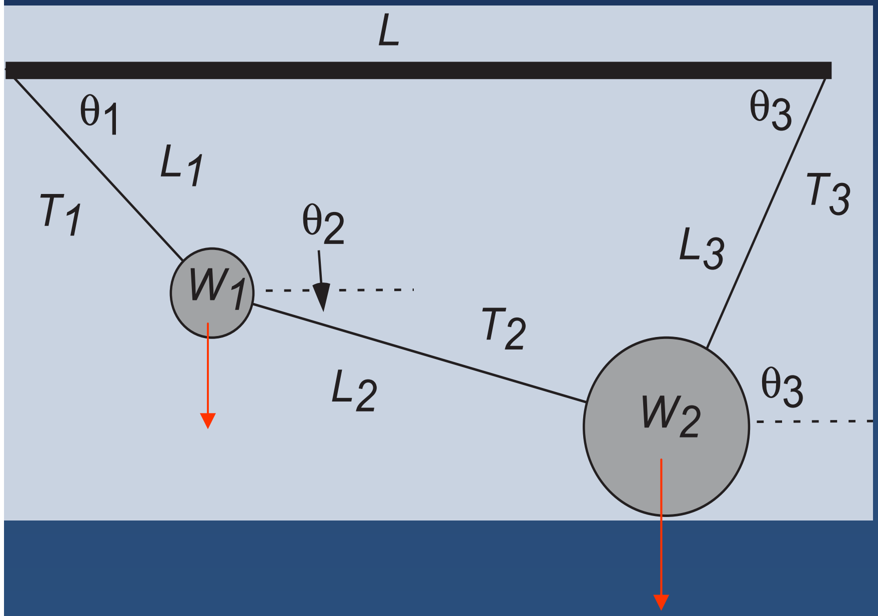
Weights on a String; Roots of Simultaneous Nonlinear Equations

1. Problem (6 unknowns):

$T_i = ?, \quad \theta_i = ?, \quad i = 1, 6$



(simple can be hard)

2. Geometric constraints:

$$
\begin{aligned}
L_1 \cos\theta_1 + L_2 \cos\theta_2 + L_3 \cos\theta_3 &= L \quad (1) \\
L_1 \sin\theta_1 + L_2 \sin\theta_2 - L_3 \sin\theta_3 &= 0 \quad (2)
\end{aligned}
$$

3. $\sum$ forces$_{x,y}$ = 0:

$$
\begin{aligned}
T_1 \sin\theta_1 - T_2 \sin\theta_2 - W_1 &= 0_{y,1} \quad (3) \\
T_1 \cos\theta_1 - T_2 \cos\theta_2 &= 0_{x,1} \quad (4) \\
T_2 \sin\theta_2 + T_3 \sin\theta_3 - W_2 &= 0_{y,2} \quad (5) \\
T_2 \cos\theta_2 - T_3 \cos\theta_3 &= 0_{x,2} \quad (6)
\end{aligned}
$$

4. Trigonometry (6 $\rightarrow$ 9 unknowns):

$$\sin^2\theta_i + \cos^2\theta_i = 1, \quad i = 1, 2, 3 \ \ (7-9)$$

Multi-D Newton-Raphson Search

- No analytic solution
- Can solve *f (x)=0*

- 9 simultaneous nonlinear equations
- Rename variables to vector [x]

$$[\mathbf{x}] = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} = \begin{bmatrix} \sin\theta_1 \\ \sin\theta_2 \\ \sin\theta_3 \\ \cos\theta_1 \\ \cos\theta_2 \\ \cos\theta_3 \\ T_1 \\ T_2 \\ T_3 \end{bmatrix},$$

$$\begin{aligned}
f_1(\mathsf{x}) &= 3x_4 + 4x_5 + 4x_6 - 8 = 0 \\
f_2(\mathsf{x}) &= 3x_1 + 4x_2 - 4x_3 = 0 \\
f_3(\mathsf{x}) &= x_7x_1 - x_8x_2 - 10 = 0 \\
f_4(\mathsf{x}) &= x_7x_4 - x_8x_5 = 0 \\
f_5(\mathsf{x}) &= x_8x_2 + x_9x_3 - 20 = 0 \\
f_6(\mathsf{x}) &= x_8x_5 - x_9x_6 = 0 \\
f_7(\mathsf{x}) &= x_1^2 + x_4^2 - 1 = 0 \\
f_8(\mathsf{x}) &= x_2^2 + x_5^2 - 1 = 0 \\
f_9(\mathsf{x}) &= x_3^2 + x_6^2 - 1 = 0
\end{aligned}$$

sin θ, cos θ = independent

Solve Matrix Equation for 9 Δx_j

(guess)

(roots) $$f_i(x_1^{n}, x_2^{n}, ..., x_9^{n}) \;=\; 0, \qquad x_i^{n} = x_i^{n-1} + \Delta x_i, \qquad (1)$$

$$f_i(x_1^{n}, x_2^{n}, ..., x_9^{n}) \;\simeq\; f_i(x_1^{n-1}, x_2^{n-1}, ..., x_9^{n-1}) + \sum_j^9 \frac{\partial f_i}{\partial x_j}(x_i^{n-1})\Delta x_j \qquad (2)$$

(linear approx)

$$f_i(x_1^{n-1}, x_2^{n-1}, ..., x_9^{n-1}) + \sum_j^9 \frac{\partial f_i}{\partial x_j}(\{x_i^{n-1}\})\,\Delta x_j = 0, \qquad (3)$$

(unknown)

- Matrix form: Standard form for linear equations

$$\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_9 \end{bmatrix}_o + \begin{bmatrix} \partial f_1/\partial x_1 & \cdots & \partial f_1/\partial x_N \\ \partial f_2/\partial x_1 & \cdots & \partial f_2/\partial x_9 \\ & \vdots & \\ \cdots & \cdots & \partial f_9/\partial x_9 \end{bmatrix}_o \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_9 \end{bmatrix} = 0 \qquad (4)$$

(known)

(unknowns)

$$\frac{\partial f_i}{\partial x_j} \simeq \frac{f_i(x_j + \Delta x_j) - f_i(x_j)}{\delta x_j} \qquad (5)$$

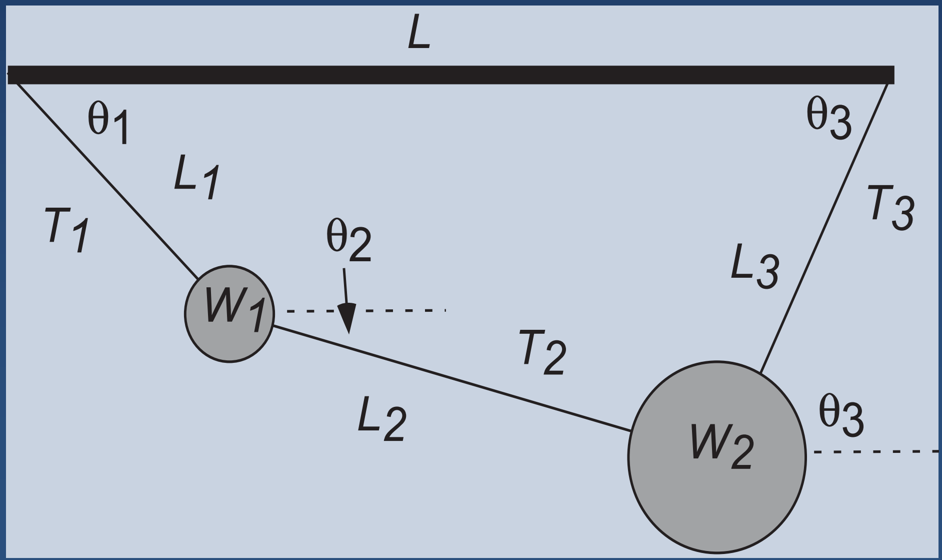
- Now use matrix library program (JAMA)

Assessment

1. Check reasonableness of W_1, W_2 solution
   a .various m, L
   b. deduced Tensions > 0, ≈ W
   c. deduced angles physical (sketch)

2. See how bad initial guess fails

3. * *3* masses (hard)



Relation of 1D and 9D Methods

- **1-D:**

$$\Delta x = -\frac{f}{f'} = -\frac{1}{f'} f \qquad (1)$$

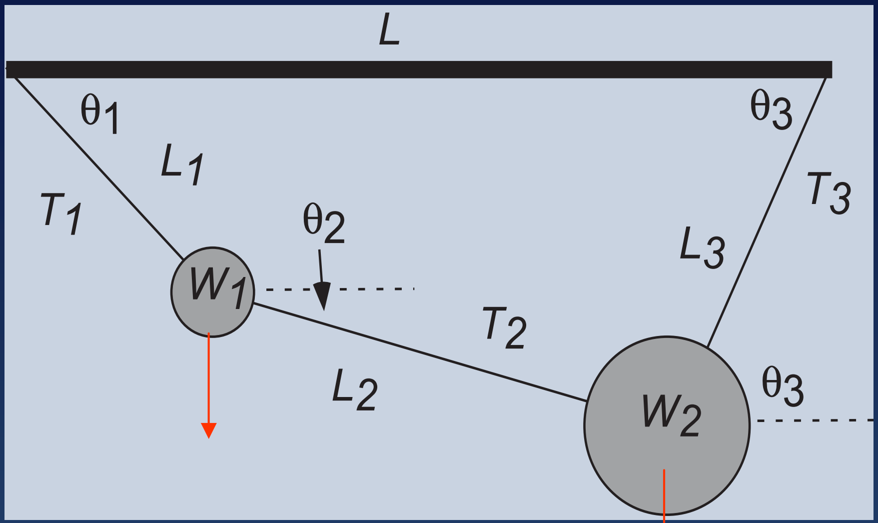
- **N-D: Linear Equations:**

$$\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_9 \end{bmatrix}_o + \begin{bmatrix} \partial f_1/\partial x_1 & \cdots & \partial f_1/\partial x_N \\ \partial f_2/\partial x_1 & \cdots & \partial f_2/\partial x_9 \\ & \vdots & \\ \cdots & \cdots & \partial f_9/\partial x_9 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_9 \end{bmatrix} = 0 \qquad (2)$$

- **Write solution as (formal)**

$$\begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_9 \end{bmatrix} = - \begin{bmatrix} \partial f_1/\partial x_1 & \cdots & \partial f_1/\partial x_N \\ \partial f_2/\partial x_1 & \cdots & \partial f_2/\partial x_9 \\ & \vdots & \\ \cdots & \cdots & \partial f_9/\partial x_9 \end{bmatrix}^{-1} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_9 \end{bmatrix} \qquad (3)$$

Recall 2 Weights on a String Problem



$$[\mathbf{x}] = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} = \begin{bmatrix} \sin\theta_1 \\ \sin\theta_2 \\ \sin\theta_3 \\ \cos\theta_1 \\ \cos\theta_2 \\ \cos\theta_3 \\ T_1 \\ T_2 \\ T_3 \end{bmatrix},$$

$$\begin{aligned}
f_1(\mathbf{x}) &= 3x_4 + 4x_5 + 4x_6 - 8 = 0 \\
f_2(\mathbf{x}) &= 3x_1 + 4x_2 - 4x_3 = 0 \\
f_3(\mathbf{x}) &= x_7x_1 - x_8x_2 - 10 = 0 \\
f_4(\mathbf{x}) &= x_7x_4 - x_8x_5 = 0 \\
f_5(\mathbf{x}) &= x_8x_2 + x_9x_3 - 20 = 0 \\
f_6(\mathbf{x}) &= x_8x_5 - x_9x_6 = 0 \\
f_7(\mathbf{x}) &= x_1^2 + x_4^2 - 1 = 0 \\
f_8(\mathbf{x}) &= x_2^2 + x_5^2 - 1 = 0 \\
f_9(\mathbf{x}) &= x_3^2 + x_6^2 - 1 = 0
\end{aligned}$$

Systems of Equations via Matrices

Solution

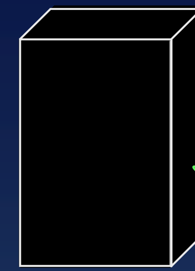
$$\begin{bmatrix} \partial f_1/\partial x_1 & \cdots & \partial f_1/\partial x_N \\ \partial f_2/\partial x_1 & \cdots & \partial f_2/\partial x_9 \\ & \vdots & \\ \cdots & \cdots & \partial f_9/\partial x_9 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_9 \end{bmatrix} = - \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_9 \end{bmatrix}_o \qquad \text{(1}$$

[?]

$$[A]\vec{x} = \vec{b} \qquad \text{(2}$$

- Many physical models $\Rightarrow$ simultaneous equations
- Place in matrix form, easier math (more abstract)
- More realistic models $\Rightarrow$ larger matrices
- Computer = excellent tool (same steps many times)

Scientific Subroutine Libraries



- Industrial strength, matrix subroutines
- > 10X faster than elementary methods
- Minimize roundoff error, failure
- Robust: high chance of success, broad class of problems
- Recommend: *do not write your own matrix subroutines*
- Also auto scales: desktop $\Rightarrow$ parallel cluster

What's the cost?

1. Must find them (not installed)
2. Must find names of all subroutines
3. May be Fortran only, C only

Classes of Matrix Problems (Math)

1. **Rules of math still apply!**
2. **N unknowns > N equations (unique)?**
3. **Equations not linearly independent?**
4. **N equations > N unknowns (fitting)?**
5. **Basic problem: system linear equations (2 masses)**

$$[A]\vec{x} = \vec{b} \qquad (1$$

$$[A]_{N\times N} \times \vec{x}_{N\times 1} = \vec{b}_{N\times 1} \qquad (2$$

- $[A]$ = known N x N matrix
- x = unknown length N vector
- b = known length N vector

Solution Linear Equations

$$[A]\vec{x} = \vec{b} \qquad (1$$

$$[A]_{N\times N} \times \vec{x}_{N\times 1} = \vec{b}_{N\times 1} \qquad (2$$

[?]

- "Best" solution: Gaussian elimination
- Triangular decomposition: no $[A]^{-1}$
- Slower, less robust: compute $[A]^{-1}$

$$[A]^{-1}[A]\vec{x} = [A]^{-1}\vec{b} \qquad [A]^{-1}\ (1$$

$$\vec{x} = [A]^{-1}\vec{b}$$

- Both methods in libes

Classes of Matrix Problems *(cont)*

Eigenvalue Problem

$$[A]\vec{x} = \lambda\vec{x}$$ *(1*

- Different matrix equation, not $[A]\vec{x} = \vec{b}$ *(2*
- x (vector), λ (number) **=** unknowns RHS
- No direct solution, $\exists$ for some λ
- When $\exists$?

Trivial solution $$([A] - \lambda[I])\,\vec{x} \quad = \quad 0$$ *(3*

$$\times\ ([A] - \lambda[I])^{-1} \qquad \Rightarrow \quad \vec{x} = 0$$ *(4*

Nontrivial solution $$\nexists\ \ ([A] - \lambda[I])^{-1}$$ *(5*

Secular Equation (Cramer's Rule) $$\det[\mathbf{A} - \lambda\mathbf{I}] = 0$$ *(6*

Evaluate det[ ] & Search

Practical Aspects of Matrix Computing

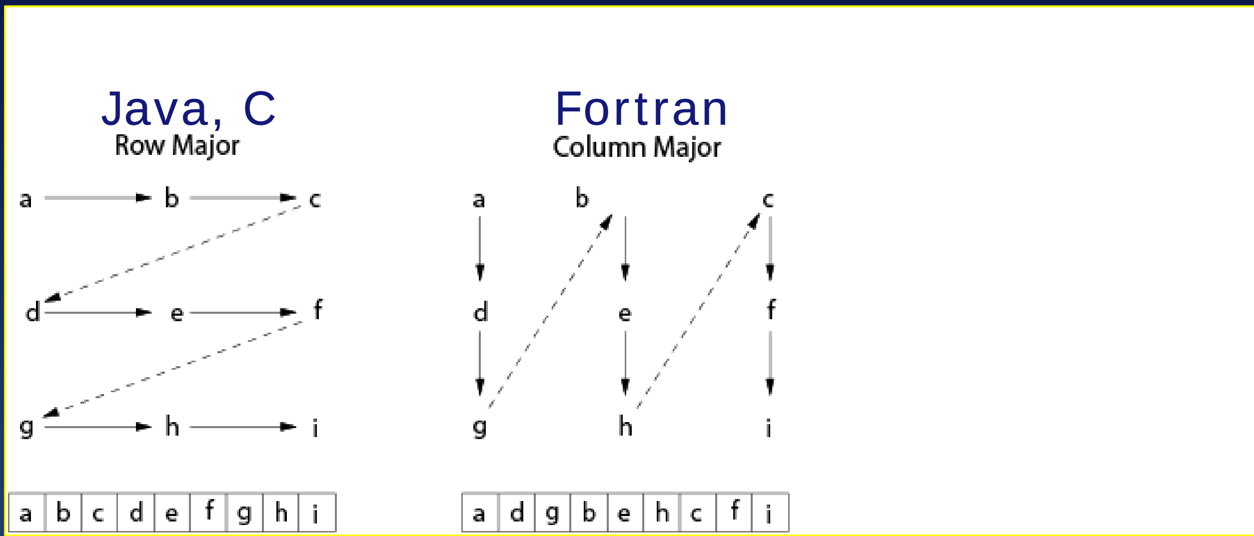
- Scientific programming bugs: often arrays
- Even vector V[N] = "array" (1-D)

$$\begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix}$$

- Rules of thumb
  - Computers are finite: size matters
  - Physical dimension of 100: **`A[100][100][100][100]`** ≈ 1GB
  - Processing time: ~ N^3 steps for **`A[N][N]`**
  - Double N $\Rightarrow$ 8X time
  - Avoid page faults: 1 word $\rightarrow$ entire page

Practical Aspects: Memory

- Matrix storage: we think blocks, computer stores linear



- Avoid large "strides"

- Don't have too many indices: V[L, Nre, Nspin, k, kp, Z, A] (1
  - $\rightarrow$ V1[k, kp], V2[k, kp], V3[k, kp] (2
- Subscript 0: math must match (count from 1 or 0?)

$$(l+1)P_{l+1} - (2l+1)xP_l + lP_{l-1} = 0$$ (3

- Physical vs logical dimensions
  - declared a[3][3], defined (') up to a[2][2]

a[1][1]' a[1][2]' a[1][3] a[2][1]' a[2][2]' a[2][3] a[3][1] a[3][2] a[3][3] (4

Implementation: Scientific Libraries, WWW

NETLIB	**WWW metalib of free math libraries**	**LAPACK**	**Linear Algebra pack**
JLAPACK	**LAPACK library in Java**	**SLATEC**	**Comprehensive Math & Stats**
ESSL	**Engr & Sci Lib (IBM)**	**IMSL**	**Intl Math & Stats**
CERNLIB	**European Cntr Nuclear Res**	**BLAS**	**Basic Linear Algebra Subs**
JAMA	**Java Matrix Lib**	**NAG**	**Numerical Algorithms Group (UK)**
Lapack++	**Linear Algebra pack in C++**	**ScaLAPACK**	**Distributed Memory LAPACK**
TNT	**C++ Template Numerical Toolkit**	**GNU GSL**	**Full Scientific Libe in C & C++**

Linear algebra	Matrix operations	Interpolation, fitting
Eigensystem analysis	Signal processing	Sorting and searching
Solution of linear eqns	Differential equations	Roots, zeros & extrema
Random-number ops	Statistical functions	Numerical quadrature

JAMA: Java Matrix Library

- JAMA = basic linear algebra package for Java
- Works well, natural, non-expert, free
- Jampack: complex matrices
- True **`Matrix`** objects; linear algebra, aligned elements
- e.g. [A] x = b

```
double[][] array = { {1.,2.,3}, {4.,5.,6.}, {7.,8.,10.} };
Matrix A = new Matrix(array);
Matrix b = Matrix.random(3,1);
Matrix x = A.solve(b);
Matrix Residual = A.times(x).minus(b);
Matrix Itest = A.inverse().times(A);          // Test inverse
```

JamaEigen.java: Eigenvalue Problem

```
import Jama.*;          import java.io.*;
public class JamaEigen {

  public static void main(String[] argv) {
  double[][] I = { {2./3,-1./4,-1./4}, {-1./4,2./3,-1./4},  {-1./4,-
1./4,2./3}};
  Matrix MatI = new Matrix(I);                              // Array →matrix
  System.out.print( "Input Matrix" );
  MatI.print (10, 5);                                         // Print matrix
  EigenvalueDecomposition E = new EigenvalueDecomposition(MatI);
double[] lambdaRe =  E.getRealEigenvalues();                 //   Eigens
 System.out.println("Eigenvalues: \t lambda.Re[]="+ lambdaRe[0]);
  Matrix V = E.getV();                                            // Vectors
  System.out.print("\n Matrix with column eigenvectors ");
  V.print (10, 5);
}}
```

Try These!

1) Find the inverse

$$[A] = \begin{bmatrix} +4 & -2 & +1 \\ +3 & +6 & -4 \\ +2 & +1 & +8 \end{bmatrix}$$

2) Check in both directions

$$[A][A]^{-1} = [A]^{-1}[A] = [I]$$

3) Verify

$$[A]^{-1} = \frac{1}{263} \begin{bmatrix} +52 & +17 & +2 \\ -32 & +30 & +19 \\ -9 & -8 & +30 \end{bmatrix}.$$

Try These (cont)!

4) Same [A], solve 3 sets simultaneous linear equations

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

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Know vector [b], Solve for [x], for 3 known [b]'s:

$$b_1 = \begin{bmatrix} +12 \\ -25 \\ +32 \end{bmatrix}, \quad b_2 = \begin{bmatrix} +4 \\ -10 \\ +22 \end{bmatrix}, \ b_3 = \begin{bmatrix} +20 \\ -30 \\ +40 \end{bmatrix}.$$

Solution

$$x_1 = \begin{bmatrix} +1 \\ -2 \\ +4 \end{bmatrix}, \ x_2 = \begin{bmatrix} +0.312 \\ -0.038 \\ +2.677 \end{bmatrix}, \ x_3 = \begin{bmatrix} +2.319 \\ -2.965 \\ +4.790 \end{bmatrix}.$$

Some Eigenvalue Problems $[A]\vec{x} = \lambda\vec{x}$

5)

$$[A] = \begin{matrix} \alpha & \beta \\ -\beta & \alpha \end{matrix} \Bigg]$$

(normalization = ?)

$$\vec{x}_{1,2} = [\ +1 \text{¨}\ \ i\ ], \qquad \lambda_{1,2} = \alpha \text{¨}\ \ i\beta$$

6) Multiple eigenvalues (degeneracy)

$$\mathbf{A} = \begin{bmatrix} -2 & +2 & -3 \\ +2 & +1 & -6 \\ -1 & -2 & +0 \end{bmatrix}$$

$$\Rightarrow \quad \lambda_1 = 5 \qquad \vec{x}_1 = \frac{1}{\sqrt{6}} \begin{bmatrix} -1 \\ -2 \\ +1 \end{bmatrix}$$

$\Rightarrow \quad \lambda_2 = \lambda_3 = -3$ Double root $\Rightarrow$ combo of eigenvectors

$$\vec{x}_2,\ \vec{x}_3 = \frac{1}{\sqrt{5}} \begin{bmatrix} -2 \\ +1 \\ +0 \end{bmatrix}, \quad \frac{1}{\sqrt{10}} \begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix}.$$

Solve N = 100 Linear Equations for $\vec{y}$

$$
\begin{aligned}
a_{11}y_1 + a_{12}y_2 + \cdots + a_{1N}y_N &= b_1, \\
a_{21}y_1 + a_{22}y_2 + \cdots + a_{2N}y_N &= b_2, \\
\cdots & \\
a_{N1}y_1 + a_{N2}y_2 + \cdots + a_{NN}y_N &= b_N
\end{aligned}
$$

$$[a]\vec{y} = \vec{b}$$

[a] = Hilbert matrix

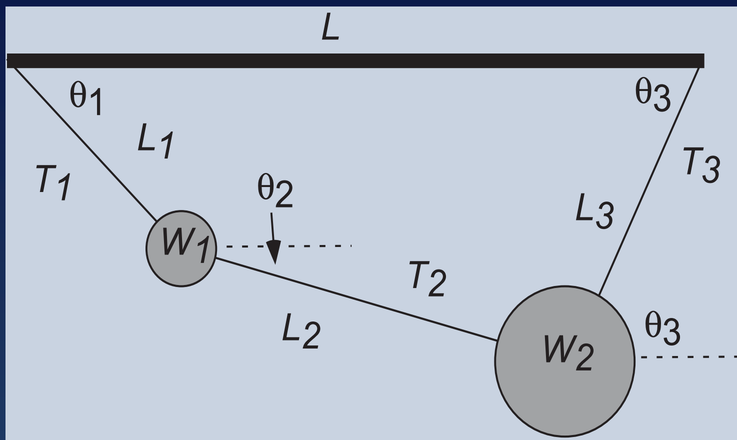
$$
[a_{ij}] = \left[\frac{1}{i+j-1}\right] = \begin{bmatrix}
1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{100} \\
\frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{101} \\
\ddots & & & & & \\
\frac{1}{100} & \frac{1}{101} & \cdots & & \cdots & \frac{1}{199}
\end{bmatrix}
$$

[b] = 1st row

$$
\mathbf{b} = \left[\frac{1}{i}\right] = \begin{bmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \\ \ddots \\ \frac{1}{100} \end{bmatrix}
$$

Verify $\Rightarrow$ $\begin{bmatrix} y_1 \\ y_2 \\ \ddots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \ddots \\ 0 \end{bmatrix}$

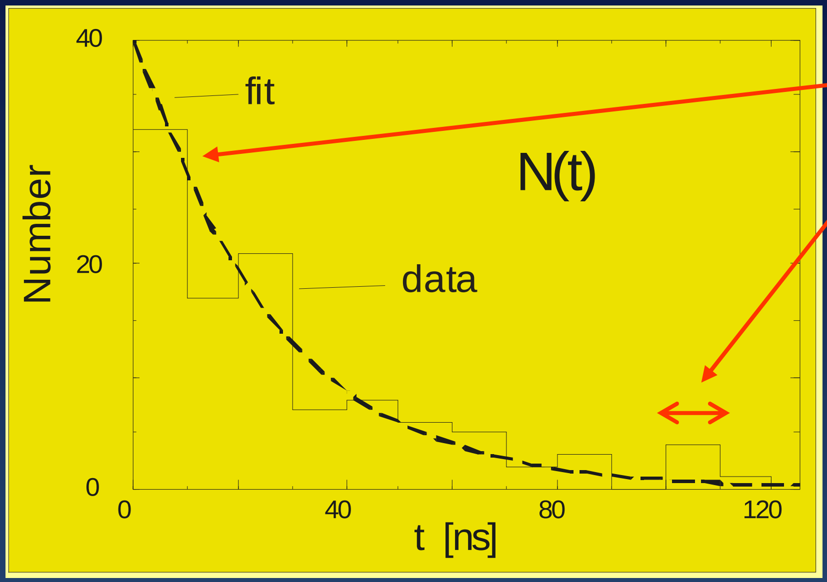
At last! Solve Masses on String



$$\begin{bmatrix} \partial f_1/\partial x_1 & \cdots & \partial f_1/\partial x_N \\ \partial f_2/\partial x_1 & \cdots & \partial f_2/\partial x_9 \\ & \vdots & \\ \cdots & \cdots & \partial f_9/\partial x_9 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_9 \end{bmatrix} = - \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_9 \end{bmatrix}_o$$

1. Is solution physical?
2. Try various weights & lengths
3. Deduced tensions > 0, proportional to weights?
4. sin θ, cos θ : sensible (sketch)?
5. Determine when initial guess not close enough.
6. Solve similar 3-m problem*.

Problem: Fitting Exponential Decay

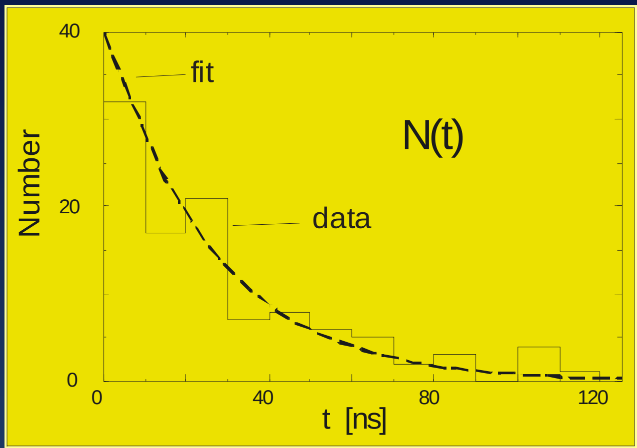


- Number $=\Delta$ N = # of π meson decays
- Time in Δ **t = 10** ns "bins"
- Curve = theoretical fit (your problem)
  - deduce π meson lifetime τ
  - tabulated τ **= 2.6 10^{-8} s**

- Theory: spontaneous decay (stochastic)
- Model: Exponential decay **(Δ t $\rightarrow$ 0)**

$$\frac{dN(t)}{dt} = \frac{dN(0)}{dt} e^{-t/\tau} \qquad (1)$$

Method: Least-Squares Fitting 1



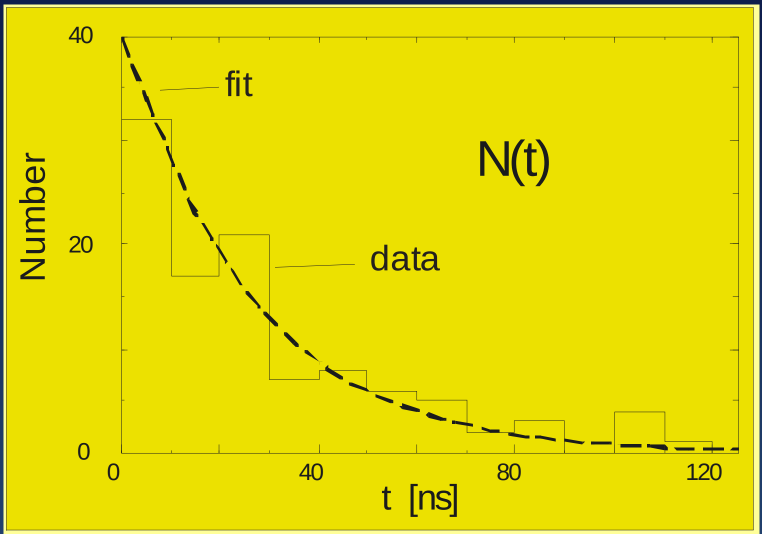
- Meaning "good" fit to data?
- Statistics = big subject (see refs)
- Three points to remember
  1. if errors, theory not pass through all
  2. if theory wrong, "best" fit terrible
  3. most best fits via search

Theory

$$\frac{dN(t)}{dt} = \frac{dN(0)}{dt} e^{-t/\tau} \qquad (1)$$

- ***y = g (x)*** describes data
- ***g (x) = g (x, a_1, a_2, … a_M) = g (x, $\vec{a}$)***
  - ***a_1, a_2, … a_M*** = parameters, part of theory
  - ***a_1*** = lifetime τ, ***a_2***= initial rate ***dN(0)/dt***

Least-Squares Fitting 2



$$\frac{dN(t)}{dt} = \frac{dN(0)}{dt}\, e^{-t/\tau} \qquad (1)$$

1. $\boldsymbol{y = g\,(x, a_1, a_2, \ldots\ a_M)}$ = Eq.(1) describes data
2. Given: N_D data values
3. $\boldsymbol{(\,x_i,\ y_i \pm \sigma_i\,),\quad i = 1,\ N_D = \#}$ data points
4. $\boldsymbol{y_i}$ = independent variable: $\boldsymbol{t}$
5. $\boldsymbol{x_i}$ = dependent variable: $\Delta\, \boldsymbol{N\,(t)}$
6. $\pm\,\sigma_i$ = uncertainty ("error") in $\boldsymbol{y_i}$

χ^2 = Measure of Goodness (the "square")

$$\chi^2 \stackrel{\text{def}}{=} \sum_{i=1}^{N_D} \left[\frac{y_i - g(x_i, \vec{a})}{\sigma_i} \right]^2 \tag{2}$$

- χ**2** = summed squares deviations of data from ***g (x)***
- $\Rightarrow$ smaller χ**2** $\Rightarrow$ better fit
- **1 /** σ_i^2 = weighting $\Rightarrow$ large errors contribute least
- Least-squares = "best" fit
- "Fit" = adjust a_i $\Rightarrow$ minimum χ**2** (the "least")
- Determine parameters in theory
- χ**2** $\cong$ $N_D - M_P$ **= #** degrees freedom, good
- Good fit: misses ~ 1/3 points
- χ**2 = 0** $\Rightarrow$ theory passes thru all data points

Equations for Best Fit

$$\chi^2 \stackrel{\text{def}}{=} \sum_{i=1}^{N_D} \left[\frac{y_i - g(x_i, \vec{a})}{\sigma_i} \right]^2 \qquad (1)$$

Solve for a_m that make $\chi 2$ a minimum:

$$\frac{\partial \chi^2}{\partial a_m} = 0 \qquad (m = 1, M_P) \qquad (2)$$

$$\Rightarrow \quad \sum_{i=1}^{N_D} \frac{[y_i - g(x_i)]}{\sigma_i^2} \frac{\partial g(x_i)}{\partial a_m} = 0 \qquad (3)$$

Means

- $\Rightarrow$ ***M_P*** simultaneous equations (nonlinear) in ***a_m***
- Work them out for each case
- Solution: trial-and-error search in ***M_P*** dimensions
- Need check: χ**2** = minimum, global minimum
- Different starting values $\Rightarrow \Delta$ min

E.G. Linear Least-Square Fit

$$g\left(x;\vec{a}\right) = a_1 + a_2 x \tag{4}$$

- Fitted function $g\,(x, \vec{a}\,)$ linear in parameter $\boldsymbol{a_i}$
- Here straight line too!
- Simple (analytic) solution ("Linear regression")
- M_P = 2 parameters: slope $\boldsymbol{a_2}$, $\boldsymbol{y}$ intercept $\boldsymbol{a_1}$
- *N.B.* still $\boldsymbol{N_D} \geq \boldsymbol{M_P} = \mathbf{2}$ of data points to fit
  - Unique solution: More data than unknowns
- Straight line (4) $\Rightarrow$ 2 derivatives for min $\chi\mathbf{2}$:

$$\frac{\partial g(x_i)}{\partial a_1} = 1, \qquad \frac{\partial g(x_i)}{\partial a_2} = x_i \tag{5}$$

- Solve $\chi\mathbf{2}$ minimization equations, ...

E.G. Linear Least-Square Fit

$$g\left(x;\vec{a}\right) = a_1 + a_2 x \tag{6}$$

Solve $\chi 2$ minimization equations (algebra):

$$a_1 = \frac{S_{xx}S_y - S_x S_{xy}}{\Delta}, \qquad a_2 = \frac{SS_{xy} - S_x S_y}{\Delta} \tag{7}$$

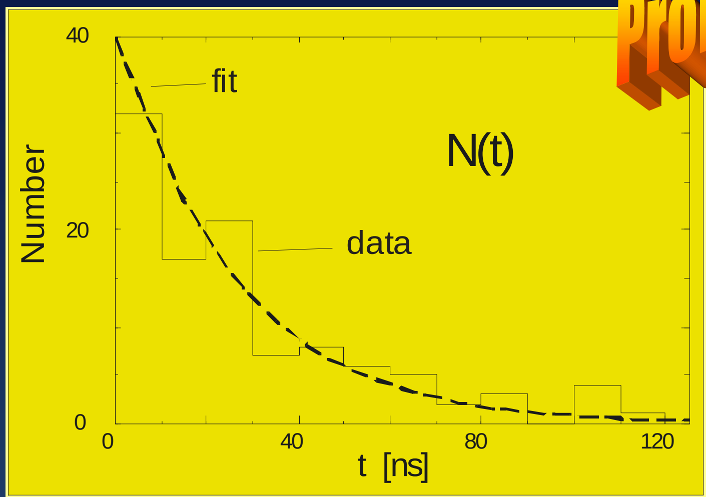
$$S = \sum_{i=1}^{N_D} \frac{1}{\sigma_i^2}, \qquad S_x = \sum_{i=1}^{N_D} \frac{x_i}{\sigma_i^2}, \tag{8}$$

$$S_{xy} = \sum_{i=1}^{N_D} \frac{x_i y_i}{\sigma_i^2}, \qquad \Delta = SS_{xx} - S_x^2 \tag{9}$$

Variance = uncertainty in parameters

$$\sigma_{a_1}^2 = \frac{S_{xx}}{\Delta}, \qquad \sigma_{a_2}^2 = \frac{S}{\Delta} \tag{10}$$

Assessment: Fit to Exponential Decay



Problem

1. Fit exponential decay law to data in Fig
2. Find best values for *dN/dt(0)* and τ
3. Judge how good the fit is

Implement

1. Construct table **(** Δ $\boldsymbol{N_i}$, Δ $\boldsymbol{t_i}$**)** from figure
2. Estimate error σ_i by "eye" or $\sqrt{N_i}$
3. Fit the semilog plot

$$\frac{dN(t)}{dt} = \frac{dN(0)}{dt}\, e^{-t/\tau} \qquad (1)$$

$$\ln\left|\frac{\Delta N(t)}{\Delta t}\right| \simeq -\frac{1}{\tau}\Delta t + \ln\left|\frac{\Delta N_0}{\Delta t}\right| \qquad (2)$$

$$y = ax + b \qquad (3)$$

4. Plot best fit with data and compare

Extension: Linear Quadratic Fit

$$g(x) = a_1 + a_2 x + a_3 x^2 \qquad (1)$$

- Recall: *linear* fit ⇒ linear in parameters **a_i** not straight line!
- Nonlinear: **g(x) = (a_1 + a_2x) e $^{a\,x}$**
- Parabola: min χ**2** ⇒ **3** simultaneous equations

$$\sum_{i=1}^{N_D} \frac{[y_i - g(x_i)]}{\sigma_i^2} \frac{\partial g(x_i)}{\partial a_1} = 0, \qquad \frac{\partial g}{\partial a_1} = 1, \qquad (2)$$

$$\sum_{i=1}^{N_D} \frac{[y_i - g(x_i)]}{\sigma_i^2} \frac{\partial g(x_i)}{\partial a_2} = 0, \qquad \frac{\partial g}{\partial a_2} = x, \qquad (3)$$

$$\sum_{i=1}^{N_D} \frac{[y_i - g(x_i)]}{\sigma_i^2} \frac{\partial g(x_i)}{\partial a_3} = 0, \qquad \frac{\partial g}{\partial a_3} = x^2. \qquad (4)$$

Solution: Linear Quadratic Fit

$$g(x) = a_1 + a_2 x + a_3 x^2 \tag{1}$$

3 eqs, 3 unknowns

$$\begin{aligned} S a_1 + S_x a_2 + S_{xx} a_3 &= S_y, &(2)\\ S_x a_1 + S_{xx} a_2 + S_{xxx} a_3 &= S_{xy}, &(3)\\ S_{xx} a_1 + S_{xxx} a_2 + S_{xxxx} a_3 &= S_{xxy} &(4)\end{aligned}$$

$$e.g.\quad S_{xy} = \sum_{i=1}^{N_D} \frac{x_i y_i}{\sigma_i^2} \tag{5}$$

JamaFit.java

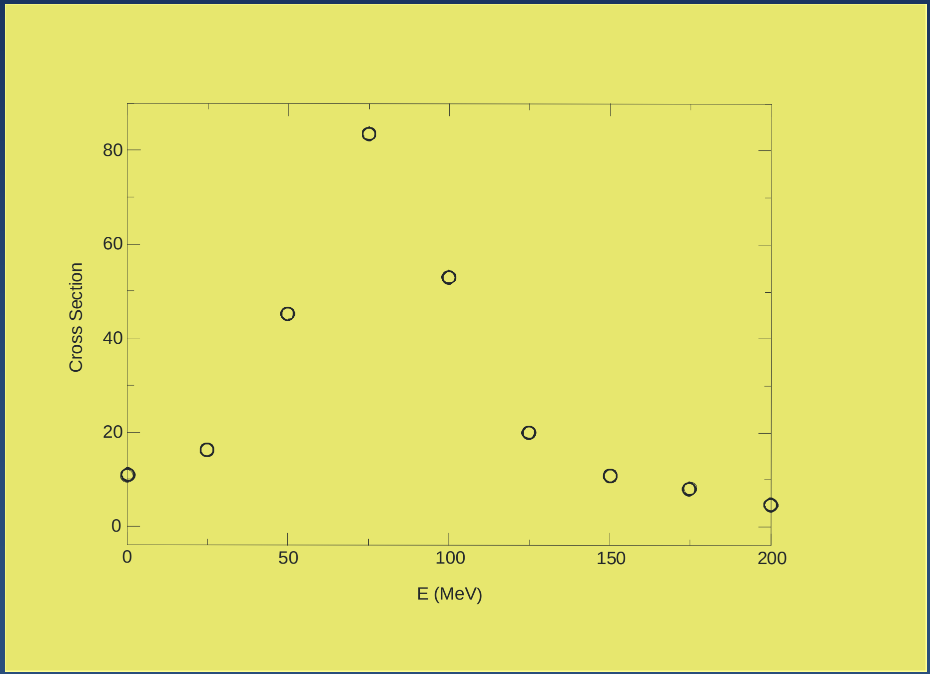
$$[\alpha]\vec{a} = \vec{\beta} \tag{6}$$

$$[\alpha] = \begin{bmatrix} S & S_x & S_{xx} \\ S_x & S_{xx} & S_{xxx} \\ S_{xx} & S_{xxx} & S_{xxxx} \end{bmatrix} \tag{7}$$

$$\vec{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}, \qquad \vec{\beta} = \begin{bmatrix} S_y \\ S_{xy} \\ S_{xxy} \end{bmatrix} \tag{8}$$

Nonlinear Fit to Cross Section (challenge)

i	1	2	3	4	5	6	7	8	9
E_i	0	25	50	75	100	125	150	175	200
$f(E_i)$	10.6	16.0	45.0	83.5	52.8	19.9	10.8	8.25	4.70
$+-\sigma_i$	9.34	17.9	41.5	85.5	51.5	21.5	10.8	6.29	4.14



$$f(E) = \frac{f_r}{(E - E_r)^2 + \Gamma^2/4}$$

- 3 parameters (OK)
- Non linear in *E* (hard)
- Newton-Raphson Search