Computational Methods (PHYS 2030)

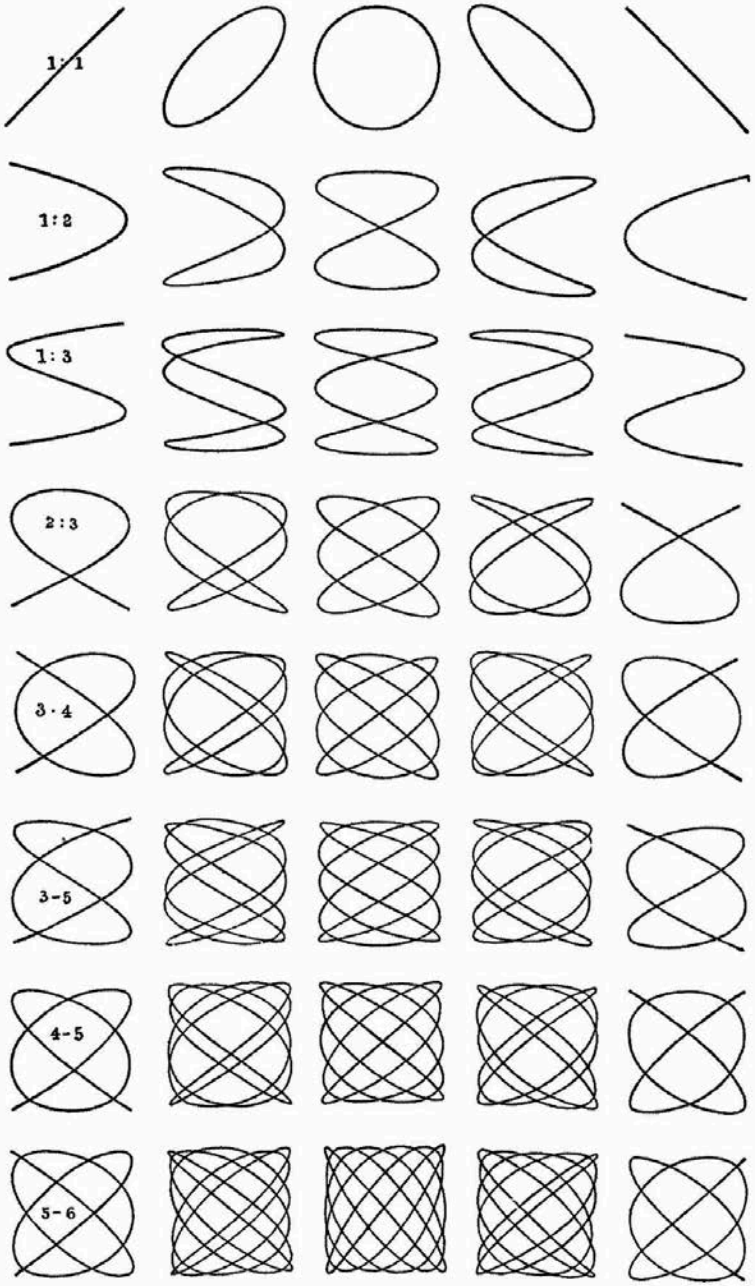
Instructors: Prof. Christopher Bergevin (cberge@yorku.ca)

Schedule: Lecture: MWF 11:30-12:30 (CLH M)

Website: http://www.yorku.ca/cberge/2030W2018.html

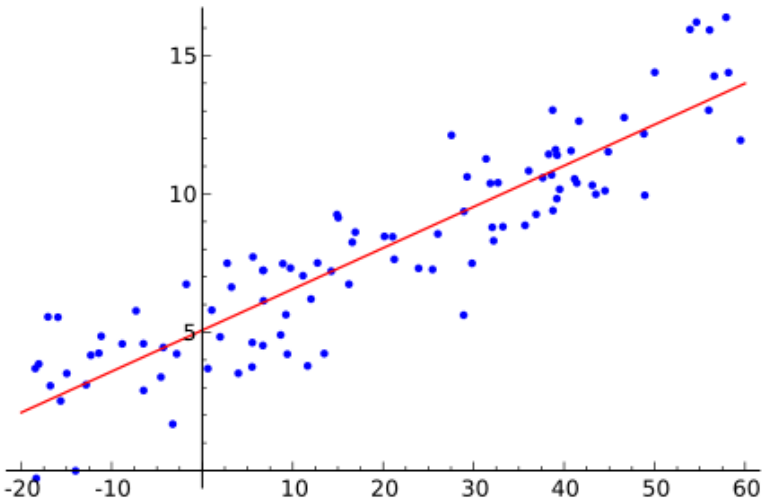
York University
Winter 2018
Lecture 14

Sound Vibrations



Regression

- Etymological roots stem from Francis Galton and the biological notion to regress down towards an average value
  (‘regression towards the mean’)



(Very) common application:
Fitting a straight line to data (linear regression)

- Very general/powerful concept, manifests in many scientific and engineering applications

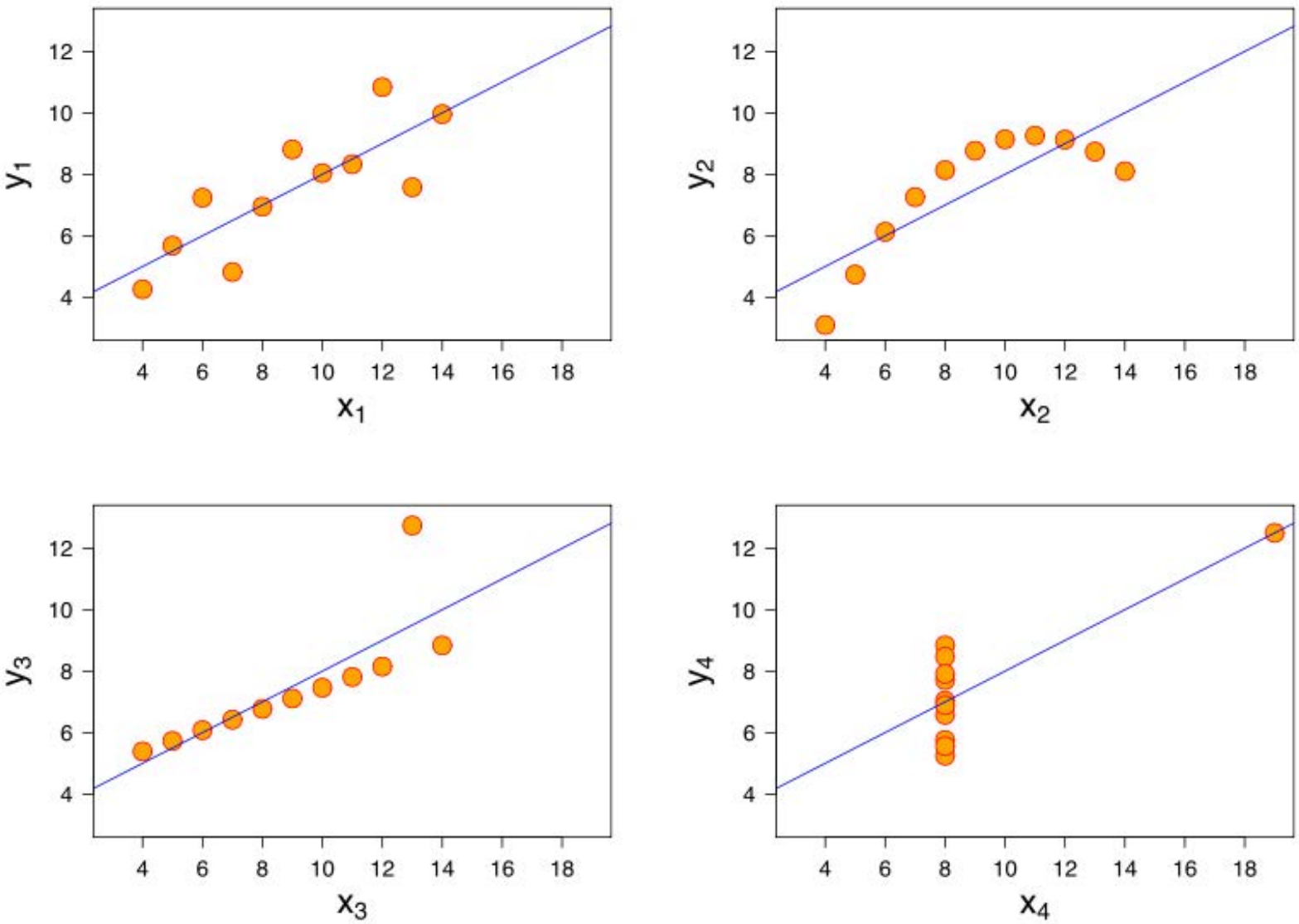
- We will initially focus on a *parametric* method known at *least-squares analysis*

Important point #1: Regression analysis typically involves ‘modeling’ in that one commonly has assumed a model they are trying to fit to the data

Important point #2: Essentially an optimization problem

wikipedia (regression analysis)

Aside: Anscombe's quartet

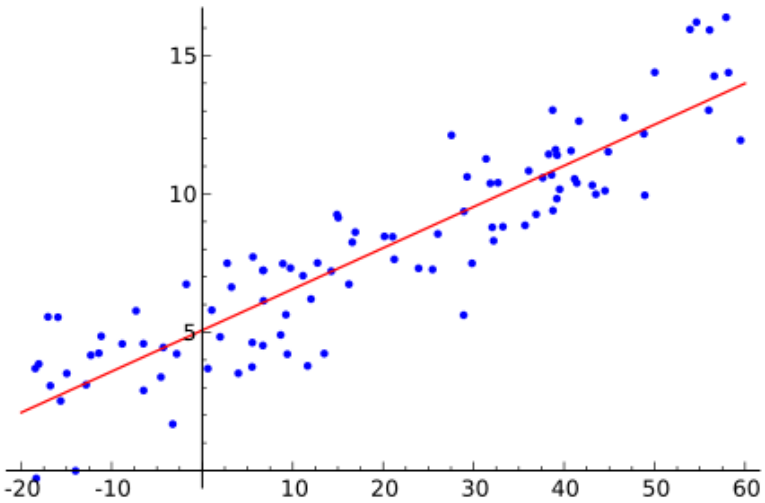


Important point #3: Be smart about how you handle data and make analysis decisions!

wikipedia (Anscomb's quartet)

Linear regression

- Useful starting point:
  - intuitive
  - has an exact solution
  - easy to implement numerically



→ Natural foundation for more advanced topics (e.g., nonlinear regression, non-parametric regression, bootstrapping)

- In this case, we have some (2-D) 'data' and our 'model' is simply a linear function

$$y = a + bx$$

- the data form x_i (independent var.) and y_i (dependent var.)
- the model is described by $y(x)$
- goal is to determine the best values of a and b

- A key quantity in 'least squares' analysis is χ^2 ("chi-squared")

We'll return to this shortly

Basic statistical considerations (we'll need these later)

Note: There is a deeper mathematical/statistical theory here (e.g., probability distributions, Method of Maximum Likelihood) we are only scratching the surface of

- Assume we have some set of data points X:

$$x_i \in X$$

$$i = 1, 2, \cdots, N$$

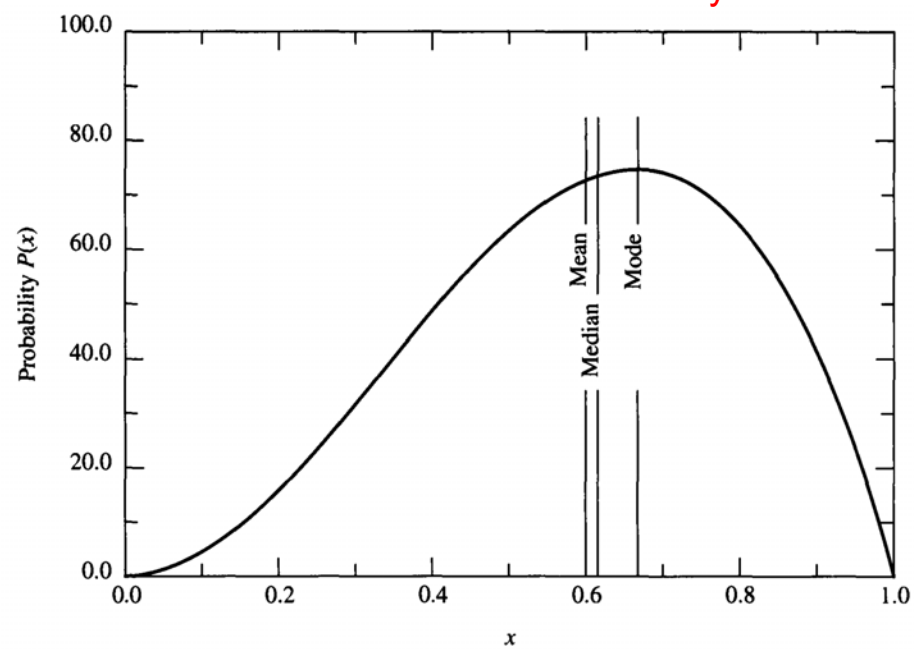
- Sigma notation: $$\sum x_i \equiv \sum_{i=1}^{N} x_i$$

- Then the **mean** of X is: $$\bar{x} \equiv \frac{1}{N} \sum x_i$$

- In the limit of large numbers, the notion of a parent distribution emerges:

$$\mu \equiv \lim_{N \to \infty} \left(\frac{1}{N} \sum x_i \right)$$

Median and Mode also commonly used



Bevington (2003)

Basic statistical considerations (we'll need these later)

- Note that in the real world, we can only have a finite number of points, so the notion of a parent distribution is an ideal one (we deal with the 'sample distribution')

$$\mu \equiv \lim_{N\to\infty}\left(\frac{1}{N}\sum x_i\right)$$

$$(\text{parent parameter}) = \lim_{N\to\infty}(\text{experimental parameter})$$

Basic rules of thumb – Try to make sure:

- Samples are random
- N is large

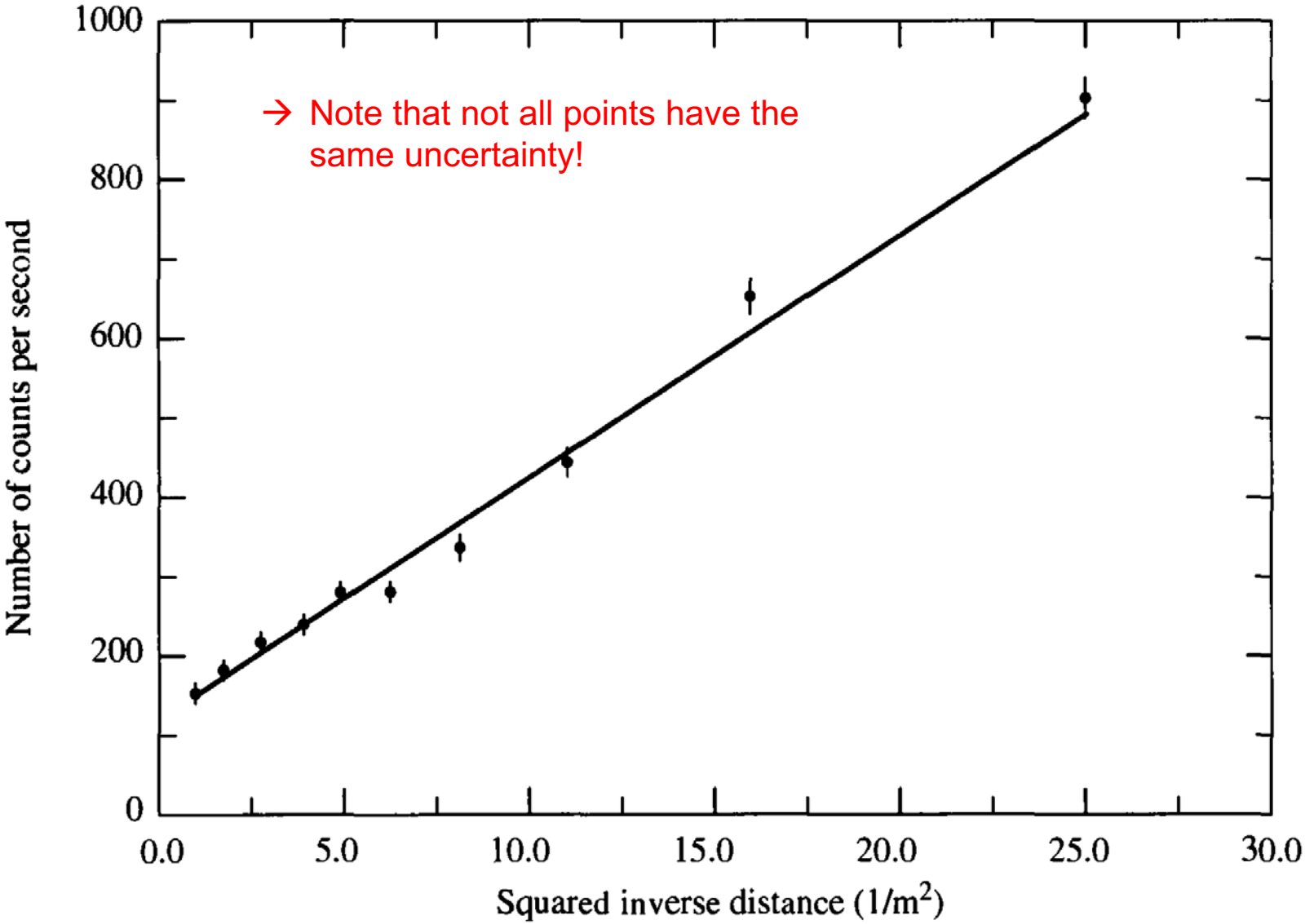
- Standard deviation – Tells us how much a given point 'deviates' from the average

$$\sigma^2 \equiv \lim_{N\to\infty}\left[\frac{1}{N}\sum(x_i-\mu)^2\right] = \lim_{N\to\infty}\left(\frac{1}{N}\sum x_i^2\right) - \mu^2$$

When N is finite:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\bar{x})^2}$$

Linear regression: Basic ideas



→ Any method we develop to determine the best fit should take into account that certain points might be 'weighted' differently

Bevington (2003)

Least squares

- We measure y_i and want to determine a function $y(x)$ such that we have a predicted value $y(x_i)$

- Deviations between observed value [y_i] and predicted value [$y(x_i)$] is Δy_i. For a linear function to fit, this is then

$$\Delta y_i = y_i - y(x_i) = y_i - a - bx_i$$

→ Goal is to determine the best values of a and b so to minimize Δy_i

- Assume underlying probability distribution is Gaussian:

$$P_i = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left\{-\frac{1}{2}\left[\frac{y_i - y_0(x_i)}{\sigma_i}\right]^2\right\}$$

Gaussian parent distributions are very common/important in physics!

That is

We shall assume that each individual measured value of y_i is itself drawn from a Gaussian distribution with mean $y_0(x_i)$ and standard deviation σ_i.

Bevington (2003)

Least squares

Various error measurements can be minimized when approximating with a given function $f(x)$. Three standard possibilities are given as follows

I. MaximumError :

$$E_\infty(f) = \max_{1<k<n} |f(x_k) - y_k|. \tag{3.1.4a}$$

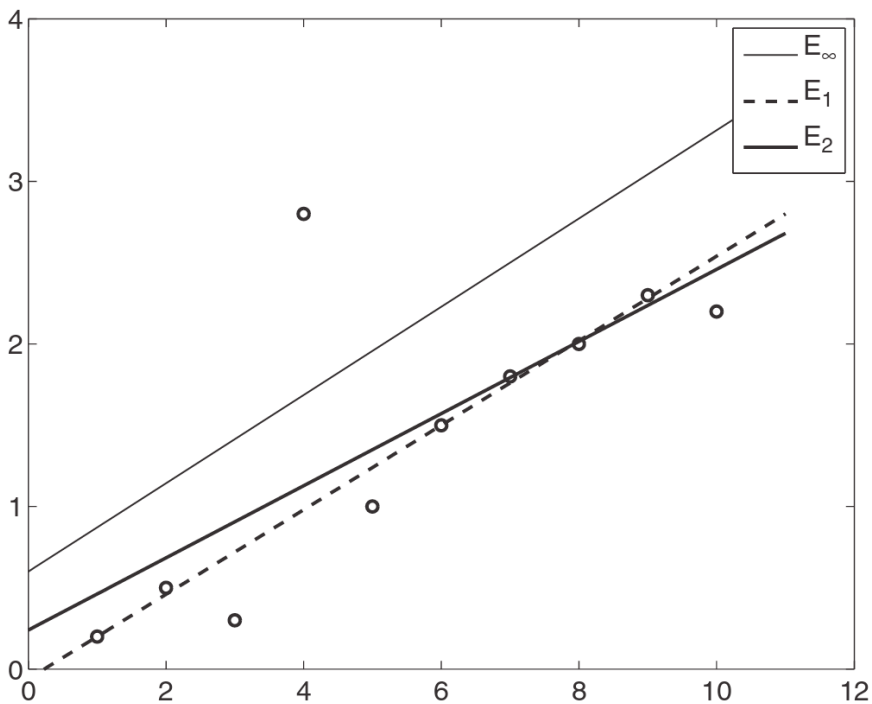
II. AverageError :

$$E_1(f) = \frac{1}{n}\sum_{k=1}^{n} |f(x_k) - y_k|. \tag{3.1.4b}$$

III. Root-meanSquare :

$$E_2(f) = \left(\frac{1}{n}\sum_{k=1}^{n} |f(x_k) - y_k|^2\right)^{1/2}. \tag{3.1.4c}$$

Basic idea: Squaring eliminates bias due cancellations



→ Numerous strategies could be employed, but a root-mean square is the most popular/common

→ Minimizing such leads to the name 'least squares'

Kutz (2013)

Least squares

- We call our 'goodness-of-fit' parameter χ^2 ("chi-squared")

Weighted sum of the squares of the deviations

$$\chi^2 = \sum \left[\frac{y_i - y(x_i)}{\sigma_i} \right]^2 = \sum \left[\frac{1}{\sigma_i} (y_i - a - bx_i) \right]^2$$

for a linear fit

- To determine the smallest value of χ^2, the following factors should be kept in mind:

1. Fluctuations in the measured values of the variables y_i, which are random samples from a parent population with expectation values $y_0(x_i)$.
2. The values assigned to the uncertainties σ_i in the measured variables y_i. Incorrect assignment of the uncertainties σ_i will lead to incorrect values of χ^2.
3. The selection of the analytical function $y(x)$ as an approximation to the "true" function $y_0(x)$. It might be necessary to fit several different functions in order to find the appropriate function for a particular set of data.
4. The values of the parameters of the function $y(x)$. Our objective is to find the "best values" of these parameters.

Bevington (2003)

Linear least squares

- To minimize χ^2, we differentiate and find the associated zeros (such will always be a minimum here)

  Note: Mathematically, this is equivalent to finding the equilibria for a set of PDEs

$$\frac{\partial}{\partial a}\chi^2 = \frac{\partial}{\partial a}\sum\left[\frac{1}{\sigma_i^2}(y_i - a - bx)^2\right]$$
$$= -2\sum\left[\frac{1}{\sigma_i^2}(y_i - a - bx_i)\right] = 0$$
$$\frac{\partial}{\partial b}\chi^2 = \frac{\partial}{\partial b}\sum\left[\frac{x_\mathrm{i}}{\sigma_i^2}(y_i - a - bx_i)^2\right]$$
$$= -2\sum\left[\frac{x_\mathrm{i}}{\sigma_i^2}(y_i - a - bx_i)\right] = 0$$

Rearrange as:

$$\sum\frac{y_i}{\sigma_i^2} = a\sum\frac{1}{\sigma_i^2} + b\sum\frac{x_i}{\sigma_i^2}$$
$$\sum\frac{x_i y_i}{\sigma_i^2} = a\sum\frac{x_i}{\sigma_i^2} + b\sum\frac{x_i^2}{\sigma_i^2}$$

→ This is just a linear system of equations (i.e., two equations, two unknowns)! Here we are solving for a and b

Bevington (2003)

Least squares

- Determinant solution:

$$a = \frac{1}{\Delta}\begin{vmatrix} \sum \frac{y_i}{\sigma_i^2} & \sum \frac{x_i}{\sigma_i^2} \\ \sum \frac{x_i y_i}{\sigma_i^2} & \sum \frac{x_i^2}{\sigma_i^2} \end{vmatrix} = \frac{1}{\Delta}\left(\sum \frac{x_i^2}{\sigma_i^2} \sum \frac{y_i}{\sigma_i^2} - \sum \frac{x_i}{\sigma_i^2} \sum \frac{x_i y_i}{\sigma_i^2}\right)$$

$$b = \frac{1}{\Delta}\begin{vmatrix} \sum \frac{1}{\sigma_i^2} & \sum \frac{y_i}{\sigma_i^2} \\ \sum \frac{x_i}{\sigma_i^2} & \sum \frac{x_i y_i}{\sigma_i^2} \end{vmatrix} = \frac{1}{\Delta}\left(\sum \frac{1}{\sigma_i^2} \sum \frac{x_i y_i}{\sigma_i^2} - \sum \frac{x_i}{\sigma_i^2} \sum \frac{y_i}{\sigma_i^2}\right)$$

$$\Delta = \begin{vmatrix} \sum \frac{1}{\sigma_i^2} & \sum \frac{x_i}{\sigma_i^2} \\ \sum \frac{x_i}{\sigma_i^2} & \sum \frac{x_i^2}{\sigma_i^2} \end{vmatrix} = \sum \frac{1}{\sigma_i^2} \sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2}\right)^2$$

→ Direct recipe to solve for a and b

Bevington (2003)

Least squares

- These expression simplify further when all uncertainties are equal ($\sigma=\sigma_i$):

$$a = \frac{1}{\Delta'}\begin{vmatrix} \Sigma y_i & \Sigma x_i \\ \Sigma x_i y_i & \Sigma x_i^2 \end{vmatrix} = \frac{1}{\Delta'}(\Sigma x_i^2 \Sigma y_i - \Sigma x_i \Sigma x_i y_i)$$

$$b = \frac{1}{\Delta'}\begin{vmatrix} N & \Sigma y_i \\ \Sigma x_i & \Sigma x_i y_i \end{vmatrix} = \frac{1}{\Delta'}(N \Sigma x_i y_i - \Sigma x_i \Sigma y_i)$$

$$\Delta' = \begin{vmatrix} N & \Sigma x_i \\ \Sigma x_i & \Sigma x_i^2 \end{vmatrix} = N \Sigma x_i^2 - (\Sigma x_i)^2$$

- Kutz's book pitches it slightly differently:

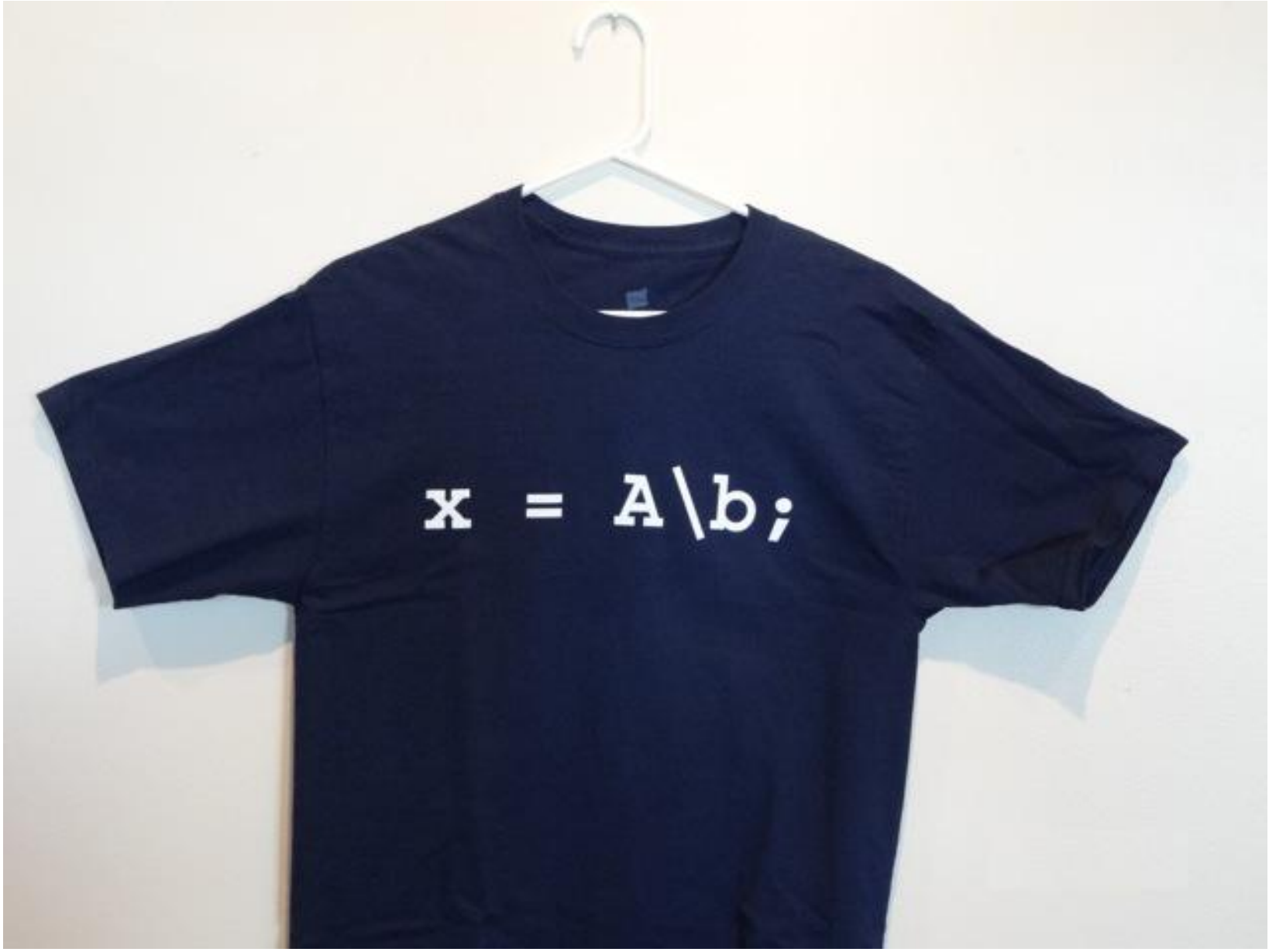
Upon rearranging, the 2×2 system of linear equations is found for A and B:

$$\begin{pmatrix} \sum_{k=1}^{n} x_k^2 & \sum_{k=1}^{n} x_k \\ \sum_{k=1}^{n} x_k & n \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{n} x_k y_k \\ \sum_{k=1}^{n} y_k \end{pmatrix}.$$

This equation can be easily solved using the backslash command in MATLAB.

Bevington (2003)
Kutz (2013)

Do you speak Matlab?



http://blogs.mathworks.com/cleve/2013/08/19/backslash/

Computational methods to minimize χ^2

- Matlab has numerous built-in functions. For a linear fit (and other polynomials), one can use `polyfit.m` (see example code EXregression1.m for syntax)

- For the linear case, the preceding formulae for the exact solution provide an explicit recipe (see example code EXregression1.m for syntax)

- Brute force estimate χ^2 for a range of parameter values and see which ones provide the smallest value. Refine your search and repeat. (this is called the grid-search method)

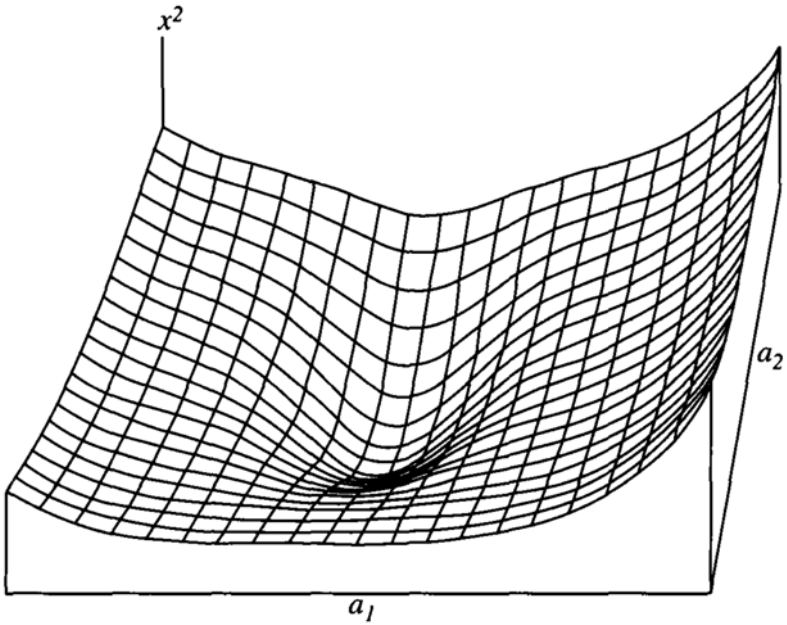


FIGURE 8.2
Chi-square hypersurface as a function of two parameters.

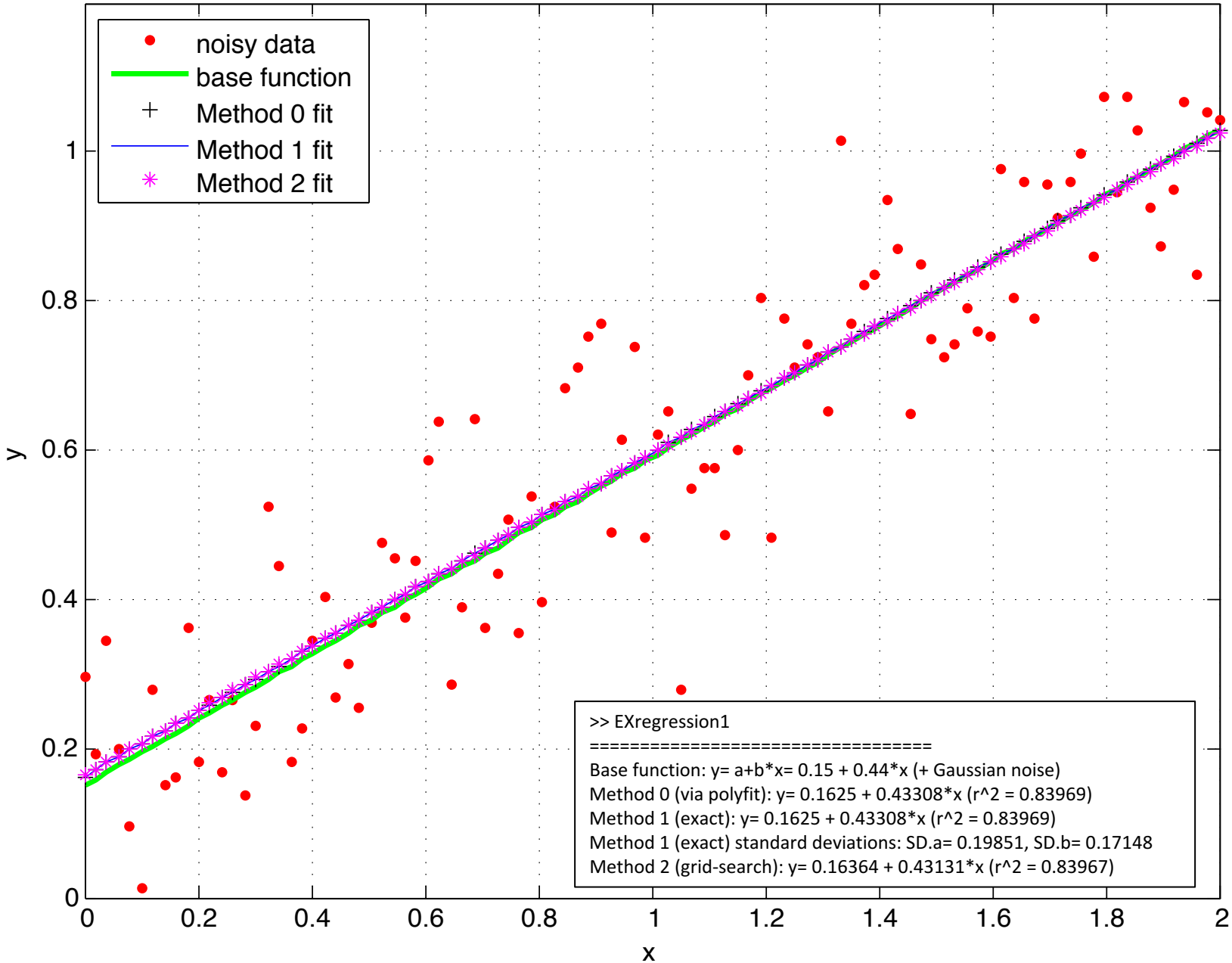
Bevington (2003)

```
% ### EXregressionEX1.m ###     10.03.14
% Fit a straight line: y(x) = a + b*x (i.e., find optimal values for a and
% b for a linear function to a given set of data) using three methods:
% Method 1. Matlab's built-in blackbox function polyfit.m
% Method 2. exact analytic solution (via Bevington)
% Method 3. brute force minimizing chi-squared via a grid-search method (user specifies grid)
% --> assumes sigma_i=1 for all points (i.e., unit variance, const. for all points)

clear; figure(1); clf;
% -------------------------------------
% specify range of x-values
xMin= 0;    % min.
xMax= 2;    % max.
xNum= 100;  % number of 'data' points
%  choose parameters of linear function (y=aD+bD*x+noise)
aD= 0.15;   % intercept
bD= 0.44;   % slope
noiseF= 0.1;   % noise factor {0.1}
% define 'grid' range for Method 2 (i.e., one needs to 'guess' here!)
gridA= linspace(0.1,0.4,100);   % intercept
gridB= linspace(0.1,0.5,100);   % slope

% -------------------------------------
data.x= linspace(xMin,xMax,xNum);
data.y= aD+ bD*data.x + noiseF*randn(numel(data.x),1)'; % determine noisy (straight) line
disp(['==================================']);
disp(['Base function: y= a+b*x= ',num2str(aD),' + ',num2str(bD),'*x (+ Gaussian noise)']);
N= numel(data.x);   % total number of points
% ------------
% Method 0: Built-in Matlab function
[p0,S]= polyfit(data.x,data.y,1);
% also calculate r^2 (coefficient of determination) via
% http://www.mathworks.com/help/matlab/data_analysis/linear-regression.html?refresh=true
yfit = polyval(p0,data.x);    % this line is equivalent to > yfit =  p(1) * x + p(2);
yresid = data.y - yfit;
SSresid = sum(yresid.^2);
SStotal = (length(data.y)-1) * var(data.y);
RS0= 1 - SSresid/SStotal;
disp(['Method 0 (via polyfit): y= ',num2str(p0(2)),' + ',num2str(p0(1)),'*x (r^2 = ',num2str(RS0),')']);
```

Assumes all uncertainties are equal ($\sigma=\sigma_i$)

```
% ------------
% Method 1: Exact linear regression (see ch.6 from Bevington, eqns.6.13 and 6.23)
Delta= N*sum(data.x.^2)-sum(data.x)^2;
a1= 1/Delta*(sum(data.x.^2)*sum(data.y)- sum(data.x)*sum(data.x.*data.y));
b1= 1/Delta*(N*sum(data.x.*data.y)- sum(data.x)*sum(data.y));
sigmaA1= sqrt(1/Delta*sum(data.x.^2));
sigmaB1= sqrt(N/Delta);
% also calculate r^2 (coefficient of determination) via
% http://en.wikipedia.org/wiki/Coefficient_of_determination
meanY= sum(data.y)/N;
bottom1= sum((data.y-meanY).^2);
top1= sum((data.y-a1-b1*data.x).^2);
RS1= 1- top1/bottom1;
disp(['Method 1 (exact): y= ',num2str(a1),' + ',num2str(b1),'*x (r^2 = ',num2str(RS1),')']);
disp(['Method 1 (exact) standard deviations: SD.a= ',num2str(sigmaA1),', SD.b= ',num2str(sigmaB1)]);
% ------------
% Method 2: Brute-force minimize chi-squared via grid search
for n=1:numel(gridA)
    for m=1:numel(gridB)
        aT= gridA(n);   % extract 'test' parameters
        bT= gridB(m);
        chiS(n,m)= sum((data.y-aT-bT*data.x).^2); % determine chi-squared and store away
    end
end
% determines coords. of global minimum in chi-squared array (two lines
% below are a quick/dirty way to find the desired coords.)
[junk,indxT] = min(chiS(:));
[p,q] = ind2sub(size(chiS),indxT);
a2= gridA(p);   b2= gridB(q);   % extract the associated values
top2= sum((data.y-a2-b2*data.x).^2);
RS2= 1- top2/bottom1;
disp(['Method 2 (grid-search): y= ',num2str(a2),' + ',num2str(b2),'*x (r^2 = ',num2str(RS2),')']);
% ------------
% visualize
plot(data.x,data.y,'r.');   % original 'data' points
hold on; grid on; xlabel('x'); ylabel('y'); axis([xMin xMax floor(min(data.y)) ceil(max(data.y))]);
plot(data.x,aD+bD*data.x,'g-','LineWidth',2); % base function (sans noise)
plot(data.x,p0(2)+p0(1)*data.x,'k+'); % Method 0 fit
plot(data.x,a1+b1*data.x,'b-'); % Method 1 fit
plot(data.x,a2+b2*data.x,'m*'); % Method 2 fit
legend('noisy data','base function','Method 0 fit','Method 1 fit','Method 2 fit','Location','NorthWest');
```

EXregression1.m



Least squares

- Bonus: We can use regression to estimate the uncertainty in our determined parameters!

$$\sigma_a^2 \simeq \sum_{j=1}^{N} \frac{\sigma_j^2}{\Delta^2}\left[\frac{1}{\sigma_j^4}\left(\sum \frac{x_i^2}{\sigma_i^2}\right)^2 - \frac{2x_j}{\sigma_j^4}\sum \frac{x_i^2}{\sigma_i^2}\sum \frac{x_i}{\sigma_i^2} + \frac{x_j^2}{\sigma_j^4}\left(\sum \frac{x_i}{\sigma_i^2}\right)^2\right]$$
$$= \frac{1}{\Delta^2}\left[\sum \frac{1}{\sigma_j^2}\left(\sum \frac{x_i^2}{\sigma_i^2}\right)^2 - 2\sum \frac{x_j}{\sigma_j^2}\sum \frac{x_i^2}{\sigma_i^2}\sum \frac{x_i}{\sigma_i^2} + \sum \frac{x_j^2}{\sigma_j^2}\left(\sum \frac{x_i}{\sigma_i^2}\right)^2\right]$$
$$= \frac{1}{\Delta^2}\left(\sum \frac{x_i^2}{\sigma_i^2}\right)\left[\sum \frac{1}{\sigma_j^2}\sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2}\right)^2\right]$$
$$= \frac{1}{\Delta}\sum \frac{x_i^2}{\sigma_i^2}$$

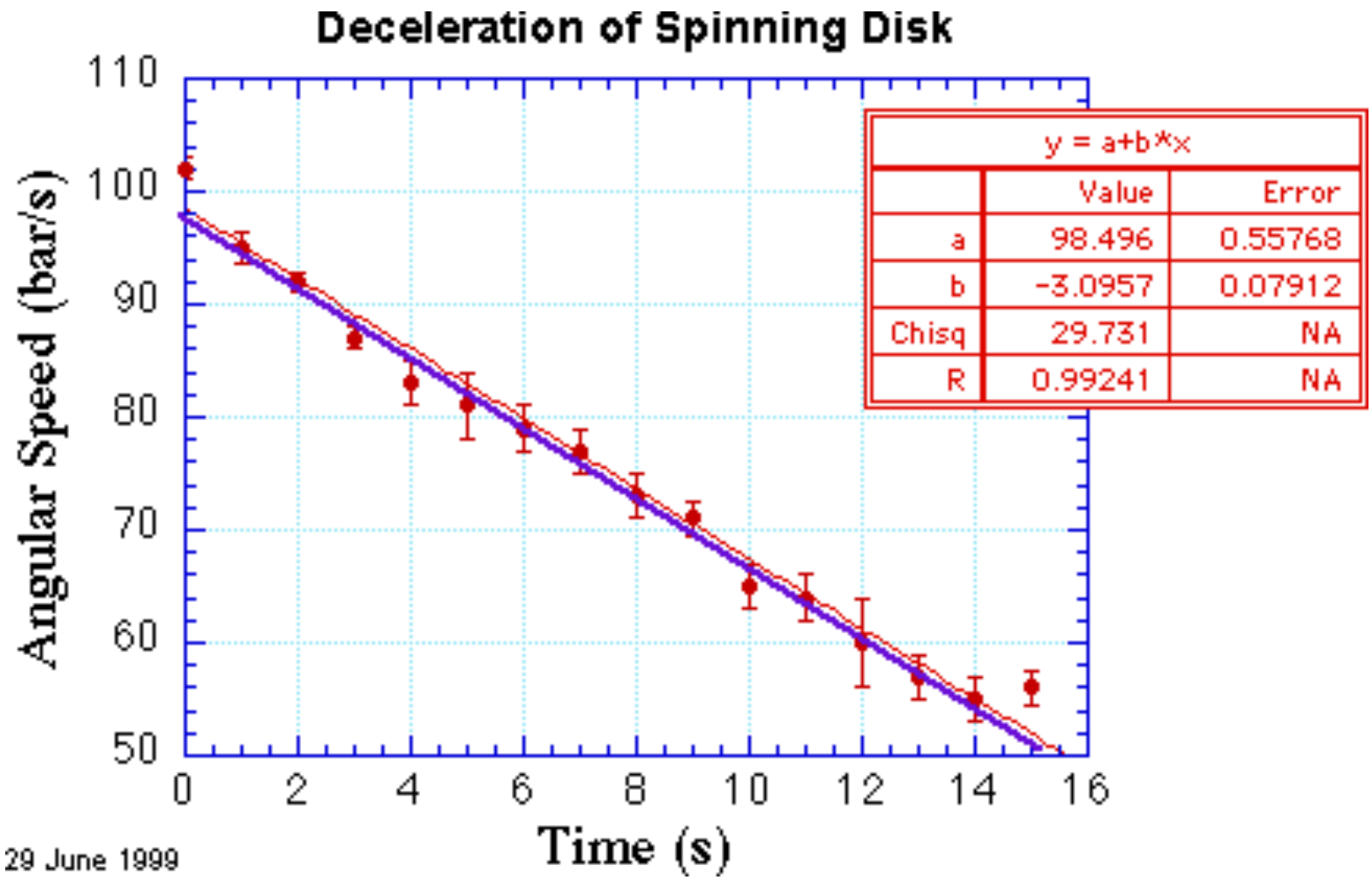
$$\sigma_b^2 \simeq \sum_{j=1}^{N} \frac{\sigma_j^2}{\Delta^2}\left[\frac{x_j^2}{\sigma_j^4}\left(\sum \frac{1}{\sigma_i^2}\right)^2 - \frac{2x_j}{\sigma_j^4}\sum \frac{1}{\sigma_i^2}\sum \frac{x_i}{\sigma_i^2} + \frac{1}{\sigma_j^4}\left(\sum \frac{x_i}{\sigma_i^2}\right)^2\right]$$
$$= \frac{1}{\Delta^2}\left[\sum \frac{x_j^2}{\sigma_j^2}\left(\sum \frac{1}{\sigma_i^2}\right)^2 - 2\sum \frac{x_j}{\sigma_j^2}\sum \frac{1}{\sigma_i^2}\sum \frac{x_i}{\sigma_i^2} + \sum \frac{1}{\sigma_j^2}\left(\sum \frac{x_i}{\sigma_i^2}\right)^2\right]$$
$$= \frac{1}{\Delta^2}\left(\sum \frac{x_j^2}{\sigma_i^2}\right)\left[\sum \frac{1}{\sigma_j^2}\sum \frac{1}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2}\right)^2\right]$$
$$= \frac{1}{\Delta}\sum \frac{1}{\sigma_i^2}$$

- These expressions simplify further when all uncertainties are equal ($\sigma=\sigma_i$):

$$\sigma_a^2 = \frac{\sigma^2}{\Delta'}\Sigma x_i^2 \qquad \text{and} \qquad \sigma_b^2 = N\frac{\sigma^2}{\Delta'}$$

- Can also calculate a 'goodness-of-fit' parameter r^2 ('coefficient of determination')

Bevington (2003)



http://www.physics.hmc.edu/analysis/fitting.php

Class exercises

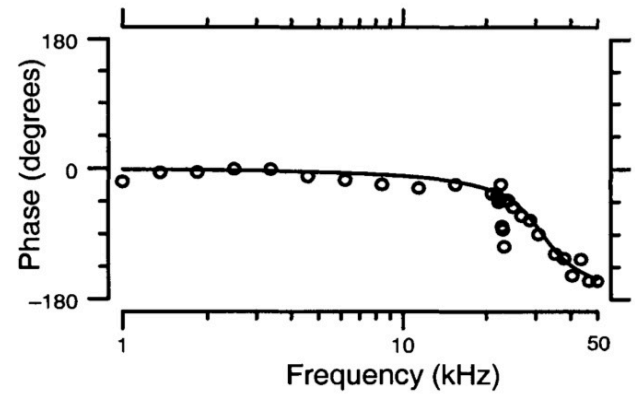
- Use deplot.m to extract the data points from the steady-state phase of the micro-mechanical oscillator or Anscombe's quartet

- Can you reproduce the same linear fits for each member of the quartet?

- Modify EXregression1.m to allow for random uncertainty associated with each data point

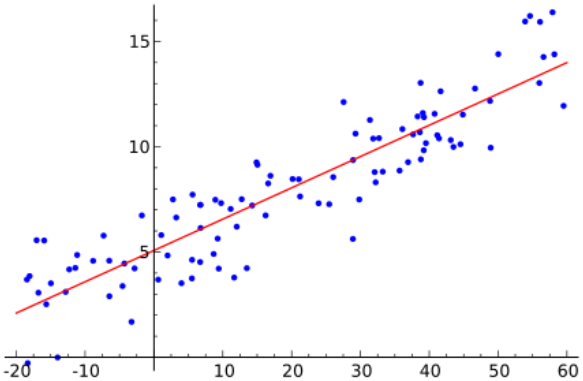
- Think about how the ideas here for the linear case could be extended to nonlinear functions

$$\delta(\omega) = \arctan\left(\frac{\gamma\omega}{\omega^2 - \omega_o^2}\right)$$



- What is we didn't have a specific function in mind? Can we still fit a curve in some useful way using regression?

Summary (re linear least squares)



'Fit' a curve (straight line) to data
→ optimization problem!

Starting point:

- data
- 'model'

$$\chi^2$$

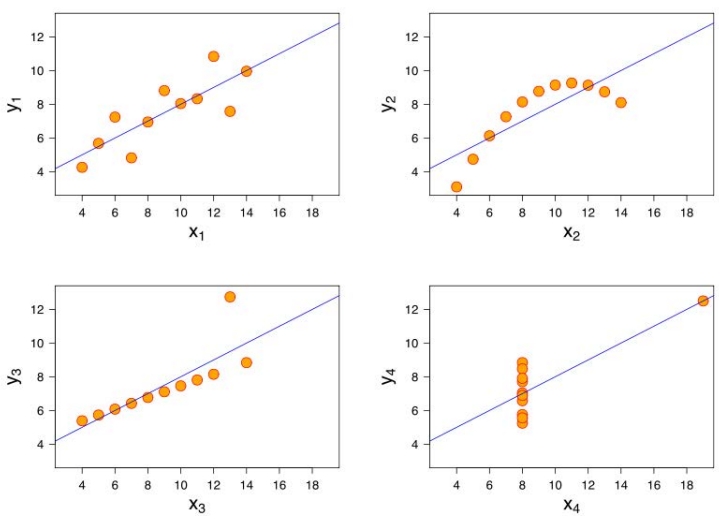
'Goodness of fit' parameter

$$a = \frac{1}{\Delta}\begin{vmatrix} \sum \frac{y_i}{\sigma_i^2} & \sum \frac{x_i}{\sigma_i^2} \\ \sum \frac{x_i y_i}{\sigma_i^2} & \sum \frac{x_i^2}{\sigma_i^2} \end{vmatrix} = \frac{1}{\Delta}\left(\sum \frac{x_i^2}{\sigma_i^2} \sum \frac{y_i}{\sigma_i^2} - \sum \frac{x_i}{\sigma_i^2} \sum \frac{x_i y_i}{\sigma_i^2}\right)$$

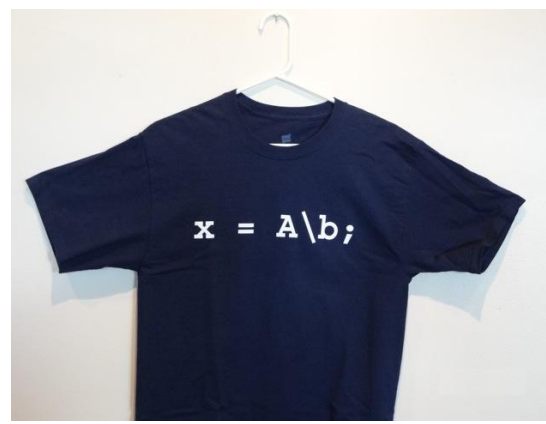
$$b = \frac{1}{\Delta}\begin{vmatrix} \sum \frac{1}{\sigma_i^2} & \sum \frac{y_i}{\sigma_i^2} \\ \sum \frac{x_i}{\sigma_i^2} & \sum \frac{x_i y_i}{\sigma_i^2} \end{vmatrix} = \frac{1}{\Delta}\left(\sum \frac{1}{\sigma_i^2} \sum \frac{x_i y_i}{\sigma_i^2} - \sum \frac{x_i}{\sigma_i^2} \sum \frac{y_i}{\sigma_i^2}\right)$$

$$\Delta = \begin{vmatrix} \sum \frac{1}{\sigma_i^2} & \sum \frac{x_i}{\sigma_i^2} \\ \sum \frac{x_i}{\sigma_i^2} & \sum \frac{x_i^2}{\sigma_i^2} \end{vmatrix} = \sum \frac{1}{\sigma_i^2} \sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2}\right)^2$$

Method: Exact solution



Be smart about the assumptions you make!



Method: Matrix solution

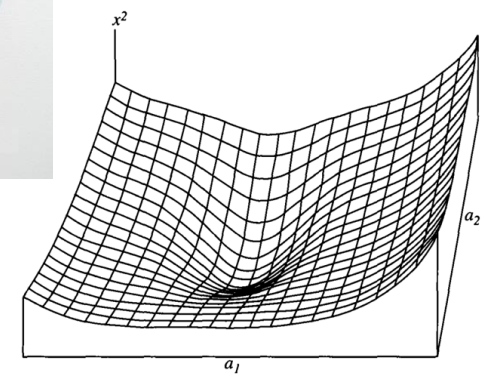


FIGURE 8.2
Chi-square hypersurface as a function of two parameters.

Method: Brute force minimizing χ^2