CS395T
Computational Statistics with Application to Bioinformatics

Prof. William H. Press
Spring Term, 2011
The University of Texas at Austin

Lecture 11

Weighted Nonlinear Least Squares Fitting
a.k.a. χ^2 Fitting
a.k.a. Maximum Likelihood Estimation of Parameters (MLE)
a.k.a. Bayesian parameter estimation
(with uniform prior and maybe
some other normality assumptions)

returned by Google for image search on "least squares fitting"!

these are not all exactly identical, but they're very close!

$$y_i = y(\mathbf{x}_i | \mathbf{b}) + e_i$$

measured values supposed to be a model, plus an error term

$$e_i \sim N(0, \sigma_i)$$

the errors are Normal, either independently…

$$\mathbf{e} \sim N(0, \mathbf{\Sigma})$$

… or else with errors correlated in some known way (e.g., multivariate Normal)

We want to find the parameters of the model **b** from the data.

Fitting is usually presented in frequentist, MLE language.
But one can equally well think of it as Bayesian:

$$\begin{aligned} P(\mathbf{b}|\{y_i\}) &\propto P(\{y_i\}|\mathbf{b})P(\mathbf{b}) \\ &\propto \prod_i \exp\left[-\tfrac{1}{2}\left(\frac{y_i - y(\mathbf{x}_i|\mathbf{b})}{\sigma_i}\right)^2\right] P(\mathbf{b}) \\ &\propto \exp\left[-\tfrac{1}{2}\sum_i \left(\frac{y_i - y(\mathbf{x}_i|\mathbf{b})}{\sigma_i}\right)^2\right] P(\mathbf{b}) \\ &\propto \exp[-\tfrac{1}{2}\chi^2(\mathbf{b})]P(\mathbf{b}) \end{aligned}$$

Now the idea is: Find (somehow!) the parameter value $\mathbf{b}_0$ that minimizes χ^2 .

For linear models, you can solve linear "normal equations" or, better, use Singular Value Decomposition. See NR3 section 15.4

In the general nonlinear case, you have a general minimization problem, for which there are various algorithms, none perfect.

Those parameters are the MLE. (So it is Bayes with uniform prior.)

The desired MLE of the parameters is thus a χ^2 minimization problem. (Not just an ad hoc choice! We maximize the posterior probability.)

$$y(x|\mathbf{b}) = b_1 \exp(-b_2 x) + b_3 \exp\left(-\frac{1}{2}\frac{(x-b_4)^2}{b_5^2}\right)$$

$$\chi^2 = \sum_i \left(\frac{y_i - y(x_i|\mathbf{b})}{\sigma_i}\right)^2$$

Nonlinear fits are often easy in MATLAB (or other high-level languages) **if** you can make a reasonable starting guess for the parameters.

```
ymodel = @(x,b) b(1)*exp(-b(2)*x)+b(3)*exp(-(1/2)*((x-b(4))/b(5)).^2)

chisqfun = @(b) sum(((ymodel(x,b)-y)./sig).^2)
```

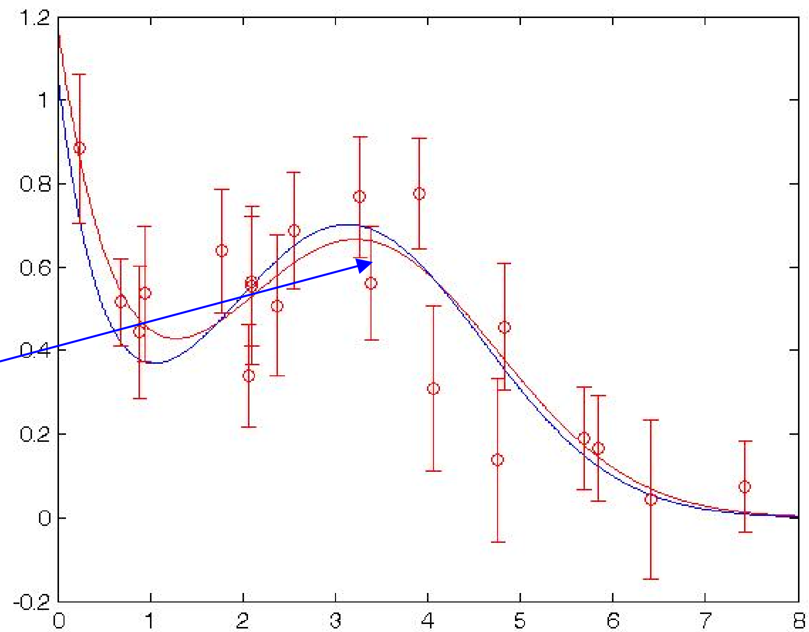
```
bguess = [1 2 .5 3 1.5]

bfit = fminsearch(chisqfun,bguess)

xfit = (0:0.01:8);

yfit = ymodel(xfit,bfit);
```

```
bfit = 1.1235     1.5210     0.6582
     3.2654     1.4832
```



Later, we'll suppose that what we really care about is the area of the bump, and that the other parameters are "nuisance parameters".

How accurately are the fitted parameters determined?
As Bayesians, we would **instead** say, what is their posterior distribution?

Taylor series of any function of a vector:

$$-\tfrac{1}{2}\chi^2(\mathbf{b}) \approx -\tfrac{1}{2}\chi^2_{\min} - \tfrac{1}{2}(\mathbf{b}-\mathbf{b}_0)^T \left[\tfrac{1}{2}\frac{\partial^2\chi^2}{\partial\mathbf{b}\partial\mathbf{b}}\right](\mathbf{b}-\mathbf{b}_0)$$

While exploring the χ^2 surface to find its minimum, we can also calculate the Hessian (2nd derivative) matrix at the minimum.

Then

$$P(\mathbf{b}|\{y_i\}) \propto \exp\left[-\tfrac{1}{2}(\mathbf{b}-\mathbf{b}_0)^T \boldsymbol{\Sigma}_b^{-1}(\mathbf{b}-\mathbf{b}_0)\right] P(\mathbf{b})$$

with

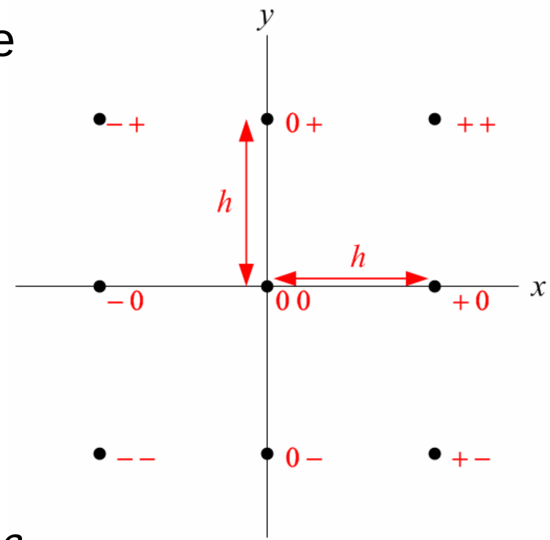
$$\boldsymbol{\Sigma}_b = \left[\tfrac{1}{2}\frac{\partial^2\chi^2}{\partial\mathbf{b}\partial\mathbf{b}}\right]^{-1}$$

covariance (or "standard error") matrix of the fitted parameters

Notice that if (i) the Taylor series converges rapidly and (ii) the prior is uniform, then the posterior distribution of the $\mathbf{b}$'s is multivariate Normal, a very useful CLT-ish result!

Numerical calculation of the Hessian by finite difference

$$\frac{\partial^2 f}{\partial x \partial y} \approx \frac{1}{2h}\left(\frac{f_{++} - f_{-+}}{2h} - \frac{f_{+-} - f_{--}}{2h}\right)$$
$$= \frac{1}{4h^2}\left(f_{++} + f_{--} - f_{+-} - f_{-+}\right)$$



bfit = 1.1235   1.5210   0.6582   3.2654   1.4832

```
chisqfun = @(b) sum(((ymodel(x,b)-y)./sig).^2)
h = 0.1;
unit = @(i) (1:5) == i;
hess = zeros(5,5);
for i=1:5, for j=1:5,
        bpp = bfit + h*(unit(i)+unit(j));
        bmm = bfit + h*(-unit(i)-unit(j));
        bpm = bfit + h*(unit(i)-unit(j));
        bmp = bfit + h*(-unit(i)+unit(j));
        hess(i,j) = (chisqfun(bpp)+chisqfun(bmm)…
          -chisqfun(bpm)-chisqfun(bmp))./(2*h)^2;
    end
end
covar = inv(0.5*hess)
```

This also works for the diagonal components. Can you see how?

For our example, $$y(x|\mathbf{b}) = b_1 \exp(-b_2 x) + b_3 \exp\left(-\frac{1}{2}\frac{(x-b_4)^2}{b_5^2}\right)$$

```
bfit =
    1.1235    1.5210    0.6582    3.2654    1.4832
hess =
   64.3290  -38.3070   47.9973  -29.0683   46.0495
  -38.3070   31.8759  -67.3453   29.7140  -40.5978
   47.9973  -67.3453  723.8271  -47.5666  154.9772
  -29.0683   29.7140  -47.5666   68.6956  -18.0945
   46.0495  -40.5978  154.9772  -18.0945   89.2739
covar =
    0.1349    0.2224    0.0068   -0.0309    0.0135
    0.2224    0.6918    0.0052   -0.1598    0.1585
    0.0068    0.0052    0.0049    0.0016   -0.0094
   -0.0309   -0.1598    0.0016    0.0746   -0.0444
    0.0135    0.1585   -0.0094   -0.0444    0.0948
```

This is the covariance structure of all the parameters, and indeed (at least in CLT normal approximation) gives their entire joint distribution!

The standard errors on each parameter separately are $\sigma_i = \sqrt{C_{ii}}$

```
sigs =
    0.3672    0.8317    0.0700    0.2731    0.3079
```

But why is this, and what about two or more parameters at a time (e.g. b_3 and b_5)?

We can Marginalize or Condition uninteresting parameters. (Different things!)

$$P(\mathbf{b}|\{y_i\}) \propto \exp\left[-\tfrac{1}{2}(\mathbf{b}-\mathbf{b}_0)^T \boldsymbol{\Sigma}_b^{-1}(\mathbf{b}-\mathbf{b}_0)\right] P(\mathbf{b})$$

Marginalize: (this is usual) Ignore (integrate over) uninteresting parameters.

In $\boldsymbol{\Sigma}_b = \left[\frac{1}{2}\frac{\partial^2 \chi^2}{\partial \mathbf{b} \partial \mathbf{b}}\right]^{-1}$ submatrix of *interesting* rows and columns is new $\boldsymbol{\Sigma}_b$

Special case of one variable at a time: Just take diagonal components in $\boldsymbol{\Sigma}_b$

Covariances are pairwise expectations and don't depend on whether other parameters are "interesting" or not.

Condition: (this is rare!) Fix uninteresting parameters at specified values.

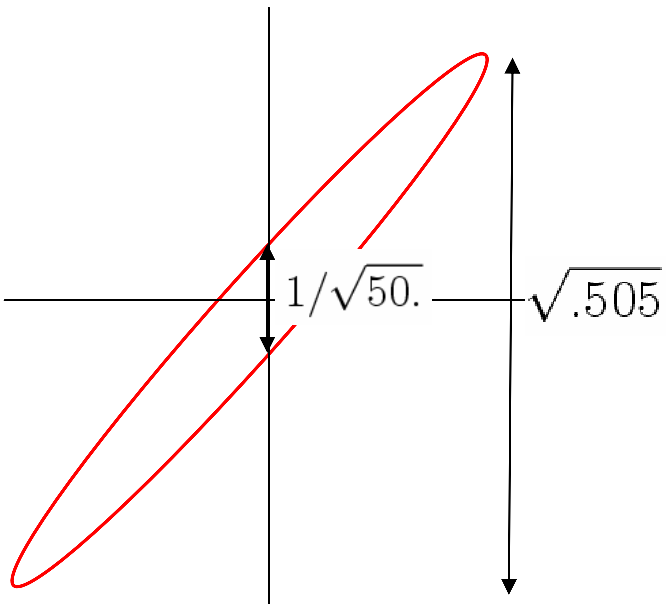
In $\boldsymbol{\Sigma}_b^{-1} = \left[\frac{1}{2}\frac{\partial^2 \chi^2}{\partial \mathbf{b} \partial \mathbf{b}}\right]$ submatrix of *interesting* rows and columns is new $\boldsymbol{\Sigma}_b^{-1}$

Take matrix inverse if you want their covariance $\boldsymbol{\Sigma}_b$

(If you fix uninteresting parameters at any value other than $\mathbf{b}_0$, the mean also shifts – exercise for reader to calculate!)

Example of 2 dimensions marginalizing or conditioning to 1 dimension:

$$P(\mathbf{b}|\{y_i\}) \propto \exp\left[-\tfrac{1}{2}(\mathbf{b}-\mathbf{b}_0)^T \boldsymbol{\Sigma}_b^{-1}(\mathbf{b}-\mathbf{b}_0)\right] P(\mathbf{b})$$



$$\boldsymbol{\Sigma}_b^{-1} = \left[\frac{1}{2}\frac{\partial^2\chi^2}{\partial\mathbf{b}\partial\mathbf{b}}\right] = \begin{pmatrix} 50. & -49. \\ -49. & 50. \end{pmatrix}$$

$$\boldsymbol{\Sigma}_b = \begin{pmatrix} .505 & .495 \\ .495 & .505 \end{pmatrix}$$

By the way, don't confuse the "covariance matrix of the fitted parameters" with the "covariance matrix of the data". For example, the data covariance is often diagonal (uncorrelated σ_i's), while the parameters covariance is essentially <u>never</u> diagonal!

If the data has correlated errors, then the starting point for $\chi^2(\mathbf{b})$ is (recall):

$$\chi^2 = [\mathbf{y}_{\{\mathbf{i}\}} - \mathbf{y}(\mathbf{x}_{\{i\}}|\mathbf{b})]^T \boldsymbol{\Sigma}^{-1} [\mathbf{y}_{\{\mathbf{i}\}} - \mathbf{y}(\mathbf{x}_{\{i\}}|\mathbf{b})] \quad \text{instead of} \quad \sum_i \left(\frac{y_i - y(\mathbf{x}_i|\mathbf{b})}{\sigma_i}\right)^2$$

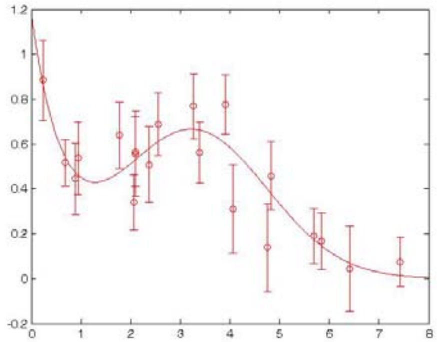
For our example, we are conditioning or marginalizing from 5 to 2 dims:

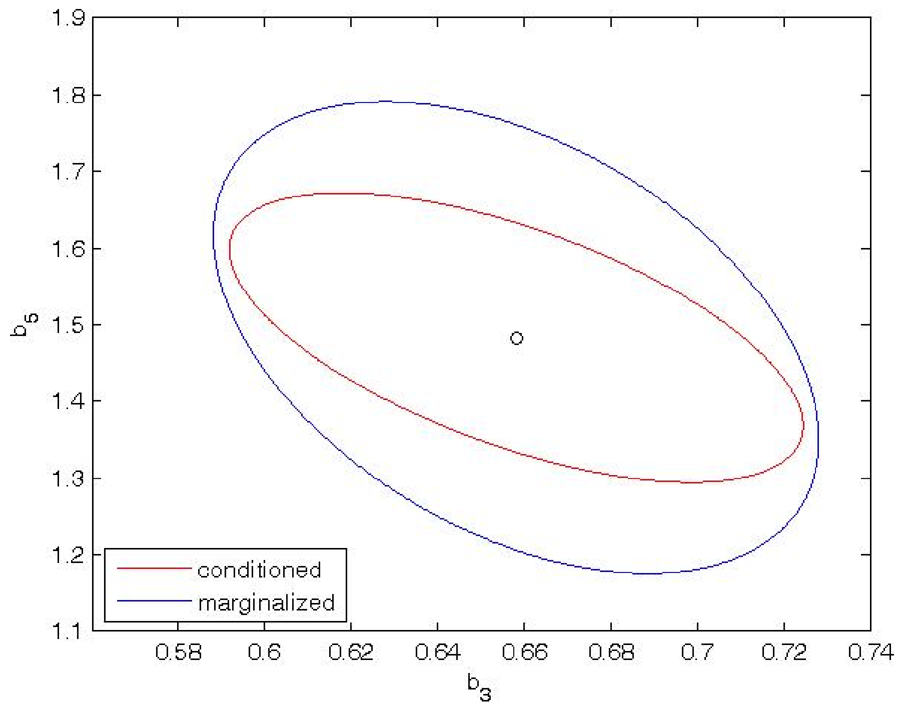
$$y(x|\mathbf{b}) = b_1 \exp(-b_2 x) + b_3 \exp\left(-\frac{1}{2}\frac{(x-b_4)^2}{b_5^2}\right)$$

the uncertainties on b_3 and b_5 jointly (as error ellipses) are

```
sigcond =
    0.0044   -0.0076
   -0.0076    0.0357

sigmarg =
    0.0049   -0.0094
   -0.0094    0.0948
```





Conditioned errors are always smaller, but are useful only if you can find <u>other ways</u> to measure (accurately) the parameters that you want to condition on.

Frequentists love MLE estimates (and not just the case with a Normal error model) because they have provably nice properties asymptotically as the size of the data set becomes large

- Consistency: converges to true value of the parameters
- Equivariance: estimate of function of parameter = function of estimate of parameter
- asymptotically Normal
- asymptotically efficient (optimal): among estimators with the above properties, it has the smallest variance

The "Fisher Information Matrix" is another name for the Hessian of the log probability (or, rather, log likelihood):

$$\mathbf{I}(\mathbf{b}) = -\left\langle \frac{\partial^2 \log P(\{y_i\}|\mathbf{b})}{\partial \mathbf{b} \partial \mathbf{b}} \right\rangle \approx 2\boldsymbol{\Sigma}_b^{-1}$$

except that, strictly speaking, it is an expectation over the population

Bayesians tolerate MLE estimates because they are almost Bayesian – even better if you put the prior back into the minimization.

But Bayesians know that we live in a non-asymptotic world: none of the above properties are exactly true for finite data sets!

What is the uncertainty in quantities other than the fitted coefficients:

Method 1: Linearized propagation of errors

nerdy math note: ∇f is technically a row (not column) vector, because it's a one-form

$\mathbf{b}_0$ is the MLE parameters estimate

$\mathbf{b}_1 \equiv \mathbf{b} - \mathbf{b}_0$ is the RV as the parameters fluctuate

$$f \equiv f(\mathbf{b}) = f(\mathbf{b}_0) + \nabla f \; \mathbf{b}_1 + \cdots$$

$$\langle f \rangle \approx \langle f(\mathbf{b}_0) \rangle + \nabla f \; \cancel{\langle \mathbf{b}_1 \rangle} = f(\mathbf{b}_0)$$

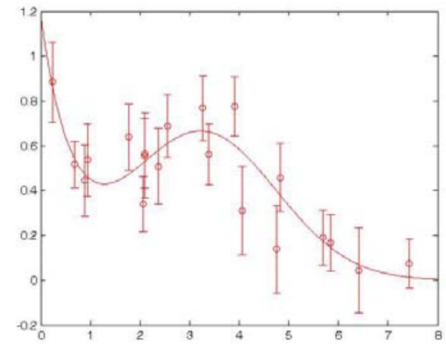
$$\left\langle f^2 \right\rangle - \langle f \rangle^2 \approx 2 f(\mathbf{b}_0)(\nabla f \; \cancel{\langle \mathbf{b}_1 \rangle}) + \left\langle (\nabla f \; \mathbf{b}_1)^2 \right\rangle$$

$$= \nabla f \; \left\langle \mathbf{b}_1 \mathbf{b}_1^T \right\rangle \nabla f^{\,T}$$

$$= \nabla f \; \mathbf{\Sigma}_b \, \nabla f^{\,T}$$

In our example, if we are interested in the area of the “hump”,

```
bfit =
    1.1235    1.5210    0.6582    3.2654    1.4832
covar =
    0.1349    0.2224    0.0068   -0.0309    0.0135
    0.2224    0.6918    0.0052   -0.1598    0.1585
    0.0068    0.0052    0.0049    0.0016   -0.0094
   -0.0309   -0.1598    0.0016    0.0746   -0.0444
    0.0135    0.1585   -0.0094   -0.0444    0.0948
```



$$f = b_3 b_5$$

$$\nabla f = (0, 0, b_5, 0, b_3)$$

$$\nabla f\ \mathbf{\Sigma}\ \nabla f^T = b_5^2 \Sigma_{33} + 2 b_3 b_5 \Sigma_{35} + b_3^2 \Sigma_{55} = 0.0336$$

$$\sqrt{0.0336} = 0.18$$

So $b_3 b_5 = 0.98 \pm 0.18$

the one standard deviation
(1-σ) error bar

Is it normally distributed?

Absolutely not! A function of normals is not normal (although, if they are all narrow, it might be close).