$$

(direct analogy to single equation case)

338

Generalization to N-eqns

- What happens when \mathbf{f} is not a linear function?

$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(x_{n+1}, \mathbf{y}_{n+1})$ may be
hard/impossible to solve exactly

So we approximate $\mathbf{f}(x_{n+1}, \mathbf{y}_{n+1})$ by

$$\mathbf{f}(x_n, \mathbf{y}_n) + \mathbf{J} (\mathbf{y}_{n+1} - \mathbf{y}_n),$$

where $J_{ij} = \partial f_i / \partial y_j$ is the Jacobian

339

Generalization to N-eqns

- The equation becomes

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(x_n, \mathbf{y}_n) + h\mathbf{J}(\mathbf{y}_{n+1} - \mathbf{y}_n),$$

$$\rightarrow (\mathbf{I} - h\mathbf{J}) \mathbf{y}_{n+1} = (\mathbf{I} - h\mathbf{J}) \mathbf{y}_n + h \mathbf{f}(x_n, \mathbf{y}_n)$$

$$\rightarrow \mathbf{y}_{n+1} = \mathbf{y}_n + h (\mathbf{I} - h\mathbf{J})^{-1} \mathbf{f}(x_n, \mathbf{y}_n)$$

This is called the *semi-implicit* Euler method

340

Generalization to N-eqns

- The method is *semi-implicit* because \mathbf{J} is
calculated at (x_n, \mathbf{y}_n)

- requires \mathbf{J} to be computed at each iteration
- usually stable
- allows much larger step sizes than are
possible in the explicit Euler method
- accuracy may be poor, especially for y_i
corresponding to large eigenvalues

341

First order ODEs: summary of methods

- Euler's method:**
 - error $\sim h^2$, of no practical interest
 - poor stability
- Implicit Euler's method**
 - error $\sim h^2$
 - good stability
- Fourth-order Runge-Kutta** (with adaptive step size
controlled by 5th order estimate)
 - good accuracy: error $\sim h^5$
 - good stability
- Bulirsch-Stoer**
 - excellent accuracy and speed
 - less robust than R-K
 - preferred method for single large step of smooth function

342

Predictor-corrector methods

- Like Bulirsch-Stoer, P-5 methods are another alternative to Runge-Kutta that are useful when the solution is smooth
 - They are MORE accurate but LESS robust
- Basic idea: if $y(x)$ is smooth, it can be extrapolated from a series of already calculated values:
 - Given $y_{n-3}, y_{n-2}, y_{n-1}$, and y_n , we obtain an estimate of y_{n+1} ("predictor" step), which we then "correct" with reference to the derivative at y_{n+1}

343

Milne's method

1. Start with 4 "known" points (use Runge-Kutta to "seed" the technique):
 $(x_{n-3}, y_{n-3}), (x_{n-2}, y_{n-2}), (x_{n-1}, y_{n-1}), (x_n, y_n)$
2. Solve for the 3rd order polynomial that fits the derivatives: $(x_{n-3}, y'_{n-3}) \dots (x_n, y'_n)$
 (where $y'_{n-3} = f(x_{n-3}, y_{n-3}) \dots$ etc.)
 to obtain $y'(x) = a + bx + cx^2 + dx^3$

344

Milne's method

3. Integrate the polynomial analytically to obtain an estimate of y_{n+1}

$$y_{n+1} - y_{n-3} = \int_{x_{n-3}}^{x_{n+1}} y'(x) dx = \frac{4h}{3} (2y'_n - y'_{n-1} + 2y'_{n-2})$$

4. We can now obtain an estimate of the slope $y'_{n+1} = f(x_{n+1}, y_{n+1})$

345

Milne's method

5. We can now improve our estimate of y_{n+1}

$$y_{n+1}^* - y_{n-1} = \int_{x_{n-1}}^{x_{n+1}} y'(x) dx = \frac{h}{3} (y'_{n-1} + 4y'_n + y'_{n+1})$$

Simpson's rule

6. Save the point (x_{n+1}, y_{n+1}^*) and iterate on 4 & 5

346

Milne's method

- Like 4th order R-K, the error is of order h^5 , but
 - the coefficient is much smaller (better accuracy)
 - the robustness is worse (blows up if $y(x)$ isn't smooth enough)
- There are many other varieties of Predictor-Corrector technique

347

Two point boundary problems (Recipes, Chapter 17)

- So far, we have discussed the initial value problem exclusively
 Here, we were given all the y_i at $x = 0$
- Next consider the two point boundary problem, in which
 - n_1 boundary conditions are given at $x = 0$
 - n_2 boundary conditions are given at $x = L$
 (with $n_1 + n_2 = N$)

348

Two point boundary problems

- Two point boundary problems are much more expensive to solve.
- Two general methods
 - “shooting methods”, in which we guess the missing boundary conditions at $x = 0$ and see how things end up at $x = L$
 - “relaxation methods”, in which we guess the initial trajectory and let the intermediate points all relax to values that satisfy our ODEs

349

Shooting methods

1. Start with a guess for the initial b.c.'s: n_1 of the elements of $\mathbf{y}(0)$ are known, and $n_2 = N - n_1$ are guesses
Let \mathbf{b} be the vector consisting of these n_2 guesses
2. Solve the initial value problem, i.e. shoot \mathbf{y} from $x = 0$ to $x = L$

350

Shooting methods

3. Let \mathbf{y}^{calc} be the n_2 components at the $x = L$ boundary
4. Determine the error in the n_2 b.c.'s at $x = L$
 $\mathbf{e} = \mathbf{y}^{\text{calc}} - \mathbf{B}$,
where \mathbf{B} are the desired boundary conditions at $x = L$
5. Iterate on \mathbf{b} to obtain $\mathbf{e} = \mathbf{0}$

351

Shooting methods

- Use Newton-Raphson to solve $\mathbf{e}(\mathbf{b}) = \mathbf{0}$

→ we iterate using $\mathbf{b}_{n+1} = \mathbf{b}_n - \mathbf{J}^{-1}\mathbf{e}(\mathbf{b}_n)$

Computing the Jacobian $\mathbf{J}_{ij} = \partial e_i / \partial b_j$ requires us to integrate the ODEs $n_2 + 1$ times (once for the initial guess and once varying each element of \mathbf{b})

352

Computational cost

- Shooting methods require $n_2 + 1$ ODE integrations per Newton-Raphson step
 - much more expensive
- Obviously, if $n_1 < n_2$, we start at the $x = L$ boundary and shoot backwards

353

Lecture 16: Relaxation methods

- Clever technique which begins with a first guess of the trajectory across the entire interval
 - Break the interval into M small steps:
 $x_1=0, x_2, \dots, x_M=L$
 - Form a grid of points, $y_{i,k}$
 $i = 1, N \quad k = 1, M \quad \rightarrow \quad N \times M \text{ matrix}$

354

Relaxation methods

- Now convert the system of equations $\mathbf{y}' = \mathbf{f}(x, \mathbf{y})$

into a set of difference equations

$$y_{i,k+1} - y_{i,k} = (x_{k+1} - x_k) f_i(\frac{1}{2}[x_{k+1} + x_k], (\frac{1}{2}[y_{k+1} + y_k]))$$

$$\rightarrow 0 = E_{i,k} \equiv y_{i,k+1} - y_{i,k} - (x_{k+1} - x_k) f_{i,k}$$

Set of $N(M-1)$ equations with $N(M-1)$ unknowns (there are N b.c.'s)

355

Relaxation methods

- Can solve this problem iteratively by multidimensional Newton-Raphson, using

$$0 = E_{i,k}(y_{i,k+1} + \Delta y_{i,k+1}, y_{i,k} + \Delta y_{i,k}) \\ = E_{i,k}(y_{i,k+1}, y_{i,k}) + \Delta y_{i,k+1} \partial E_{i,k} / \partial y_{i,k+1} + \Delta y_{i,k} \partial E_{i,k} / \partial y_{i,k}$$

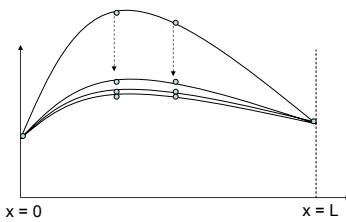
to solve for the required $\Delta y_{i,k}$

For each iteration, we need to solve a $MN \times MN$ linear system to determine the required correction to $y_{i,k}$ (but the system is fortunately a sparse one)

356

Relaxation methods

- The procedure is repeated until the trajectory relaxes to the correct one



357

Relaxation methods

- Grid size is critical \rightarrow put more points where the gradient is largest
 - If not known a priori, shooting methods may be better
 - Relaxation methods work best for smooth functions (use shooting methods for oscillatory functions)
 - Relaxation methods are favored when shooting methods are very unstable (e.g. in presence of a growing exponential solution)

358

Partial differential equations

(Recipes, Chapter 19)

- The Physical Universe is described by Partial Differential Equations

- Maxwell's equations
- The Schrodinger Equation
- The Navier Stokes Equation
- The Einstein Field equation

359

Partial differential equations

- PDEs are more complicated than ODEs to deal with analytically: not surprisingly, they are much more difficult to handle numerically
 - many more methods
 - could be the subject of an entire course

360

Partial differential equations

- We saw with ODEs that there was an important distinction between initial value problems and two point boundary problems
- In PDEs, the very FORM of the equation determines where the bc's can be specified

361

Partial differential equations

Two key types of problem:

- *Initial value problems:*
Given the state of the system at $t=0$, compute the time evolution
- *Boundary value problems*
Find a static (time-independent) solution subject to boundary conditions on a surface enclosing a volume

362

Partial differential equations

Simplify the discussion by considering 2 \square problems

- Examples of initial value problems:
 - Solution to the 1 (spatial)D wave equation, given $\phi(x,t)$ at time $t = 0$
 $\partial^2\phi/\partial x^2 - c^{-2}\partial^2\phi/\partial t^2 = 0$
 - Solution to 1 (spatial)D diffusion equation, given $\phi(x,t)$ at time $t = 0$
 $\kappa\partial^2\phi/\partial x^2 - \partial\phi/\partial t = 0$

363

Partial differential equations

Simplify the discussion by considering 2 \square problems

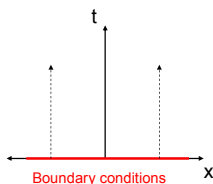
- Examples of boundary value problems:
 - Solution to Poisson's equation,
 $\partial^2\phi/\partial x^2 + \partial^2\phi/\partial y^2 - \rho = 0$

within some volume V enclosed by a boundary S on which the b.c's are specified

364

Boundary conditions

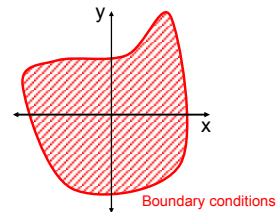
- In initial value problems, the boundary condition is typically specified at $t = 0$ and the solution is propagated forward in time, step by step



365

Boundary conditions

- In boundary value problems, the boundary condition is specified on some closed surface



366

Boundary conditions

- The methods of solution are somewhat analogous to the case of ODEs:
- Initial value problems:
 - step forward in time, using derivatives determined by the PDEs
- Boundary value problems:
 - solve within the volume using relaxation methods
 - or, create a phony time-dependence and integrate to $t \rightarrow \infty$

367

Mathematical description

- It is usually fairly clear on physical grounds whether a problem is an *initial value problem* or a *boundary value problem*

Electro/magnetostatic problem: BVP
 Time indep. Schrodinger equation: BVP
 Time dependent Schrodinger equation: IVP
 Wave equation: IVP
 Diffusion equation: IVP

368

Mathematical description

- Note, however, that the type of b.c. is also determined by the FORM of the equation
- For example, you can solve $\partial^2\phi/\partial x^2 + \partial^2\phi/\partial y^2 = 0$ (Laplace's equation) given arbitrary boundary conditions on a closed loop in the $x-y$ plane
- but you cannot solve $\partial^2\phi/\partial x^2 - c^2\partial^2\phi/\partial t^2 = 0$ (wave equation)

369

Mathematical description

- Some PDEs (“hyperbolic” and “parabolic”) have “characteristic curves” on which arbitrary boundary conditions *cannot* be imposed
 - example: the wave equation
- Some PDEs (“elliptic”) do not have such curves
 - example: Laplace's equation

370

Mathematical description

- For a 2nd-order PDE in two variables:

$$A \frac{\partial^2 \phi}{\partial x^2} + 2B \frac{\partial^2 \phi}{\partial x \partial y} + C \frac{\partial^2 \phi}{\partial y^2} + D \frac{\partial \phi}{\partial x} + E \frac{\partial \phi}{\partial y} + F \phi + G = 0$$

the characteristic curves (if they exist) are given by

$$\frac{dy}{dx} = \frac{B}{A} \pm \frac{1}{A} \sqrt{B^2 - AC}$$

371

Mathematical description

- Such PDEs are categorized according to the value of $B^2 - AC$

$B^2 - AC < 0 \rightarrow$ “elliptic eqn”
 \rightarrow No characteristic curves
 $B^2 - AC = 0 \rightarrow$ “parabolic eqn”
 \rightarrow 1 family of characteristic curves
 $B^2 - AC > 0 \rightarrow$ “hyperbolic eqn”
 \rightarrow 2 families of characteristic curves

372

Mathematical description

- Examples:
 - Poisson/Laplace's equation:
 $(A,B,C) = (1,0,1) \rightarrow B^2 - AC < 0$
 \rightarrow elliptic
 \rightarrow no characteristic curves
 \rightarrow any boundary conditions allowed

373

Mathematical description

- Examples:
 - Diffusion equation:
 $(A,B,C) = (k,0,0) \rightarrow B^2 - AC = 0$
 \rightarrow parabolic
 \rightarrow characteristic curves exist
 \rightarrow not all boundary conditions allowed

374

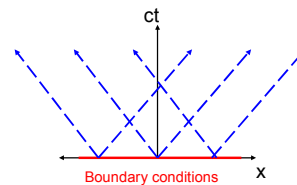
Mathematical description

- Examples:
 - Wave equation:
 $(A,B,C) = (1,0,-c^2) \rightarrow B^2 - AC = c^2 > 0$
 \rightarrow hyperbolic
 \rightarrow 2 families of characteristic curve exist
 with $dt/dx = \pm c^{-1}$
 or $dx/dt = \pm c$

375

Mathematical description

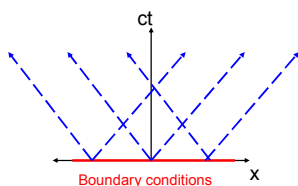
- Characteristic curves for the wave equation:
 once a boundary condition is specified at one point on a characteristic curve, it cannot be arbitrarily specified at a 2nd point



376

Mathematical description

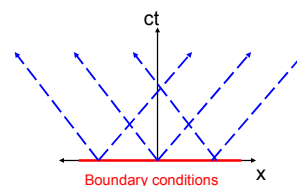
- Information propagates along the characteristic curves (at speed c in the $+x$ and $-x$ directions)



377

Mathematical description

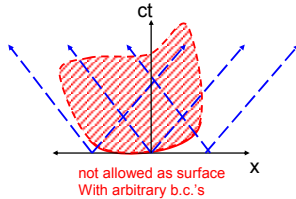
- If the b.c.'s are determined at $t = 0$, the behavior for $t > 0$ is completely specified



378

Mathematical description

- We can't arbitrarily specify ϕ on a boundary that crosses a given characteristic curve more than once



379

Lecture 17: Initial value problems

- Let's start with initial value problems, and consider numerical solution to the simplest PDE we can think of

$$\partial u / \partial t + c \partial u / \partial x = 0 \quad (\text{with } u \text{ a scalar})$$

for which the solution is a wave travelling in the +x direction, $u = f(x - ct)$

(The PDE satisfied for any function f)

380

Initial value problems

- Create a grid in space and time, where subscripted index j labels space and superscripted index n labels time

Our grid then has
 $x_j = j\Delta x$ and $t^n = n\Delta t$

Given u^n_j , (i.e. the spatial dependence of U at the n th time step), we wish to step forward in time and determine u^{n+1}_j

(NB n and $n+1$ are superscripts, not powers)

381

Finite difference equations

- As usual, we use *finite difference equations* to represent derivatives
- Simplest choice for the time derivative: forward Euler differencing

$$(\partial u / \partial t)_{j,n} = (u_j^{n+1} - u_j^n) / \Delta t$$

allows us to compute u_j^{n+1} in terms of only quantities known at timestep n .

382

Finite difference equations

- Simple choice for the spatial derivative:

$$(\partial u / \partial x)_{j,n} = (u_{j+1}^n - u_{j-1}^n) / 2\Delta x$$

- Finite difference equation becomes

$$(u_j^{n+1} - u_j^n) / \Delta t + (c/2\Delta x) (u_{j+1}^n - u_{j-1}^n) = 0$$

$$\rightarrow u_j^{n+1} = u_j^n - (c\Delta t / 2\Delta x) (u_{j+1}^n - u_{j-1}^n)$$

383

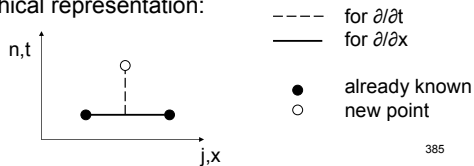
Forward Time Center Space (FTCS)

- This simplest possible scheme is called Forward Time Center Space (FTCS)
- The error is of order $(\Delta t)^2(\Delta x)^3$
 - second order accurate in space, because we take the difference between positions on either side to determine $(\partial u / \partial x)_{j,n}$
 - first order accurate in time, because we approximate the derivative $(\partial u / \partial t)_{j,n}$ by its mean value over the range n to $n+1$.

384

Forward Time Center Space (FTCS)

- This simple scheme is *explicit* (u_j^{n+1} can be written explicitly in terms of quantities already calculated)
- It is also a *single-level* scheme (in that it only involves values at time level n)
- Graphical representation:



385

Stability of PDEs

- Stability is always a key issue in PDEs (more so than for ODEs)
- Q: under what circumstances is the FTCS scheme stable?
- Address this with the von Neumann stability analysis

386

von Neumann stability analysis

- Suppose we start with a sine wave of wavelength $2\pi/k$: $u_j^n = \xi^n e^{ikj\Delta x}$
- At the next step, the FTCS scheme yields

$$u_j^{n+1} = u_j^n - (c\Delta t / 2\Delta x) (u_{j+1}^n - u_{j-1}^n)$$

$$= u_j^n (1 - (c\Delta t / 2\Delta x) [e^{ik\Delta x} - e^{-ik\Delta x}])$$

$$= \xi^n (1 - i (c\Delta t / \Delta x) \sin(k\Delta x))$$

387

von Neumann stability analysis

- So at step $n+1$, our sine wave has amplitude $\xi^{n+1} = (1 - i (c\Delta t / \Delta x) \sin(k\Delta x)) \xi^n$
- Define $R \equiv |\xi^{n+1}/\xi^n| = \sqrt{1 + [c\Delta t \sin(k\Delta x)/\Delta x]^2} > 1$

The magnitude is therefore growing exponentially for all k (i.e. for all Fourier components of u_j^n)

→ unconditional instability for any stepsize however small!

→ FTCS **never** works except over very short time periods!

388

The Lax method

Amazingly, a very minor change to this scheme (introduced by Lax) fixes this unconditional instability

Instead of

$$u_j^{n+1} = u_j^n - (c\Delta t / 2\Delta x) (u_{j+1}^n - u_{j-1}^n)$$

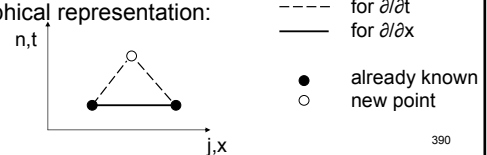
we use

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - (c\Delta t / 2\Delta x)(u_{j+1}^n - u_{j-1}^n)$$

389

The Lax method

- Like FTCS, the Lax scheme is
 - is *explicit* (u^{n+1} can be written explicitly in terms of quantities already calculated)
 - is a *single-level* scheme (in that it only involves values at time level n)
 - accurate to order $(\Delta t)^1(\Delta x)^2$
- Graphical representation:



390

Stability of the Lax method

- von Neumann stability analysis: starting again with a sine wave of wavelength $2\pi/k$:
 $u_j^n = \xi^n e^{ikj\Delta x}$
- At the next step, the Lax scheme yields

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - (c\Delta t/2\Delta x)(u_{j+1}^n - u_{j-1}^n)$$

$$= u_j^n \left(\frac{1}{2}[e^{ik\Delta x} + e^{-ik\Delta x}] - (c\Delta t/2\Delta x)[e^{ik\Delta x} - e^{-ik\Delta x}] \right)$$

$$= \xi^n (\cos(k\Delta x) - i(c\Delta t/\Delta x)\sin(k\Delta x))$$

391

von Neumann stability analysis

Now, the amplitude growth is governed by $|R|$, where $R \equiv \xi^{n+1}/\xi^n$

$$= \sqrt{\{\cos^2(k\Delta x) + [c\Delta t/\Delta x]^2 \sin^2(k\Delta x)\}}$$

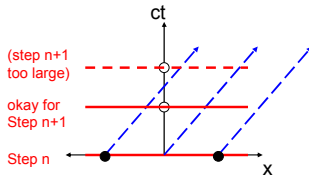
$$= 1 + ([c\Delta t/\Delta x]^2 - 1) \sin^2(k\Delta x)$$

The stability criterion is then
 $|R| < 1 \rightarrow c\Delta t < \Delta x$ (for any k)

392

Courant-(Friedrichs-Lewy) stability criterion

- This stability criterion $c\Delta t < \Delta x$ tells us that Δt must be sufficiently small that we don't require information from points beyond those sampled at step n



393

Relation of Lax to FTCS

- Note also that the Lax scheme applied to $\partial u/\partial t + c \partial u/\partial x = 0$

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - (c\Delta t/2\Delta x)(u_{j+1}^n - u_{j-1}^n)$$
 can be written

$$u_j^{n+1} = u_j^n - (c\Delta t/2\Delta x)(u_{j+1}^n - u_{j-1}^n) + \frac{1}{2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n)$$

↑
Additional term added to FTCS recipe

394

Relation of Lax to FTCS

- This additional term is the finite difference expression for $\partial^2 u/\partial x^2$
 In fact, we would use just this recipe if used FTCS to solve

$$\partial u/\partial t + c \partial u/\partial x - [(\Delta x)^2/2\Delta t] \partial^2 u/\partial x^2 = 0$$

↑
Diffusion term damps out amplitude growth

395

Upwind differencing

- For simple problems, an asymmetric expression for the spatial derivative can remove instability: we can force information to flow in the physical wave direction

$$u_j^{n+1} = u_j^n - (c\Delta t/\Delta x)(u_{j+1}^n - u_j^n) \quad c < 0$$

$$u_j^{n+1} = u_j^n - (c\Delta t/\Delta x)(u_j^n - u_{j-1}^n) \quad c > 0$$

396

Errors in the integration of PDEs

- Exponential decay is clearly preferable
- Short wavelength perturbations decay fastest because
decay rate $\propto 1 - |R| \propto \sin^2(k\Delta x)$

Not surprisingly, our integration does best for Fourier components that are well sampled (with $k\Delta x \ll 1$)

397

Errors in the integration of PDEs

- Clearly, for this simple case, we know that the true solution is $|R| = 1$
(our wave propagates at constant amplitude along the x axis)
- The FTCS method always yields $|R| > 1$
– exponential growth for all k and Δt
- The Lax method yields $|R| < 1$ if the Courant criterion is satisfied
– exponential decay

398

Errors in the integration of PDEs

- So far, we have considered only amplitude errors
- For the Lax method, we had
 $\xi^{n+1} = \xi^n (\cos(k\Delta x) - i(c\Delta t/\Delta x) \sin(k\Delta x))$
 $\rightarrow \xi^n e^{-i(ck\Delta t)}$ in the limit $k\Delta x \rightarrow 0$
But for finite $k\Delta x$, $\arg(\xi^{n+1}/\xi^n) \neq -i(ck\Delta t)$
- Phase errors can cause problems because they artificially introduce dispersion (wave propagation rate depends on k)

399

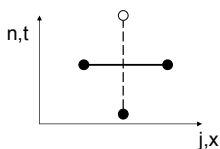
Staggered leapfrog

- We can achieve 2nd order accuracy in time if we use
 $(\partial u/\partial t)_{j,n} = (u_j^{n+1} - u_j^{n-1}) / 2\Delta t$
instead of
 $(\partial u/\partial t)_{j,n} = (u_j^{n+1} - u_j^n) / \Delta t$
- This is called staggered leapfrog

400

Staggered leapfrog

- This is no longer a *single-level* scheme (in that it now involves values at time level n and $n-1$)
...but it is still explicit
- Graphical representation:



401

Staggered leapfrog

- For $\partial u/\partial t + c \partial u/\partial x = 0$, staggered leapfrog yields
 $(u_j^{n+1} - u_j^{n-1}) = (c\Delta t/\Delta x) (u_{j+1}^n - u_{j-1}^n)$

The von Neumann stability analysis gives us

$$\xi^{n+1} - \xi^{n-1} = \xi^n (-2i (c\Delta t/\Delta x) \sin(k\Delta x))$$

$$R^2 - 1 = -2i (c\Delta t/\Delta x) \sin(k\Delta x) R$$

$$R = -i (c\Delta t/\Delta x) \sin(k\Delta x) \pm \sqrt{\{1 - (c\Delta t/\Delta x)^2 \sin^2(k\Delta x)\}}$$

402

Staggered leapfrog

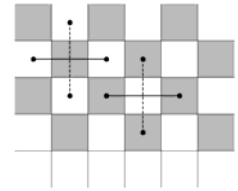
- For any $(c\Delta t/\Delta x)^2 \sin^2(k\Delta x) < 1$, $|R| = 1$

Thus, if the Courant condition applies, this 2nd order method works perfectly for our simple wave equation: the amplitude neither grows nor dissipates

403

Staggered leapfrog

- For more complex equations, staggered leapfrog can become unstable because the odd and even gridpoints are completely uncoupled; this can lead to mesh instability (see figure 19.1.16 from *Recipes*)



404

Staggered leapfrog

- Solution: couple the two meshes by introducing a weak diffusive coupling term

$$a(u_{j+1}^n - 2u_j^n + u_{j-1}^n)$$

with $a \ll 1$. This keeps them in phase.

405

Lecture 18: Diffusive initial value problems

- As a second problem, let's now consider a very simple second order PDE, the 1D diffusion equation with a constant diffusion constant $\kappa \partial^2 u / \partial x^2 - \partial u / \partial t = 0$ (This is inherently more stable than the previous case)

The obvious finite difference equation for the spatial 2nd derivative is

$$\partial^2 u / \partial x^2 = (u_{j+1}^n - 2u_j^n + u_{j-1}^n) / (\Delta x)^2$$

406

Diffusive initial value problems

- Using FTCS, this becomes $u^{n+1}_j - u^n_j = (\kappa \Delta t / (\Delta x)^2) (u^n_{j+1} - 2u^n_j + u^n_{j-1})$
- For small enough Δt , this is actually stable. The von Neumann equation is:

$$\begin{aligned} \xi^{n+1} &= \xi^n (1 + [\kappa \Delta t / (\Delta x)^2] [e^{ik\Delta x} - 2 + e^{-ik\Delta x}]) \\ &= \xi^n (1 - 2[\kappa \Delta t / (\Delta x)^2] [1 - \cos(k\Delta x)]) \\ &= \xi^n (1 - 4[\kappa \Delta t / (\Delta x)^2] \sin^2(k\Delta x / 2)) \end{aligned}$$

407

Diffusive initial value problems

- $|R| = |\xi^{n+1} / \xi^n| = |1 - 4(\kappa \Delta t / (\Delta x)^2) \sin^2(k\Delta x / 2)|$
This is < 1 (for all k) if $4(\kappa \Delta t / (\Delta x)^2) < 2$, which requires $\Delta t < (\Delta x)^2 / 2\kappa$

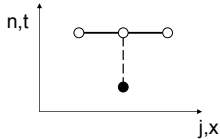
Information diffuses over a length L in time $\sim L^2 / \kappa \rightarrow$ need $\sim 2L^2 / (\Delta x)^2$ steps to solve the problem

408

Implicit techniques

- Suppose we want a solution that is accurate over large timescales/distances
- Try an implicit technique (recall ODEs)

$$u^{n+1}_j - u^n_j = (\kappa \Delta t / (\Delta x)^2) (u^{n+1}_{j+1} - 2u^{n+1}_j + u^{n+1}_{j-1})$$



409

Implicit techniques

- Can write this $\mathbf{M} \mathbf{u}^{n+1} = \mathbf{u}^n$ where \mathbf{M} is a tridiagonal matrix
- The implicit method involves the inversion of a tridiagonal matrix, which can be done quickly enough [O(N) operations]
- Stability analysis yields $R = 1 / (1 + 4(\kappa \Delta t / (\Delta x)^2) \sin^2(k \Delta x / 2))$ which is < 1 for all Δt
- But the accuracy is still only first order in Δt

410

Crank-Nicholson

- We can cleverly modify this to achieve 2nd order accuracy in time

Compute the spatial derivative using

$$\frac{1}{2}(u^{n+1}_{j+1} - 2u^{n+1}_j + u^{n+1}_{j-1}) + \frac{1}{2}(u^n_{j+1} - 2u^n_j + u^n_{j-1})$$

in place of

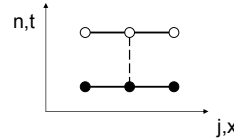
$$(u^{n+1}_{j+1} - 2u^{n+1}_j + u^{n+1}_{j-1})$$

- This estimate of $\kappa \partial^2 u / \partial x^2$ applies half way between time step n and $n+1$, the exact point at which the time derivative applies to 2nd order accuracy

411

Crank-Nicholson

- This is called Crank Nicholson, and looks like this



- Can write this $\mathbf{M}_1 \mathbf{u}^{n+1} = \mathbf{M}_2 \mathbf{u}^n$ where we now have two tridiagonal matrices \mathbf{M}_1 and \mathbf{M}_2

412

Boundary Value Problems

- Now let's consider elliptic equations, e.g. Poisson's equation in two dimensions

$$\partial^2 u / \partial x^2 + \partial^2 u / \partial y^2 = g(x, y)$$

These do not have any characteristic curves, so information doesn't flow in the problem

413

Boundary Value Problems

- $\partial^2 u / \partial x^2 + \partial^2 u / \partial y^2 = g(x, y)$ can be treated as a (very large) linear problem

Set up a N_x by N_y grid of points

The finite difference equation is

$$\frac{(u_{j-1,m} - 2u_{j,m} + u_{j+1,m})}{\Delta x^2} + \frac{(u_{j,m-1} - 2u_{j,m} + u_{j,m+1})}{\Delta y^2} = g_{j,m}$$

414

Boundary Value Problems

- Introduce a label $i = N_y(j-1) + m$ to label all points, which yields

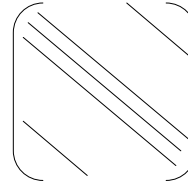
$$\frac{(u_{i-N_y} - 2u_i + u_{i+N_y})}{\Delta x^2} + \frac{(u_{i-1} - 2u_i + u_{i+1})}{\Delta y^2} = g_i$$

This is a linear problem, $\mathbf{A} \cdot \mathbf{u} = \mathbf{g}$

415

Boundary Value Problems

....where \mathbf{A} has the form



(tridiagonal with upper and lower stripe)

416

Boundary Value Problems

Unfortunately, this linear problem is often impractically large

Example: 3-D Laplace's equation with 100 x 100 x 100 grid points $\rightarrow 10^{12}$ matrix elements

Alternative and (often-used) approach: add a phony time dependence (introduces an extra dimensionality) and integrate to the limit of large time

$$\partial^2 u / \partial x^2 + \partial^2 u / \partial y^2 - g(x,y) = \partial u / \partial t$$

417

Boundary Value Problems

- This turns out to be stable provided

$$2\Delta t \left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2} \right) \leq 1$$

- We take the equality, and take $\Delta x = \Delta y = \Delta$ (square grid) to obtain

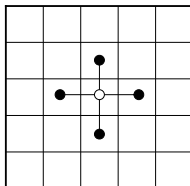
$$u_{j,m}^{n+1} = \frac{1}{4} \left\{ u_{j-1,m}^n + u_{j+1,m}^n + u_{j,m-1}^n + u_{j,m+1}^n - \Delta^2 g_{j,m} \right\}$$

418

Boundary Value Problems

- This is called the Jacobi method

Picture:



419

Boundary Value Problems

- Unfortunately, this converges extremely slowly: requires of order $N_x N_y$ time steps to achieve convergence
 - Problem: small scale structure converges rapidly, but large scale structure converges slowly
 - Solution: switch between coarser and finer grids

420

Boundary Value Problems

- Example scheme:
 - 1) start with fine grid
 - 2) march through a few time steps
 - 3) coarsen grid by averaging nearby points
 - 4) march through a few time steps
 - 5) refine grid by interpolating between adjacent grid points

....keep repeating 2 – 6

421

Lecture 19: Statistics & data analysis (Recipes, Chapters 14 and 15*)

- Two types of problem

Probability theory is a rigorous branch of mathematics
→ allows us to calculate the probability of observing t if θ is true: $P(t | \theta)$

Statistics is a much less rigorous branch of mathematics and deals to a large extent with the inverse problem: given a measured value of t , what can we say about θ ?

*see also, *Statistics: a Guide to the Use of Statistical Methods in the Physical Sciences*, by Roger J. Barlow (Wiley)

422

Statistics and data analysis

Example of a result in probability theory: if the average number of radioactive decays occurring during some time interval is μ , the probability of observing n such events is $P(n | \mu) = \mu^n e^{-\mu}/n!$

Important notes:

μ is a continuous parameter (can take any non-negative real value)

n is an observable (random variable) and is an integer

P is not symmetric under interchange of n and μ (except in the limit of large n and μ)

423

Classical versus Bayesian statistics

- In statistics, there are two approaches to answering “given a measured value of t , call it t_m , what can we say about θ ?”
- Classical statistics approach:
 - There is a fixed, unknown value of θ , and all we can do is to quote values of θ , call them θ_1 and θ_2 , for which we know the probabilities of finding $t \leq t_m$ and $t \geq t_m$ in many repeated experiments: $\theta_1 < \theta < \theta_2$ is called the confidence interval

424

Classical versus Bayesian statistics

- In statistics, there are two approaches to answering “given a measured value of t , call it t_m , what can we say about θ ?”
- Bayesian statistics approach:
 - θ is a random variable that can be described by a probability distribution. Our experimental results and some ASSUMPTIONS allow us to describe the distribution

425

Classical versus Bayesian statistics

Bayesian and classical statistics meet in 2 places

1) Systems described by a Gaussian probability distribution

$$P(t | \theta) = \exp(-[t-\theta]^2/2\sigma^2) / (2\pi)^{1/2}\sigma = P(\theta | t)$$

i.e. P is symmetric in t and θ

- Classicists integrate over t
- Bayesians integrate over θ

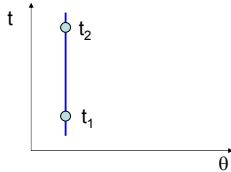
426

Classical versus Bayesian statistics

Bayesian and classical statistics meet in 2 places

2) Properly constructed 1-D problems

Measurements define confidence intervals which can be determined by the Neyman construction



Make many vertical lines and mark points t_1 and t_2 such that

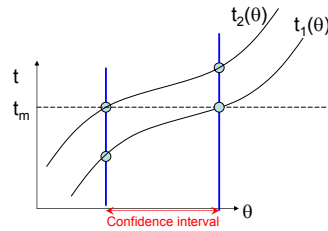
$$P(t > t_2 | \theta) = \alpha$$

$$P(t < t_1 | \theta) = \alpha$$

427

Confidential intervals for 1-D problems

- Neyman construction: $\int_{t_2}^{\infty} P(t | \theta) dt = \int_{-\infty}^{t_1} P(t | \theta) dt = \alpha$



428

Confidential intervals for 1-D problems

- This construction defines a central confidence interval of probability content $\beta = 1 - 2\alpha$
- A common convention is to choose $\beta = 0.683 (\pm 1\sigma \text{ for a Gaussian})$

429

Confidential intervals for 1-D problems

- The center of the confidence interval is not uniquely defined: common choices are
 - A symmetrized interval, $\theta = \bar{\theta} \pm \Delta\theta$, where $\bar{\theta} = (\theta_1 + \theta_2) / 2$ and $\Delta\theta = |\theta_1 - \theta_2| / 2$
 - Use the 50% value defined by $\int_{t_{50}}^{\infty} P(t | \theta) dt = \int_{-\infty}^{t_{50}} P(t | \theta) dt = 0.5$ to write $\theta = \theta_{50 - (\theta_{50} - \theta_1)}$

430

Bayesian statistics

- The Neyman construction is also something that a Bayesian can love:

Bayesian statistics starts with

$$P(A \cap B) = P(A|B) P(B) = P(B|A) P(A)$$

(Bayes' theorem)

In our language, we can write this

$$P(t|\theta) dt \cdot P(\theta) d\theta = P(\theta|t) d\theta \cdot P(t) dt$$

431

Bayesian statistics

- Our definition of t_1 and t_2 implies

$$\int_{t_1}^{t_2} P(t | \theta) dt = 1 - 2\alpha = \beta$$

- Multiply by $P(\theta)$ and integrate $d\theta$ to obtain

$$\int_{-\infty}^{\infty} d\theta P(\theta) \int_{t_1}^{t_2} dt P(t | \theta) = \beta \int_{-\infty}^{\infty} P(\theta) d\theta$$

$$\Rightarrow \int_{-\infty}^{\infty} d\theta \int_{t_1(\theta)}^{t_2(\theta)} dt P(t | \theta) P(\theta) = \beta$$

432

Bayesian statistics

- Apply Bayes' Theorem, and interchange order of integration:

$$\int_{-\infty}^{\infty} dt P(t) \int_{\theta_1(t)}^{\theta_2(t)} d\theta P(\theta | t) = \beta$$

- Suppose we measure the value $t = t_m$. Then $P(t) = \delta(t - t_m)$, and we can write

$$\int_{\theta_1(t_m)}^{\theta_2(t_m)} P(\theta | t_m) d\theta = \beta$$

433

Bayesian statistics

- This equation $\int_{\theta_1(t_m)}^{\theta_2(t_m)} P(\theta | t_m) d\theta = \beta$

says that given the measurement, $t = t_m$, the random variable θ has a probability β of lying between $\theta_1(t_m)$ and $\theta_2(t_m)$

Holds for integrated confidence intervals in 4 D

434

Summary

- By understanding our experiment, we know $P(t|\theta) dt$
- We would like to know $P(\theta|t_m) d\theta$ for a given measurement $t=t_m$
- We can compute

$$\int_{\theta_1(t_m)}^{\theta_2(t_m)} P(\theta | t_m) d\theta = \beta$$

435

“Bad” Bayesian statistics

- Classic fallacy is to assume that $P(\theta|t) = P(t|\theta) P(\theta) / P(t) = P(t|\theta)$

Example: we observe n radioactive decays within a given time period Δt and assume

$$P(\mu|n) = P(n|\mu) = n! e^{-n\mu} / \mu^n$$

to obtain confidence limits on the decay rate $\mu/\Delta t$

436

Basic definitions

- Our textbook devotes several sections to describing the comparison of measured distribution functions of random variables. The basic definitions are useful:

- Mean $\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$

- Variance, $Var(x) = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2 \approx \langle x^2 \rangle - \langle x \rangle^2$

- Standard deviation, $\sigma = Var(x)^{1/2}$

437

Basic definitions

- Skew $Skew(x) = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^3 / \sigma^3$

- Kurtosis $Kurt(x) = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^4 / \sigma^4 - 3$

- These 3rd and 4th moments are conventionally defined to be dimensionless and equal to zero for a Gaussian distribution

The book discusses the comparison of the means and variances of different distributions (Student's t-test and F test)

438

Basic definitions

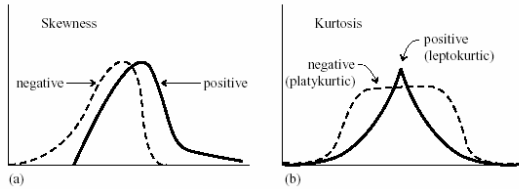


Figure 14.1.1. Distributions whose third and fourth moments are significantly different from a normal (Gaussian) distribution. (a) Skewness or third moment. (b) Kurtosis or fourth moment.

from Recipes

439

Central limit theorem

- An important property of the variance is that it is additive (like the mean)
- If $z = x + y$, where x and y are random variables drawn independently from different distributions, then

$$\bar{z} = \bar{x} + \bar{y}$$

$$\text{Var}(z) = \text{Var}(x) + \text{Var}(y)$$

- Hence, if X is the sum of N random variables x_i drawn from a set of arbitrary distributions, $R_i(x_i)$, then

$$\bar{X} = \sum_{i=1}^N \bar{x}_i \quad \text{Var}(X) = \sum_{i=1}^N \text{Var}(x_i)$$

440

Central limit theorem

- The CLT says that in the limit of large N , the distribution of X is Gaussian:

$$P(X) = \frac{e^{-(X-\bar{X})^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}$$

- This justifies many of the Gaussian approximations commonly made in statistics

441

Central limit theorem

- Example: the mean of a sample $\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$

itself tends to be normally distributed with a “standard error”

$$\delta\bar{x} = \sqrt{\frac{\sigma_x^2}{N}} = \sqrt{\frac{\text{Var}(x)}{N}}$$

442

Central limit theorem: “proof”

- Consider a variable x_1 that has a probability distribution $P_1(x_1)$, and a variable x_2 that has a probability distribution $P_2(x_2)$

Let $X = x_1 + x_2$

Then $P(X) dX = \int P_1(x_1) P_2(X - x_1) dx_1 dX$

$$\rightarrow P = P_1 * P_2 \rightarrow \tilde{P} = \tilde{P}_1 \tilde{P}_2$$

(where tilde denotes a Fourier transform)

443

Central limit theorem: “proof”

- We know that

$$\begin{aligned} \tilde{P}_1(k) &= \int e^{ikx_1} P_1(x_1) dx_1 \\ &= \int P_1(x_1) dx_1 + ik \int x_1 P_1(x_1) dx_1 - \frac{1}{2} k^2 \int x_1^2 P_1(x_1) dx_1 + \dots \\ &= 1 + ik\bar{x}_1 - \frac{1}{2} k^2 \langle x_1^2 \rangle + O(k^3) \end{aligned}$$

444

Central limit theorem: “proof”

- Taking the natural log, and using $\ln(1+y) = y - \frac{1}{2}y^2 + O(y^3)$, we find that

$$\ln \tilde{P}_1(k) = ik \langle x_1 \rangle - \frac{1}{2}k^2 \langle x_1^2 \rangle + \frac{1}{2}k^2 \langle x_1 \rangle^2 + O(k^3)$$

$$= ik\bar{x}_1 - \frac{1}{2}k^2 \text{Var}(x_1) + O(k^3)$$
- Thus the probability distribution for $X=x_1 + x_2$ has

$$\begin{aligned} \ln \tilde{P}(k) &= \ln \tilde{P}_1(k) \tilde{P}_2(k) = \ln \tilde{P}_1(k) + \ln \tilde{P}_2(k) \\ &= ik(\bar{x}_1 + \bar{x}_2) - \frac{1}{2}k^2 \{\text{Var}(x_1) + \text{Var}(x_2)\} + O(k^3) \\ &= ik\bar{X} - \frac{1}{2}k^2 \text{Var}(X) + O(k^3) \end{aligned}$$

445

Central limit theorem: “proof”

- This expression is clearly true when X is the sum of any number of random variables

$$\ln \tilde{P}(k) = ik\bar{X} - \frac{1}{2}k^2 \text{Var}(X) + O(k^3)$$

Key point: as we add more and more random variables, the probability distribution gets broader and broader \rightarrow its Fourier transform is described better and better by the terms of lowest order in k

446

Central limit theorem: “proof”

- So in the limit of large N , we take

$$\begin{aligned} \ln \tilde{P}(k) &= ik\bar{X} - \frac{1}{2}k^2 \text{Var}(X) \\ \Rightarrow \tilde{P}(k) &= e^{ik\bar{X}} e^{-\frac{1}{2}k^2 \text{Var}(X)} \end{aligned}$$

447

Central limit theorem: “proof”

- The inverse Fourier transform yields

$$\begin{aligned} \tilde{P} &= \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-ikx} e^{ik\bar{X}} e^{-\frac{1}{2}k^2 \sigma^2} dk \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \exp\left[-\frac{1}{2}[k\sigma - i(\bar{X}-x)/\sigma]^2\right] e^{-\frac{1}{2}i(\bar{X}-x)/\sigma^2} dk \\ &= \frac{1}{2\pi} \sqrt{2\pi/\sigma^2} e^{-\frac{1}{2}i(\bar{X}-x)/\sigma^2} \\ &= \frac{e^{-\frac{1}{2}i(\bar{X}-x)/\sigma^2}}{\sqrt{2\pi\sigma^2}} \end{aligned}$$

448

Lecture 20: Hypothesis testing

- Much of statistics involves hypothesis testing
 - compare a new interesting hypothesis, H_1 (the “Alternative hypothesis”) to the boring, old, well known case, H_0 (the “Null Hypothesis”)
 - or, decide whether to reject a single hypothesis, H_0 , even without a competing hypothesis

449

Hypothesis testing

- A hypothesis may be either
 - Simple: completely defined without any free parameters
 - Composite: having free parameters to be determined from the data

450

Hypothesis testing

- Let W be the space of all possible physical observations (experimental outcomes)
- We want to divide this into
 - The “critical region”: w , the set of all observations for which we reject H_0
 - The “acceptance region”: $W - w$, the set of observations for which we accept H_0
- To do this, we define a function $x(w)$ called the “test statistic”. The art of hypothesis testing amounts to making a good choice of $x(w)$

451

Hypothesis testing

- To define a good test statistic we need to understand two quantities:
 - The **significance** α of the test is the probability that a true null hypothesis will (erroneously) fail the test (a Type I error): $\alpha = P(x \in w | H_0)$
 - The **power** of the test, $1 - \beta$, is the probability that the null hypothesis will fail the test if the alternative hypothesis H_1 is true $1 - \beta = P(x \in w | H_1)$

452

Hypothesis testing

- Alternatively, $\beta = 1 - \text{power} = 1 - P(x \in w | H_1) = P(x \in W - w | H_1)$

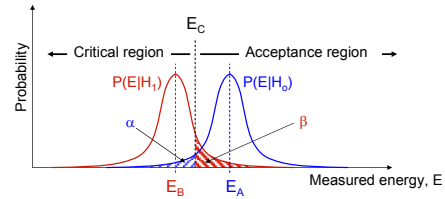
This is the probability that the Null Hypothesis will be (erroneously) accepted even if the Alternative Hypothesis is correct (Type II error)

- We want to choose a test statistic (and define a critical region) which minimizes α and β

453

Example

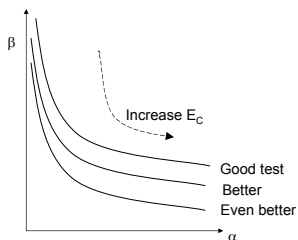
- We detect a photon and measure its energy E : was it emitted by
 - atom A (which emits at energy E_A) – call this the Null Hypothesis, H_0
 - or atom B (which emits at energy E_B) – alternative hypothesis H_1



454

Example

- An ideal test mimimizes BOTH α and β
- Typically, we can improve one at the cost of worsening the other by varying a parameter (E_C)



455

Type I and Type II errors

- The tradeoff we choose between Type I and Type II errors depends upon which we consider more serious, e.g. in a criminal trial, we have:

Null Hypothesis, H_0 : the defendant is innocent

Alternative hypothesis, H_1 : the defendant is guilty

Type I error: innocent defendant wrongly convicted \rightarrow probability = α

Type II error: guilty defendant wrongly acquitted \rightarrow probability = β

Type I error is considered worse: \rightarrow set a very small α (guilt must be proved “beyond a reasonable doubt”) even though $1 - \beta$ is thereby reduced (the guilty are more likely to be acquitted)

456

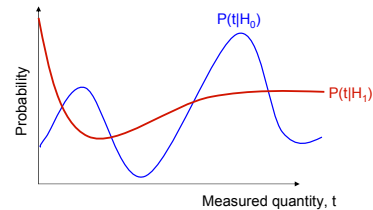
Neyman Pearson test

- Regardless of how we weigh the seriousness of Type 1 and Type II errors, we always want to minimize β for given α , (or minimize α for given β).
- In the example considered previously, there was only one reasonable way of defining the critical region (by means of a cut at E_C)

457

Neyman Pearson test

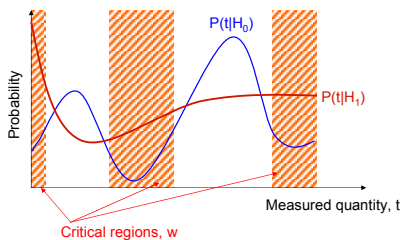
- But suppose we have a more complicated situation



458

Neyman Pearson test

- We should reject the hypothesis when $P(t|H_1)$ is large and $P(t|H_0)$ is small



459

Neyman Pearson test

- The optimal test always rejects the hypothesis when the *likelihood ratio* $P(t|H_1)/P(t|H_0)$ exceeds some constant
- Where we *set* the threshold likelihood ratio depends on how we set the tradeoff between minimizing α and β (i.e. whether we are more averse to making Type 1 or Type II errors)

460

Goodness of fit

- Another type of test simply considers whether a single Null Hypothesis H_0 should be rejected
 - We can still find the probability of rejecting H_0 when it is true
 - But we cannot evaluate a probability of accepting H_0 when it is false
- Most famous test of this kind is called the Chi-Squared test

461

Chi-squared

- Consider a set of N independent measurements y_i with $i = 1, N$ at N values of a dependent variable x_i
 - Suppose the measurements are normally distributed with standard deviations σ_i .
 - Joint N -dimensional probability distribution for obtaining results y_1, y_2, \dots, y_N is

$$P(y_1, y_2, \dots, y_N) dy_1 dy_2 \dots dy_N = \prod_{i=1}^N \frac{\exp(-[y_i - \bar{y}_i(x_i)]^2 / 2\sigma_i^2)}{\sqrt{2\pi\sigma_i^2}} dy_i$$

462

Chi-squared

- Define $\chi^2 \equiv \sum_{i=1}^N \frac{(y_i - \bar{y}_i(x_i))^2}{\sigma_i^2}$

- Then

$$P(y_1, y_2, \dots, y_N) dy_1 dy_2 \dots dy_N = \prod_{i=1}^N \frac{\exp(-[y_i - \bar{y}_i(x_i)]^2 / 2\sigma_i^2)}{\sqrt{2\pi\sigma_i^2}} dy_i$$

$$\Rightarrow P(y_1, y_2, \dots, y_N) = \frac{\exp(-1/2 \chi^2)}{\prod_{i=1}^N \sqrt{2\pi\sigma_i^2}}$$

463

Chi-squared

- Our Chi-squared test will reject H_0 if χ^2 exceeds some threshold value χ_0^2
- If H_0 is true, the probability that χ^2 exceeds χ_0^2 is

$$\alpha = P(\chi^2 > \chi_0^2 | H_0) \propto \int_{\chi_0^2}^{\infty} (d^N y / d\chi^2) \exp(-1/2 \chi^2) d\chi^2$$

- The element of hypervolume $d^N y$ is proportional to $\chi^{N-1} d\chi = 1/2 \chi^{N-2} d\chi^2$

464

Chi-squared

- Hence the significance of the test becomes

$$\alpha = P(\chi^2 > \chi_0^2 | H_0) = k \int_{\chi_0^2}^{\infty} (\chi^2)^{1/2 N - 1} \exp(-1/2 \chi^2) d\chi^2$$

- The normalization constant k is obtained from

$$1 = P(\chi^2 > 0 | H_0) = k \int_0^{\infty} t^{1/2 N - 1} \exp(-1/2 t) dt = k \cdot 2^{1/2 N} \Gamma(1/2 N)$$

465

Chi-squared

- The gamma function is defined (*Recipes*, Ch 6.1) by

$$\Gamma(a) = \int_0^{\infty} t^{a-1} e^{-t} dt$$

For integral a, $\Gamma(a) = (a-1)!$, and $\Gamma(1/2) = \pi^{1/2}$

The incomplete Gamma function is given by

$$\gamma(a, x) = \int_0^x t^{a-1} e^{-t} dt$$

466

Chi-squared

- Hence, the probability of χ^2 exceeding χ_0^2 if H_0 is true can be written

$$\alpha = P(\chi^2 > \chi_0^2 | H_0)$$

$$\equiv Q(1/2 N, 1/2 \chi_0^2) = 1 - \frac{2^{-1/2 N}}{\Gamma(1/2 N)} \gamma(1/2 N, 1/2 \chi_0^2)$$

467

Chi-squared

- Check for $N = 1$, for which $\Gamma(1/2 N) = \Gamma(1/2) = \pi^{1/2}$

$$\alpha = P(\chi^2 > \chi_0^2 | H_0)$$

$$\equiv Q(1/2, 1/2 \chi_0^2) = 1 - \frac{2^{-1/2}}{\pi^{1/2}} \gamma(1/2, 1/2 \chi_0^2)$$

$$= 1 - \frac{1}{(2\pi)^{1/2}} \int_0^{1/2 \chi_0^2} t^{-1/2} \exp(-t) dt$$

$$= 1 - \frac{1}{(2\pi)^{1/2}} \int_0^{1/2 \chi_0^2} (1/2 \chi^2)^{-1/2} \exp(-1/2 \chi^2) d(1/2 \chi^2)$$

$$= 1 - \frac{1}{\pi^{1/2}} \int_0^{\chi_0} \exp(-1/2 \chi^2) d\chi = \frac{1}{\pi^{1/2}} \int_{\chi_0}^{\infty} \exp(-1/2 \chi^2) d\chi$$

468

χ^2 for a composite hypothesis

- Suppose now that H_0 is composite, containing m free parameters, a_j with $j = 1, m$
- Recipe: determine set of a_j that minimizes χ^2 and call the resultant minimum χ_{\min}^2

The probability that this minimum exceeds χ_0 is $Q(1/2\nu, 1/2\chi_0)$, where $\nu = N - m$ is the “number of degrees of freedom”

469

Event Counting

- Many experiments (especially in particle physics) count events:

Examples:

- number of events in an accelerator satisfying a particular criterion
- number of X-ray photons detected in a particular photon energy bin

470

Event Counting

- The observed number of photons, n_i , in energy bin i centered at energy x_i , has a Poisson distribution
- In the limit of large n_i (10 – 20 will usually suffice), this can be approximated adequately by a Gaussian distribution with variance n_i
- Suppose the Null Hypothesis predicts an expectation value $g_i(x_i)$ for bin i

$$g_i(x_i) = \int_{\text{bin } i} f(x_i|H_0) dx_i$$

471

Event Counting

- Our test statistic is then $\chi^2 \equiv \sum_{i=1}^N \frac{(n_i - g_i(x_i))^2}{n_i}$

Even without a prediction for $g_i(x_i)$, we can test a null hypothesis that two samples, n_i and m_i , are drawn from the same distribution (e.g. test the hypothesis that the X-ray spectrum is the same now as it was 1 year ago)

If the total number of events are the same,

$$\chi^2 \equiv \sum_{i=1}^N \frac{(n_i - m_i)^2}{n_i + m_i}$$

472

Event Counting

- Summary:
 - Chi-squared works well provided the number of events is sufficient to justify the approximation to a Gaussian
 - It consists of two parts: (1) a test statistic and (2) a probability function
 - It is distribution-free (doesn't depend on $f(x_i)$)
 - It needs lots of data (want many bins, each with 10 – 20 events)
 - It depends on binning
- Question:
 - Are there bin-INDEPENDENT tests that work on smaller samples? (Answer: In 1-D, yes)

473

Lecture 21: Order statistics

- Suppose we have N measurements of a scalar, $x_i = 1, N$
- Take all measurements and sort them into ascending order $x_1 \leq x_2 \leq x_3 \dots \leq x_N$
- Define the measured running integral

$$S_N(x) = 0 \text{ for } x < x_1$$

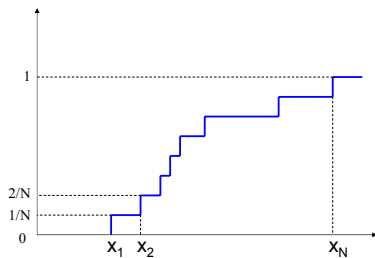
$$= i/N \text{ for } x_i \leq x < x_{i+1}$$

$$= 1 \text{ for } x \geq x_N$$

474

Order statistics

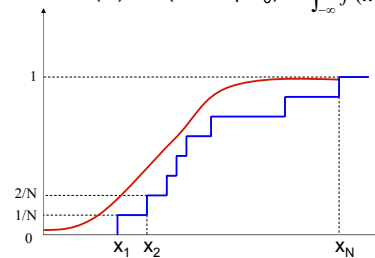
- So $S_N(x)$ looks like this:



475

Order statistics

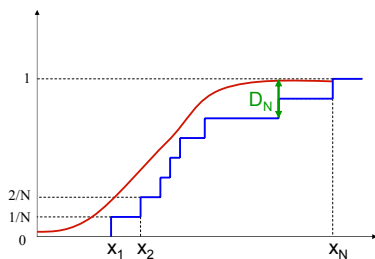
- We would like to compare this with the predicted function $F(x') = P(x < x' | H_0) = \int_{-\infty}^{x'} f(x | H_0) dx$



476

Order statistics

- The most famous statistic is $D_N \equiv \max |S_N(x) - F(x)|$



477

Kolmogorov-Smirnov test

- Turns out the probability function for $N^{1/2}D_N$ is distribution-free and calculable for N large (≥ 20 or so)

$$\alpha = P(N^{1/2}D_N > z) = 2 \sum_{j=1}^{\infty} (-1)^{j-1} \exp(-2j^2 z^2)$$

$\xrightarrow{z > a \text{ few}} 2 \exp(-2z^2)$

478

Kolmogorov-Smirnov test

- D_N and $P(N^{1/2}D_N > z)$ form the K- test which is
 - Independent of binning
 - Valid for small/medium sample sizes (also large, of course)
 - Loses sensitivity near endpoints (x_1 or x_N), since both integrals tend to same value (0 or 1) (this shortcoming is addressed in some of the variants discussed in the book)
 - Like χ^2 , can be generalized to the case of comparing two measured distributions to see if they are drawn from different samples

479

Kolmogorov-Smirnov test

- Comparing two different samples, our measured running integrals are S_M and S_N
- Defining $D_{NM} = \max |S_M(x) - S_N(x)|$, the probability of $(MN/[M+N])^{1/2}D_{NM}$ exceeds a given value z is again

$$\alpha = P([NM/(N+M)]^{1/2}D_{NM} > z)$$

$$= 2 \sum_{j=1}^{\infty} (-1)^{j-1} \exp(-2j^2 z^2)$$

(i.e. $(MN/[M+N])^{1/2}D_{NM}$ replaces $N^{1/2}D_N$)

480

Cramer-Von Mises-Smirnov test

- Less well known is the statistic

$$W^2 = \int_{-\infty}^{\infty} (S_N(x) - F(x))^2 f(x) dx = \int_0^1 (S_N(x) - F(x))^2 dF$$

The quantity NW^2 is also distribution free and has a calculable probability function

$$\alpha = P(NW^2 > z)$$

$$= 1 - \sum_{j=1}^{\infty} (-1)^j \binom{-1/2}{j} (4j+1)^{1/2} \exp(-[4j+1]^2/16z) K_{1/4}([4j+1]^2/16z)$$

← Binomial coeff.
← Modified Bessel fn
481

Cramer-Von Mises-Smirnov test

- This test
 - works for very small samples ($N \geq 3$ or so)
 - is more sensitive than K-S test
 - is the most powerful 4-parameter test
 - cannot be easily generalized to compare 2 measured distributions
 - can be written in a more easily evaluated form:

$$W^2 = \frac{1}{N} \left(\frac{1}{12N} + \sum_{i=1}^N \left[F(x_i) - \frac{(2i-1)}{2N} \right]^2 \right)$$

482

Modeling data

- So far, we have discussed only hypothesis testing.
- Often, we *assume* some composite hypothesis and wish to constrain its adjustable parameters by experiment
- Suppose there are m unknown parameters, a_j (with $j = 1, m$) and N data points

483

Modeling data

- The data are typically of two kinds:
 - 1) We vary some external variable x , and make N measurements of some quantity y to obtain the results

$$y_i(x_i) \pm \sigma_i \text{ (with } i = 1, N)$$
 where σ_i is the Gaussian uncertainty on y_i

Example: we adjust the voltage $x_i = V_i$ across a diode to N values and measure the current $y_i = I_i \pm \sigma_i$ for each value

484

Modeling data

- The data are typically of two kinds:
 - 2) We measure N values of the random variable x_i in an experiment

Example: measurement of individual photon energies in X-ray astronomy

485

Modeling data

- In both cases, we would like to determine the parameter set that best approximates the data
- Need an "estimator" which measures how well the model matches the data
- For experiments of the first type, we commonly use the joint Gaussian probability function

$$P(\underline{y} | \underline{a}) = \frac{\exp(-1/2 \chi^2(\underline{a}))}{\prod_{i=1}^N \sqrt{2\pi\sigma_i^2}}$$

- We want to maximize $P(\underline{y} | \underline{a})$ by minimizing $\chi^2(\underline{a})$ (and can also see if the model fits by calculating the significance)

486

Confidence intervals

- Suppose we have carried out the minimization of $\chi^2(\mathbf{a})$
- The value of \mathbf{a} at the minimum yields our BEST ESTIMATE of the free parameters, $a_i \rightarrow$ call this \mathbf{a}^*
- Question: what are our confidence limits on the a_i ?

487

Confidence intervals

- Because our probability distribution is symmetric under interchange of y_i and $\bar{y}_i(x_i, \underline{a})$, we can use

$$P(\underline{a} | \underline{y}) = P(\underline{y} | \underline{a}) = \frac{\exp(-1/2 \chi^2(\underline{a}))}{\prod_{i=1}^N \sqrt{2\pi\sigma_i^2}}$$

488

Confidence intervals

- Let's use a Taylor expansion for $\chi^2(\mathbf{a})$:

$$\chi^2(\underline{a}) = \chi^2(\underline{a}^*) + \sum_j (a_j - a_j^*) \left. \frac{\partial \chi^2}{\partial a_j} \right|_{\underline{a}^*} + \frac{1}{2} \sum_{j,k} (a_j - a_j^*)(a_k - a_k^*) \left. \frac{\partial^2 \chi^2}{\partial a_j \partial a_k} \right|_{\underline{a}^*}$$

\rightarrow Zero

$$= \chi^2(\underline{a}^*) + (\underline{a} - \underline{a}^*)^T \underline{C} (\underline{a} - \underline{a}^*)$$

where $\underline{C}_{jk} = \frac{1}{2} \left. \frac{\partial^2 \chi^2}{\partial a_j \partial a_k} \right|_{\underline{a}^*}$

and the probability density is

$$P(\underline{a}) \propto \exp(-1/2 \chi^2) = \exp[-1/2 (\underline{a} - \underline{a}^*)^T \underline{C} (\underline{a} - \underline{a}^*)] \quad 489$$

Confidence intervals

- Define $\Delta \mathbf{a} = \mathbf{a} - \mathbf{a}^*$
- Then $\Delta \chi^2 = \chi^2 - \chi_{\min}^2$
with $\Delta \chi^2 = \Delta \mathbf{a}^T \mathbf{C} \Delta \mathbf{a} \rightarrow \mathbf{C}^{-1} = \Delta \mathbf{a} \Delta \mathbf{a}^T / \Delta \chi^2$
which is called the error matrix

490

Confidence intervals

The diagonal elements of the error matrix $(\mathbf{C}^{-1})_{ii}$ tell us $(\Delta a_i)^2 / \Delta \chi^2$ for each parameter provided that the 2nd order Taylor expansion is adequate

Thus, the Bayesian probability for a given value of Δa_i is $P(\Delta a_i | y) \propto \exp(-1/2 \Delta \chi^2) \propto \exp(-1/2 (\Delta a_i)^2 / (\mathbf{C}^{-1})_{ii})$

....a Gaussian with mean 0 and standard deviation $\sigma_i = \sqrt{(\mathbf{C}^{-1})_{ii}}$

491

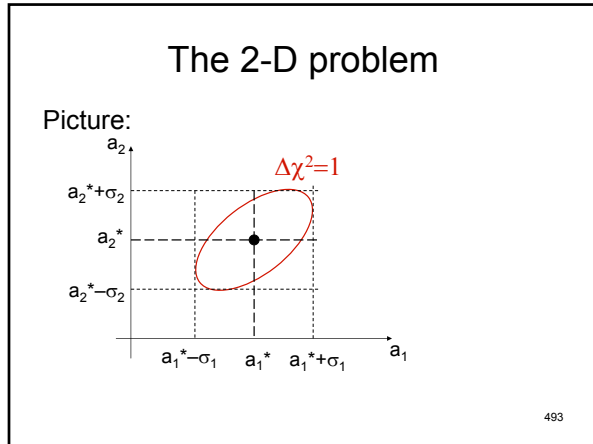
The 2-D problem

- Let's consider the 2-D problem in which we have just two free parameters a_1 and a_2
- The data yield

$$\Delta \chi^2(a_1, a_2) = \chi^2(a_1, a_2) - \chi_0^2 = [c_{11} (\Delta a_1)^2 + c_{22} (\Delta a_2)^2 + 2c_{12} (\Delta a_1 \Delta a_2)] + \dots$$

Note that contours of fixed $\Delta \chi^2$ are ellipses in the (a_1, a_2) plane

492



The 2-D problem

- We find that

$$\sigma_1^2 = (\mathbf{C}^{-1})_{11} = c_{22}/(c_{11}c_{22} - c_{12}^2)$$

$$\sigma_2^2 = (\mathbf{C}^{-1})_{22} = c_{11}/(c_{11}c_{22} - c_{12}^2)$$

The 68% confidence limit on a_1 is $a_1^* \pm \sigma_1$
 The 68% confidence limit on a_2 is $a_2^* \pm \sigma_2$

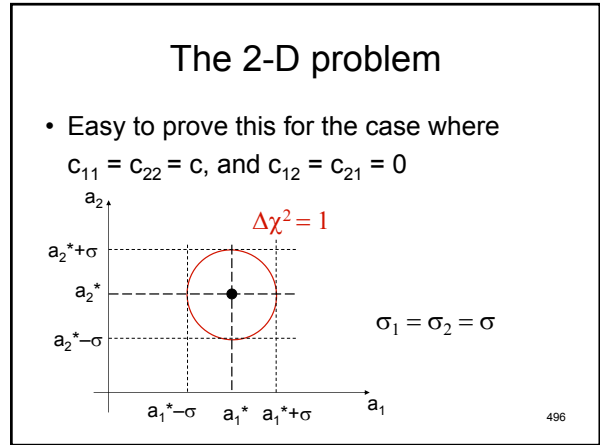
494

The 2-D problem

- The joint probability that a_1 and a_2 lie within the $\Delta\chi^2 = 1$ ellipse is NOT 68%, however.
- The *joint probability content* within the ellipse comes out to be $1 - \exp(-\frac{1}{2}\Delta\chi^2) = 0.39$

and the ellipse with 68% probability content has $\Delta\chi^2 = 2.90$

495



The 2-D problem

- In this case, the joint probability is

$$P \propto \exp(-\frac{1}{2}\Delta\chi^2) da_1 da_2$$

$$= \exp(-\frac{1}{2}c(\Delta a_1^2 + \Delta a_2^2)) da_1 da_2$$

$$= \exp(-\frac{1}{2}c(r^2)) 2\pi r dr = \pi \exp(-\frac{1}{2}c(r^2)) dr^2$$

$$\propto \exp(-\frac{1}{2}\Delta\chi^2) d(\Delta\chi^2)$$

and thus the probability that $\Delta\chi^2$ exceeds some value, z , is simply $e^{-\frac{1}{2}z}$

497

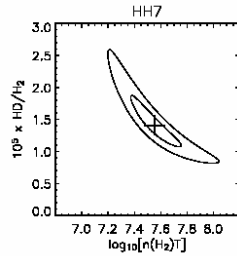
Non Gaussianity

- So far we have assumed that the 2nd order Taylor expansion is adequate out to some value of $\Delta\chi^2$
- This isn't necessarily the true for the desired value of $\Delta\chi^2$
- In that case, we have to plot the actual contours and get our confidence limits from them
- Leads to non symmetric confidence intervals

498

Non Gaussianity

- Recent example: limits on the pressure and HD/H₂ abundance ratio in interstellar gas clouds observed with the Spitzer Space Telescope



$$\Delta\chi^2 = 1 \text{ and } 2$$

499

Summary: statistical tasks and tools

Tasks	Statistic	Test/method (rejection criterion)
1 Hypothesis testing		
a) Simple hypothesis H ₁ versus simple hypothesis H ₀	Likelihood ratio, P(t H ₁)/P(t H ₀)	Neyman-Pearson P(t H ₁)/P(t H ₀) > k
b) Simple hypothesis H ₀ alone	Chi-squared	P(χ ² > χ ₀ ² H ₀) < α
c) Composite hypothesis H ₀ alone	Minimum χ ²	P(χ ² > χ _{min} ² H ₀) < α
d) Simple hypothesis H ₀ alone, where the data are the counts of a small number of events	Order statistic, D _N Order statistic, W ²	Kolmogorov-Smirnov Cramer-Von Mises-Smirnov
e) Hypothesis that two data sets are drawn for the same distribution	Chi-squared Order statistic, D _{NM}	P(χ ² > χ ₀ ² H ₀) < α Kolmogorov-Smirnov
2 Modeling of data (parameter fitting)		
	Chi-squared	χ ² minimization with c.l. from Δχ ² contours or error matrix

500

Lecture 22: Multivariate analysis and principal component analysis

- So far, we have been considering hypothesis testing and data modeling when a given quantity (e.g. X-ray photon counts) is observed as a function of an independent variable (e.g. photon energy)
- Another important task involves the analysis of a sample of objects for which a set of quantities has been measured. None is singled out as the independent variable.

Example: SAT-R, SAT-V, GPA

501

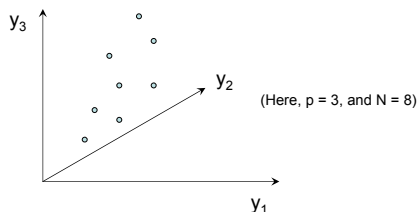
Multivariate data

- Suppose we have p quantities that have been measured for each of N objects in a sample:
- For each object, we can plot a vector, **y**, in p dimensional space to represent the values of the p measured quantities: the kth component of **y** represents the value of the kth measured quantity.
- The entire sample is represented by a set of N such vectors, **y**⁽ⁱ⁾ for i = 1, N

502

Multivariate data

- We can plot the data on a p-dimensional scatter plot



503

Multivariate data

- The sample has a mean, $\bar{\mathbf{y}} = \frac{1}{N} \sum_{i=1}^N \mathbf{y}^{(i)}$

such that \bar{y}_k is the mean of the kth measurable quantity

For each measured quantity, we can define the variance,

$$\text{Var}(y_k) = \frac{1}{N-1} \sum_{i=1}^N (y_k^{(i)} - \bar{y}_k)^2$$

504

Covariance matrix

- We can also examine correlations between the quantities by defining the covariance matrix

$$S_{kl} = \frac{1}{N-1} \sum_{i=1}^N (y_k^{(i)} - \bar{y}_k)(y_l^{(i)} - \bar{y}_l)$$

- The covariance matrix is clearly a $p \times p$ symmetric matrix
- The diagonal elements are just the variances: $S_{kk} = \text{Var}(y_k)$
- The off-diagonal elements represent correlations (because S_{km} is zero if y_k and y_m are uncorrelated)

505

Correlation matrix

- Elements of the covariance matrix are clearly not scale invariant or – in general – dimensionless

- It is often convenient to renormalize to obtain the correlation matrix $R_{kl} = \frac{S_{kl}}{\sqrt{S_{kk}S_{ll}}}$

which is dimensionless, has diagonal elements equal to unity and off-diagonal elements in the range $-1 \leq R_{kl} \leq +1$. These are called the linear correlation coefficients

506

Principal component analysis

- Principal component analysis is a powerful tool for analysing multivariate data.
- In principal component analysis, we look for linear combinations of measured variables that are *uncorrelated* with each other

$$z_1 = a_{11}y_1 + a_{12}y_2 + \dots + a_{1p}y_p$$

$$z_2 = a_{21}y_1 + a_{22}y_2 + \dots + a_{2p}y_p \text{ etc...}$$

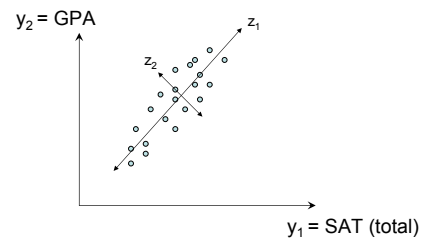
which we write $\mathbf{z} = \mathbf{A} \mathbf{y}$

where \mathbf{A} is an orthonormal matrix representing a rotation

507

Principal component analysis: graphical interpretation

The matrix \mathbf{A} rotates the axes to align them with the major and minor axes of the ellipsoid



508

Principal component analysis

- The covariance matrix for $\mathbf{z} = \mathbf{A} \mathbf{y}$ is now diagonal:

$$S'_{mn}(\mathbf{z}) = \text{Var}(z_m) \text{ for } m = n$$

$$= 0 \text{ otherwise}$$

How does this relate to the covariance matrix, \mathbf{S} , for \mathbf{y} ?

509

Principal component analysis

$$S'_{mn} = \frac{1}{N-1} \sum_{i=1}^N (A_{mk} y_k^{(i)} - A_{mk} \bar{y}_k)(A_{nl} y_l^{(i)} - A_{nl} \bar{y}_l)$$

$$= A_{nl} A_{mk} S_{kl}$$

Using summation convention (sum over repeated indices l, m, k, n)

In other words, $\mathbf{S}' = \mathbf{A} \mathbf{S} \mathbf{A}^T = \mathbf{A} \mathbf{S} \mathbf{A}^{-1}$

$$\rightarrow \mathbf{S} = \mathbf{A}^{-1} \mathbf{S}' \mathbf{A}$$

This is just the eigenvalue decomposition of \mathbf{S} :

$$\mathbf{S} = \mathbf{U} \mathbf{D} \mathbf{U}^{-1}$$

where $\mathbf{D} = \mathbf{S}' = \text{diag}(\lambda_k)$

and the columns of $\mathbf{U} = \mathbf{A}^T$ are the eigenvectors

510

Principal component analysis

So obtaining the principal components is entirely equivalent to finding the eigenvectors of the covariance matrix: each eigenvector gives us one principal component

The corresponding eigenvalue tells us the variance of that principal component (PC), and the PCs with the largest variances are the most important.

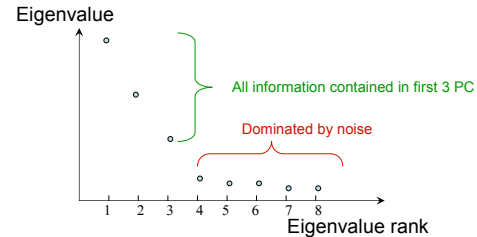
A key application of PCA is to reduce the dimensionality of the problem in the case where the eigenvalues cover a wide dynamic range

→ We need only consider the first few PC instead of all p PC, greatly simplifying the description of the correlations

511

Scree graphs

We number the principal components in order of decreasing eigenvalue, and plot the eigenvalue as a function of its rank



512

Preconditioning for scale invariance

We noted previously that the covariance matrix is not scale invariant: result of PCA can depend on the units used

Often makes sense to precondition the variables by dividing each variable y by the square root of its variance: effectively amounts to working with the correlation matrix in place of the covariance matrix

513

Example 1: properties of winged aphids (Jeffer's 1967)

Jeffer's analysed a sample of 40 winged aphids for which 19 variables had been measured

LENGTH body length
 WIDTH body width
 FORWING forewing length
 HINWING hind-wing length
 SPIRAC number of spiracles
 ANTSEG 1 length of antennal segment I
 ANTSEG 2 length of antennal segment II
 ANTSEG 3 length of antennal segment III
 ANTSEG 4 length of antennal segment IV
 ANTSEG 5 length of antennal segment V
 ANTSPIN number of antennal spines
 TARSUS 3 leg length, tarsus III
 TIBIA 3 leg length, tibia III
 FEMUR 3 leg length, femur III
 ROSTRUM rostrum
 OVIPOS ovipositor
 OVSPIN number of ovipositor spines
 FOLD anal fold
 HOOKS number of hind-wing hooks

From Renscher's *Methods of Multivariate analysis*

514

Correlation matrix for aphid properties

Table 12.2 Correlation Matrix for Winged Aphid Variables (Lower Triangle)

r1	1.000																			
r2	.934	1.000																		
r3	.927	.941	1.000																	
r4	.909	.944	.933	1.000																
r5	.524	.487	.542	.499	1.000															
r6	.799	.821	.856	.833	.703	1.000														
r7	.854	.865	.886	.889	.719	.923	1.000													
r8	.789	.834	.846	.885	.253	.499	-.751	1.000												
r9	.835	.863	.862	.850	.462	.752	-.793	.745	1.000											
r10	.645	.878	.803	.881	.507	.836	-.913	.797	.805	1.000										
r11	.458	-.486	-.522	-.488	-.174	-.317	-.383	-.497	-.356	-.371	1.000									
r12	.917	.942	.940	.945	.516	.846	.907	.861	.848	.902	.712	1.000								
r13	.859	.961	.956	.952	.404	.849	.914	.876	.877	.901	.458	.859	1.000							
r14	.953	.954	.946	.949	.452	.823	.886	.878	.883	.891	.517	.913	.953	1.000						
r15	.895	.899	.882	.908	.551	.831	.891	.794	.818	.848	.458	.859	.953	.895	1.000					
r16	.691	.652	.694	.623	.815	.812	.855	.410	.620	.712	.458	.859	.953	.895	.691	1.000				
r17	.327	.305	.356	.272	.746	.553	-.567	-.667	-.300	-.384	.458	.859	.953	.895	.327	.691	1.000			
r18	-.676	-.712	-.667	-.736	-.233	-.904	-.302	-.738	-.666	-.629	.458	.859	.953	.895	-.676	-.712	-.676	1.000		
r19	.702	.729	.746	.777	.285	.499	.392	-.793	.871	.668	.458	.859	.953	.895	.702	.729	.702	.702	1.000	

From Renscher's *Methods of Multivariate analysis*

515

Eigenvalues of correlation matrix

Table 12.3 Eigenvalues of the Correlation Matrix of the Winged Aphid Data

Component	Eigenvalue	Percent of Variance	Cumulative Percent
1	13.861	73.0	73.0
2	2.379	12.5	85.4
3	.748	3.9	89.4
4	.562	2.6	92.0
5	.278	1.4	93.5
6	.266	1.4	94.9
7	.193	1.0	95.9
8	.157	.8	96.7
9	.140	.7	97.4
10	.123	.6	98.1
11	.092	.4	98.6
12	.074	.4	99.0
13	.060	.3	99.3
14	.042	.2	99.5
15	.036	.2	99.7
16	.024	.1	99.8
17	.020	.1	99.9
18	.011	.1	100.0
19	.003	.0	100.0

From Renscher's *Methods of Multivariate analysis*

516

Eigenvectors of correlation matrix

Table 12.4 Eigenvectors for the First Four Components of the Winged Aphid Data

Variable	Eigenvectors			
	1	2	3	4
LENGTH	.96	-.06	.03	-.12
WIDTH	.98	-.12	.01	-.16
FORWING	.99	-.06	-.06	-.11
HINWING	.98	-.16	.03	-.00
SPRAC	.61	.74	-.20	1.00
ANTSEG 1	.91	.33	.04	.02
ANTSEG 2	.96	.30	.00	-.04
ANTSEG 3	.88	-.43	.06	-.18
ANTSEG 4	.90	-.08	.18	-.01
ANTSEG 5	.84	.05	.11	.03
ANTSPIN	-.49	.37	1.00	.27
TARSUS 3	.95	-.02	.03	-.29
TIBIA 3	1.00	-.05	.09	-.31
FEMUR 3	.99	-.12	.12	-.31
ROSTRUM	.96	.02	.08	-.06
GVIPOS	.76	.73	-.03	-.09
OVSPIN	-.41	1.00	-.16	-.06
FOLD	.64	.64	.04	-.80
HOOBS	-.76	-.52	.06	.72

y_{11} →

y_{18} →

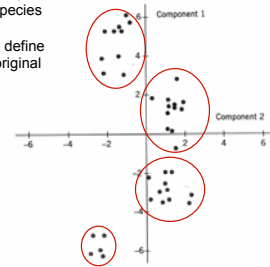
From Renscher's *Methods of Multivariate analysis*

517

Data points in the PC1 – PC2 plane

These clusters apparently represent 4 different species

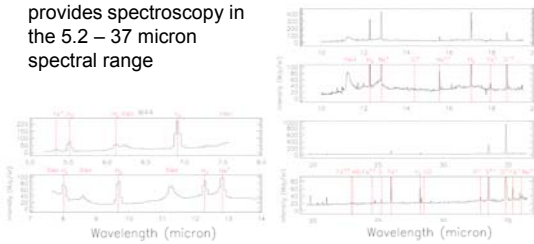
Would be very hard to define these in terms of the original 19 parameters



518

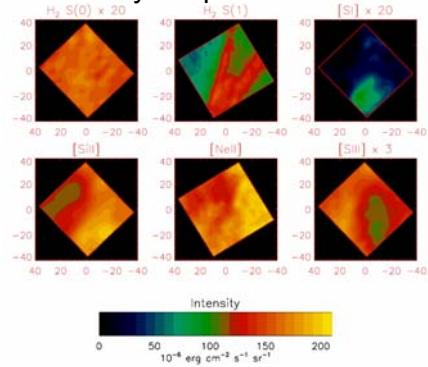
Example 2: comparing spectral line maps in astronomy

- Spitzer Space Telescope provides spectroscopy in the 5.2 – 37 micron spectral range



519

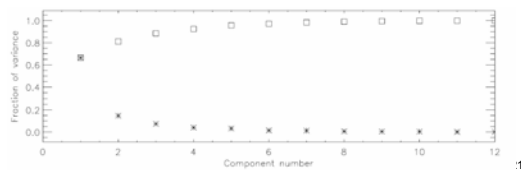
Spectral lines can be mapped and show a variety of spatial distributions



520

Principal component analysis

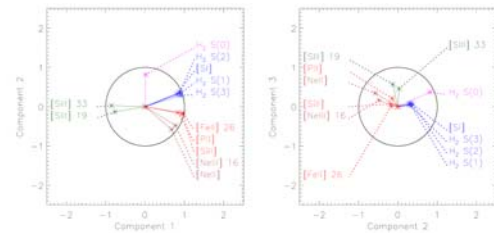
- Use PCA to write each map as a linear combination of orthogonal maps: M_1, M_2, M_3, \dots where M_1 accounts for most of the information, then M_2 etc. First two maps contain most of the information: 4th and higher components are noise dominated



1

PCA allows the various transitions to be grouped

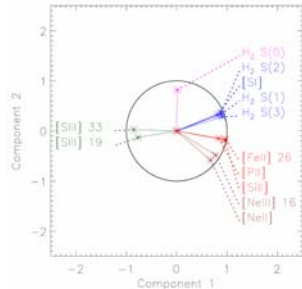
- Create an "h-plot", in which we show the coefficients A_{mk} for the first three maps ($m=1,2,3$)



PCA allows the various transitions to be grouped

Five distinct groups

- 1) H₂ S(0)
Cool neutral gas
- 2) Other H₂ lines and SI
Warm neutral gas
- 3) FeII, PII, SIII
Ionized gas
- 4) NeIII, NeII
Highly ionized gas
- 5) SIII
Highly-ionized gas at low density



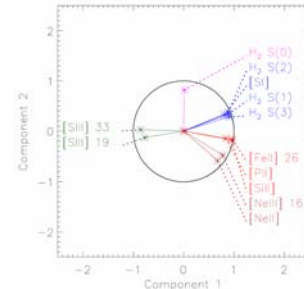
523

PCA allows the various transitions to be grouped

Note: cosine of angle between vectors represents correlation coefficient

$$\begin{aligned}
 R_{kr} &= A_{mk} A_{nr} R'_{nr} \\
 &= A_{mk} A_{nr} \delta_{mn} \\
 &= A_{mk} A_{mr} \\
 &= A_{1k} A_{1r} + A_{2k} A_{2r} = \cos \theta
 \end{aligned}$$

(if first two components dominate)



524

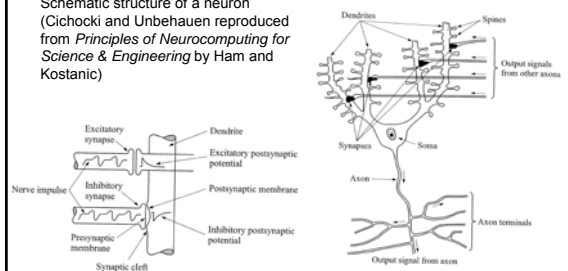
Lecture 23: Artificial neural networks

- Broad field that has developed over the past 20 to 30 years
- Confluence of statistical mechanics, applied math, biology and computers
- Original motivation: mathematical modeling of neurological networks
- Practical applications: pattern recognition (e.g. applied to speech, handwriting, underwriting)
 - used by USPS to read handwritten zip codes
 - Can be very fast, particularly when implemented in special purpose hardware

525

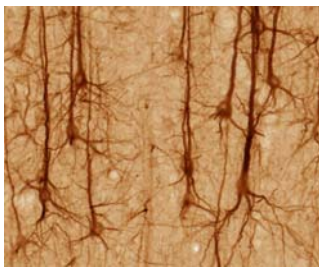
Biological neurons

Schematic structure of a neuron (Cichocki and Unbehauen reproduced from *Principles of Neurocomputing for Science & Engineering* by Ham and Kostanic)



526

Image of primate neurons

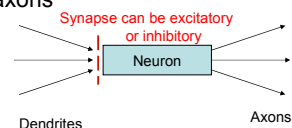


From brain-maps.org. Human brain contains ~ 10¹¹ neurons

527

Biological neurons

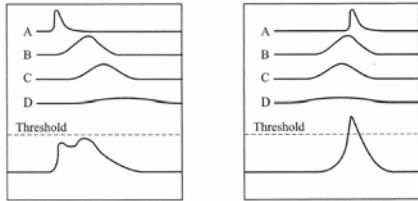
- Input signals come from the axons of other neurons, which connect to dendrites (input terminals) at the synapses
- If a sufficient excitatory signal is received, the neuron fires and sends an output signal along the axons



528

Threshold effect

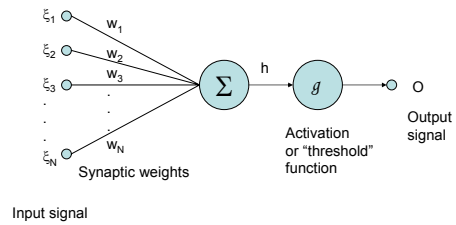
- The firing of the neuron occurs when a threshold excitation is reached



530

Mathematical model

- Nonlinear model of an artificial neuron



Input signal

Activation or "threshold" function

Output signal

530

Mathematical model

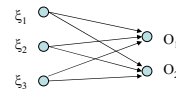
- Input and output signals are normalized, typically over the range $[-1, +1]$ or $[0, +1]$
- Activation function can be
 - linear: $g(h) = h$
 - step-like: $g(h) = \text{sgn}(h)$
 - sigmoid: $g(h) = \tanh(\beta h)$ or $1/(1+e^{-2\beta h})$
- Neuron output, $y = g(\sum(w_i x_i))$

531

Network architecture

- Here, we confine our attention to feed-forward networks (a.k.a. "perceptrons") \rightarrow no feedback loops: transfer of information is unidirectional

Simplest example: 1-layer perceptron with N inputs (ξ_k with $k = 1, N$) connected to M outputs (O_i with $i = 1, M$) via MN synaptic weights W_{ik}

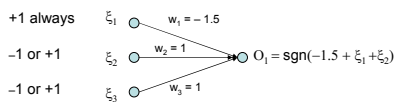


$$O_i = g\left(\sum_{k=1, N} W_{ik} \xi_k\right)$$

532

Example: the logical AND function

- Represent by a 1 x 3 perceptron



(example of setting thresholds with hardwired inputs)

533

Training the network

- How can we teach the network to yield desired responses?

- Need a set of desired inputs and outputs: ξ_i^μ and ζ_i^μ , for $\mu=1, p$
- Need a measure of learning: the "cost function", $E(\mathbf{w})$, that we seek to minimize

Matrix of synaptic weights, W_{ik}

- Finding the best set of weights is then an MN dimensional minimization problem

534

The cost function, E(w)

2 common choices

$$1) \quad E_{MSE}(\mathbf{w}) = \frac{1}{2} \sum_i \sum_\mu (\zeta_i^\mu - O_i^\mu)^2 = \frac{1}{2} \sum_i \sum_\mu \left(\zeta_i^\mu - g \left(\sum_k w_{ik} \zeta_k^\mu \right) \right)^2$$

(mean square error)

$$2) \quad E_{RE}(\mathbf{w}) = \frac{1}{2} \sum_i \sum_\mu \left((1 + \zeta_i^\mu) \ln \frac{(1 + \zeta_i^\mu)}{(1 + O_i^\mu)} + (1 - \zeta_i^\mu) \ln \frac{(1 - \zeta_i^\mu)}{(1 - O_i^\mu)} \right)$$

(relative entropy, for specific case of the tanh activation function)

535

Minimizing E(w)

- Simple training method: start with initial guess of weights and change in accord with

$$\Delta w_{ik} = -\eta \frac{\partial E}{\partial w_{ik}}$$

"Learning rate" ~ 0 to 1

(Move along direction of steepest descent)

536

Minimizing E(w)

- The derivatives are easy to compute

$$\frac{\partial E_{MSE}}{\partial w_{ik}} = - \sum_\mu (\zeta_i^\mu - O_i^\mu) g' \left(\sum_k w_{ik} \zeta_k^\mu \right) \zeta_k^\mu$$

which is conveniently written as $\frac{\partial E_{MSE}}{\partial w_{ik}} = - \sum_\mu \delta_i^\mu \zeta_k^\mu$

with $\delta_i^\mu = (\zeta_i^\mu - O_i^\mu) g' \left(\sum_k w_{ik} \zeta_k^\mu \right)$

and

$$g'(h) = 2\beta g(h)[1 - g(h)] \quad \text{for } g = \tanh(\beta h)$$

$$g'(h) = \beta [1 - g^2(h)] \quad \text{for } g = 1/(1 + e^{-2\beta h})$$

537

Minimizing E(w)

- For the RE cost function, we find that

$$\frac{\partial E_{RE}}{\partial w_{ik}} = - \sum_\mu \delta_i^\mu \zeta_k^\mu$$

where, for the tanh activation function,

$$\delta_i^\mu = (\zeta_i^\mu - O_i^\mu) \beta$$

538

Minimizing E(w)

- So the procedure is to start with an initial set of starting weights and to change them iteratively according to

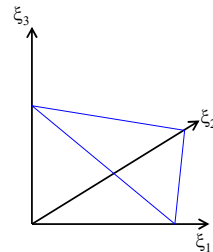
$$\Delta w_{ik}^\mu = \eta \delta_i^\mu \zeta_k^\mu$$

until the cost function changes by less than some preset amount.

539

Geometric interpretation

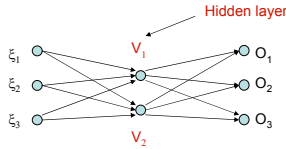
- For each output node, there is an N - 1 dimensional hyperplane which separates input values yielding $O > 0$ from those with $O < 0$



540

Multi-layer networks

- Interest in feed forward networks was limited until it was realized that 2 layer networks could describe any continuous function of the inputs



and a three layer network can describe any function of the inputs

541

Multi-layer networks

- Obviously, if the activation function is linear, the two-layer network is equivalent to a one-layer network with synaptic weights $q_{ik} = \sum_j w_{jk} W_{ij}$
- But for the sigmoid or step-function (sgn) activation function, interesting new behavior can result

542

Training multilayer perceptrons

- As before we minimize the cost function, e.g.

$$E_{MSE}(\mathbf{w}, \mathbf{w}) = \frac{1}{2} \sum_i \sum_\mu (\xi_i^\mu - O_i^\mu)^2$$

but now we have to vary two sets of weights.

- The derivatives w.r.t. W_{ik} are

$$\frac{\partial E_{MSE}}{\partial W_{ik}} = -\sum_\mu (\xi_i^\mu - O_i^\mu) g' \left(\sum_k W_{ik} V_k^\mu \right) V_k^\mu = -\sum_\mu \delta_i^\mu V_k^\mu$$

where

$$\delta_i^\mu \equiv (\xi_i^\mu - O_i^\mu) g'(H_i^\mu) \quad \text{and} \quad H_k^\mu \equiv \sum_k W_{ik} V_k^\mu$$

543

Training multilayer perceptrons

- The derivatives w.r.t. w_{jk} are computed using the chain rule

$$\begin{aligned} \frac{\partial E_{MSE}}{\partial w_{jk}} &= \sum_\mu \frac{\partial E}{\partial V_j^\mu} \frac{\partial V_j^\mu}{\partial w_{jk}} \\ &= -\sum_i \sum_\mu (\xi_i^\mu - O_i^\mu) g'(H_i^\mu) W_{ij} g'(h_j^\mu) \xi_j^\mu \end{aligned}$$

For each layer, we update the weights by moving along the direction of steepest descent:

$$\Delta W_{ik} = -\eta \frac{\partial E}{\partial W_{ik}} \quad \Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}}$$

544

Training multilayer perceptrons

- So the training procedure is
- Initialize all weights to random values
 - Propagate input signal **forwards** through network to compute the intermediate and output signals
 - Compute the cost function and its derivatives w.r.t. each weight, starting with the final layer and working **backwards**
 - Update all weights
 - Return to 2 (or stop if the convergence criterion is met)

545

Improvements in minimization

- As we know from previous lectures, the method of steepest descent can be very slow. We can use conjugate gradient or variable metric methods
- Alternatively, we can add a "momentum term" so that we include some of the previous step

$$\Delta W_{ik}|_{\text{new}} = -\eta \frac{\partial E}{\partial W_{ik}} + \alpha \Delta W_{ik}|_{\text{previous}}$$

where α is the momentum parameter (typically ~ 0.9 and must be between 0 and 1)

This can smooth the approach to the minimum. The best algorithms vary α and η as the minimum E is approached

546

Applications of ANN

- Underwriting
Input: information about borrower/insured
Output: loan/insurance outcome
Training set: previous experience
- Speech and handwriting recognition
- Financial predictions
Attempts to "beat" the stock market not successful:
consistent with the "efficient market" hypothesis
- Forecasting: weather, solar flares
- Diagnosis/classification: medical, astronomical

547

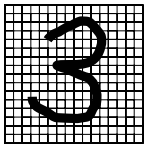
Flexible networks

- While the number of input and output nodes is generally fixed by the problem, the number of hidden layers and nodes within them can be varied to reduce the cost function
- 3-layer perceptrons are always sufficient, although the use of more layers may reduce the required number of nodes

548

Example: Handwriting recognition

- Input: 16 x 16 pixel image: $N = 256$



$\xi = 1$ for pixels where
ink is present, -1
otherwise
 $\rightarrow 2^{256} \sim 10^{77}$
possible input states

- Output: 10 nodes: $O_k = 1$ if number is k and -1 otherwise

549

Example: Handwriting recognition

Feed forward neural net developed by Le Cun et al. with four hidden layers
Training set: 10^4 images digitized from addresses on actual US mail
 28×28 grayscale pixels
Test set (not used for training): ~ 3000 additional images
4635 nodes, 98442 connections

550

Example: Handwriting recognition

Performance after 30 adaptation cycles:

- 1.1% error on training set
- 3.4% error on test set

Can achieve 1% error on test set if it rejects 5.7% of the characters

551

Example: photometric redshifts

- Use pattern recognition derive hard to measure parameter from observations of easy to measure parameter
- Sloan Digital Sky Survey provides broad-band photometric data (in 5 bands) for $\sim 10^8$ objects (mainly galaxies) and spectra for $\sim 10^6$

Spectra allow the redshift to be determined unequivocally, providing the distance and allowing the 3-D distribution to be determined

552

Example: photometric redshifts

- While only 1% of the objects are observed spectroscopically, “photometric redshifts” can be estimated for the other 99%
 - relative and absolute fluxes at 5 observed bands are correlated with z
- Collister and Lahav (2004, PASP, 116, 345) used artificial neural networks (e.g. 3 layer perceptron) to determine photometric redshifts: “ANNz” program

553

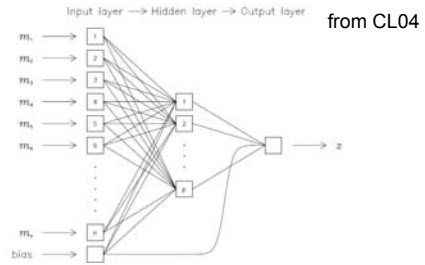


FIG. 1.— A schematic diagram of a multi-layer perceptron, as implemented by ANNz, with input nodes taking, for example, magnitudes $m_i = -2.5 \log_{10} f_i$ in various filters, a single hidden layer, and a single output node giving, for example, redshift z . The architecture is $n:p:1$ in the notation used in this paper. Each connecting line carries a weight w_{ij} . The bias node allows for an additive constant in the network function defined at each node. More complex networks can have additional hidden layers and/or outputs.

54

Example: photometric redshifts

- CL04 used a 3-layer perceptron with a 5:10:10:1 architecture
- Training set: 10^4 galaxies with spectroscopic redshifts
- Used “committee” of five independently trained networks
- Cost function modified to prevent blowup of weights

555

2.1. Network training

Given a suitable training set of galaxies for which we have both photometry, \mathbf{m} , and a spectroscopic redshift, z_{spec} , the ANN is trained by minimizing the *cost function*

$$E = \sum_k (z_{\text{phot}}(\mathbf{w}, \mathbf{m}_k) - z_{\text{spec},k})^2, \quad (2)$$

with respect to the weights, \mathbf{w} , where $z_{\text{phot}}(\mathbf{w}, \mathbf{m}_k)$ is the network output for the given input and weight vectors, and the sum is over the galaxies in the training set. To ensure that the weights are *regularized* (i.e. that they do not become too large), an extra quadratic cost term

$$E_w = \beta \sum_{i,j} w_{ij}^2, \quad (3)$$

is added to equation 2.

556

Example: photometric redshifts

- Results are quite impressive
- R.m.s. error in z is ~ 0.02 (versus 0.07 for competing method)

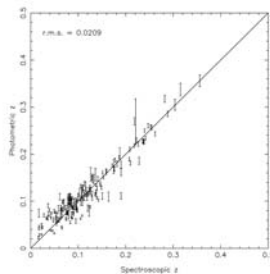


FIG. 3.— A subset of 200 galaxies randomly selected from the results of Fig. 2, and with the error bars calculated by ANNz shown. These are a combination of contributions from photometric noise (§2.2) and network variance (§2.3).