The Variance-Covariance Matrix

Our biggest feat so-far has been fitting a linear function to a set of data by minimizing the least squares differences from the fit to the data with fminsearch. When analyzing non-linear data, you have to use a program like Matlab as many types of data cannot be linearized such that Excel can analyze it. This also means that you do not have a direct route to calculating the error of your fit, as you have been using the error calculated from the linear least squares result (this doesn't work on non-linear data!).

Now you can make one of two assumptions- 1) error cannot be calculated from a non-linear fit or 2) error can be calculated from a non-linear fit, I just don't know how.
Unless you're a fre@¿*ħ↗≈*in' moron, you should pick 2.

Here is how! The variance-covariance matrix.

I will not cover the derivation, not that I don't understand it (I so *totally* do) but it is several pages of algebra long. I will show you the formula for the error analysis and prove it works by applying it to a linear set of data. Then we will use it for other problems…

Step 1: define a matrix of partial derivatives.

This is the only semi-hard part, you have to calculate the partial derivative of the function you are fitting your data to for each variable that you are minimizing in your least squares fit (remember fminsearch from the previous lesson?). Let's say you are fitting data to a function f(m,b) which has two variables that were fit my minimizing the error with fminsearch, call them m and b. The first step to calculate σ_m and σ_b is to derive the partial derivative matrix:
M=

$\frac{\partial f(m,b)}{\partial m} \bullet \frac{\partial f(m,b)}{\partial m}$	$\frac{\partial f(m,b)}{\partial m} \bullet \frac{\partial f(m,b)}{\partial b}$
$\frac{\partial f(m,b)}{\partial b} \bullet \frac{\partial f(m,b)}{\partial m}$	$\frac{\partial f(m,b)}{\partial b} \bullet \frac{\partial f(m,b)}{\partial b}$

Where $\frac{\partial f(m,b)}{\partial m}$ is the partial derivative with respect to the variable your fitting (m) and likewise for $\frac{\partial f(m,b)}{\partial b}$ with variable b. Note that the diagonal (1,1) and (2,2) elements are **NOT** equal to the second derivative $\frac{\partial f(m,b)^2}{\partial^2 m}$ but are in fact just the first derivative squared, i.e. $\left(\frac{\partial f(m,b)}{\partial m}\right)^2$. The off-diagonal elements are the partial derivatives multiplied by each other. If this is confusing now, we will make it a lot more clear with an example on the next page. For now, let me simplify the matrix **M** as:

$\left(\frac{\partial f(m,b)}{\partial m}\right)^2$	$\frac{\partial f(m,b)}{\partial m} \bullet \frac{\partial f(m,b)}{\partial b}$
$\frac{\partial f(m,b)}{\partial b} \bullet \frac{\partial f(m,b)}{\partial m}$	$\left(\frac{\partial f(m,b)}{\partial b}\right)^2$

Now let me be totally honest here: the matrix **M** is actually a sum over all data N points. You also must include the error of the data points (σ_i); overall this is properly expressed as:

$$\begin{vmatrix} \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial m} \right)^2 & \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial m} \cdot \frac{\partial f(m,b)}{\partial b} \right) \\ \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial b} \cdot \frac{\partial f(m,b)}{\partial m} \right) & \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial b} \right)^2 \end{vmatrix}$$

Now before this gets confusing, let's cement everything with a simple example. Let's fit this data here, which was included as the example data in the Handout section (dataset3.txt):

x_i	y_i	σ_i
0	1.9	1
1	4.2	1
2	5.9	1
3	7.8	1
4	11	1
5	12.2	1
6	14.1	1

If I do a linear least squares analysis of this data using llsq.m, I get the following output:
The slope is 2.06071 ± 0.0790085
The intercept is 1.975 ± 0.284869
R^2 is 0.992704

If I use fminsearch, I get the same slope and intercept. But fminsearch doesn't return errors! To go about this, we will use the variance covariance matrix method. Also, don't lose sight of the fact that we are doing this because not all data are linear, and we can't use the linear least squares method to calculate errors.
The first step is to define the partial derivatives of the function f(x(i)) = y(i) = m × x(i) +b:

$\frac{\partial f(m,b)}{\partial m}$ = x(i). Done. Next, $\frac{\partial f(m,b)}{\partial b}$ = 1. Done! Monkey-Donkey-Super Done! Let's put this into the matrix:

$$\begin{vmatrix} \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial m} \right)^2 & \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial m} \cdot \frac{\partial f(m,b)}{\partial b} \right) \\ \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial b} \cdot \frac{\partial f(m,b)}{\partial m} \right) & \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(m,b)}{\partial b} \right)^2 \end{vmatrix}$$

which is now:

$$\begin{vmatrix} \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot (x(i) \cdot x(i)) & \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot (x(i) \cdot 1) \\ \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot (1 \cdot x(i)) & \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot (1 \cdot 1) \end{vmatrix}$$

Simplifying gives:

$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot x(i)^2$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot x(i)$
$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot x(i)$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2}$

Let's calculate each element with Matlab to show you how simple these formulas are:
Load in the dataset. The first column is the x data, second is the y data, and the third are the errors σ. To define the 2-by-2 matrix above (called m):

```
>> m(1,1)=0; for i=1:7 m(1,1)=m(1,1)+(1/dataset3(i,3)^2)*dataset3(i,1)^2; end;
>> m(1,2)=0; for i=1:7 m(1,2)=m(1,2)+(1/dataset3(i,3)^2)*dataset3(i,1); end;
>> m(2,1)=0; for i=1:7 m(2,1)=m(2,1)+(1/dataset3(i,3)^2)*dataset3(i,1); end;
>> m(2,2)=0; for i=1:7 m(2,2)=m(2,2)+1; end;
>> m
m =
   91   21
   21    7
```

YOU'RE NOW DONE WITH THE HARD PART!!

Step 2- This matrix must be inverted.

Now what does it mean to invert a matrix? For a matrix **M**, the inverse of the matrix (called $\mathbf{M}^{-1}$) has the property such that $\mathbf{M} \cdot \mathbf{M}^{-1} = \mathbf{1}$. Not quite the number 1, but a matrix with the same number of elements as M, each diagonal element being the number 1 (others are 0). So in this example:

a	b
c	d

The inverse is (see http://mathworld.wolfram.com/MatrixInverse.html for more info):

$\left(\frac{1}{ad-bc}\right) \cdot d$	$-\left(\frac{1}{ad-bc}\right) \cdot b$
$-\left(\frac{1}{ad-bc}\right) \cdot c$	$\left(\frac{1}{ad-bc}\right) \cdot a$

Multiply the matrix with its inverse using the rules of matrix algebra and you get:

1	0
0	1

Here is the great thing- **don't worry about any of this**- Matlab does everything for you!

```
>> m_inv=inv(m)
m_inv =
   0.0357   -0.1071
  -0.1071    0.4643
```

Lets double check it:

```
>> m_inv*m
ans =
    1    0
    0    1
```

See, once you set up the **M** matrix and its inverse, you don't have to do anything else!
YAY! YAY MATLAB!!

Step 3- Lets first talk about the data we are fitting: we will define N as the number of data points to fit, and lets call the actual data we wish to fit y(i). In our case, there are 2 variables to the function we are fitting the data to (m and b), so let's define p as this quantity (p=2). For step three, we have to know something about the deviation of the data from the fit; that obviously must play a part in the errors of m and b. To do this part, define s_y^2 as:

$$s_y^2 = \frac{\sum_{i=1}^{N} \left(y(i) - fit(i)\right)^2}{N - p} .$$

Almost done! Define the best fit from fminsearch for the line (which are the same parameters off my Excel spreadsheet oddly enough):

```
>> for i=1:7 fit(i)=2.060714*dataset3(i,1)+1.975; end;
```

Next step:

```
>> sy2=0;
>> p=7-2;
>> for i=1:7 sy2=sy2+((dataset3(i,2)-fit(i))^2)/p; end;
>>sy2
sy2 =
   0.1748

>> varcovar=m_inv*sy2
varcovar =
   0.0062   -0.0187
  -0.0187    0.0812
```

This is it! The Variance-Covariance Matrix!!

Actually, this is kinda like learning that the ultimate answer to the ultimate question in the universe is the number 42. You have to fully understand the question to truly get the answer.

The variance-covariance is actually equal to:

σ_m^2	$\sigma_m \times \sigma_b$
$\sigma_b \times \sigma_m$	σ_b^2

So if you want to know the error of the slope you type:

```
>> sqrt(varcovar(1,1))
ans =
```

0.0790

Likewise for the intercept:

```
>>sqrt(varcovar(2,2))
ans =
```

0.2849

Rememer the result from the linear least squares analysis above? The linear least squares result was:

slope = 2.0607 ± **0.0790**

intercept: 1.975 ± **0.2849**.

Looks like the variance-covariance matrix works!

Last bit, let's look back at the variance-covariance matrix:

```
>> varcovar
varcovar =
   0.0062   -0.0187
  -0.0187    0.0812
```

How did we know that the square root of the (1,1) element (upper left handed) is the error in the slope? Easy, that was defined when you took the derivative of the equation for a line with respect to the slope as: $\frac{\partial f(m,b)}{\partial m}$, likewise for the intercept.

Now what do the off-diagonal elements $\sigma_m \times \sigma_b$ and $\sigma_b \times \sigma_m$ mean? First, the off-diagonal elements are the "covariances." The coviariance elements tell you that if your calculated slope is too low, then your calculated intercept it too high (as in this case the covariances are negative.). If they are positive, then an underestimate in the slope means that your intercept is also underestimated (i.e. they are "going-together"). A large off-diagonal element means that there is a lot of "cross-talk" between the m and b variables. Ideally, the covariances are 0, meaning that if you made a bad fit to the intercept, it did nothing to your estimate of the slope. Thus, the variables are independent. This is unfortunately rarely the case with real data.

Here is the most important part- starting from the beginning, who the heck sez you have to work with linear fits?

Effect of errors.

Now we can examine exactly how the errors of each point affect the results. First, change the σ values of the whole set from 1 to 0.5:

```
>>dataset3(:,3)=.5;
```

Next, repeat the analysis above, and you will find:

```
>> sqrt(varcovar(1,1))
ans =
   0.0395
```

Likewise for the intercept:

```
>>sqrt(varcovar(2,2))
ans =
   0.1424
```

Note how intuitive this is? If the associated error with each datapoint is half, then the uncertainties in all measured values are likewise half of what they were before! Next, try to double the original errors to 2, and you will see that the associated errors likewise double.

YOU NOW CAN CALULCATE THE ERROR OF ANY FUNCTION YOU CHOOSE AS THE BEST TO FIT YOUR DATA

Another example, an exponential decay.

An exponential decay is described by the equation $f(t) = A \cdot e^{-k \cdot t}$, where A is the amplitude, t is time, and k is the decay constant. Let's figure out how to make a variance-covariance matrix from this equation. Noting that we are only fitting two variables, and starting from the beginning:

M =

$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(A,k)}{\partial A} \right)^2$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(A,k)}{\partial A} \cdot \frac{\partial f(A,k)}{\partial k} \right)$
$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(A,k)}{\partial k} \cdot \frac{\partial f(A,k)}{\partial A} \right)$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(\frac{\partial f(A,k)}{\partial k} \right)^2$

Now let's fill in the individual pieces:

$\frac{\partial f(A,k)}{\partial A} = e^{-k \cdot t}$ and $\frac{\partial f(A,k)}{\partial k} = -A \cdot e^{-k \cdot t} \cdot t$. Now let's plug this into the matrix, and we get

M=

$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(e^{-k \cdot t}\right)^2$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(e^{-k \cdot t} \cdot -A \cdot e^{-k \cdot t} \cdot t\right)$
$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(-A \cdot e^{-k \cdot t} \cdot t \cdot e^{-k \cdot t}\right)$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot \left(-A \cdot e^{-k \cdot t} \cdot t\right)^2$

This can be simplified as:

$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot e^{-2 \cdot k \cdot t}$	$\sum_{i=1}^{N} \left(\frac{1}{\sigma_i^2}\right) \cdot -A \cdot t \cdot e^{-2 \cdot k \cdot t}$
$\sum_{i=1}^{N} \left(\frac{1}{\sigma_i^2}\right) \cdot -A \cdot t \cdot e^{-2 \cdot k \cdot t}$	$\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \cdot A^2 \cdot t^2 \cdot e^{-2 \cdot k \cdot t}$

Now you will finish the rest for your next assignment.