System Identification II

Recap and Examples

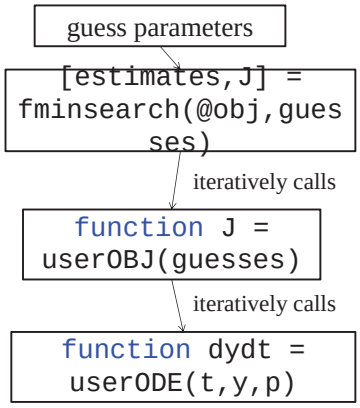
Recap of System Identification

- Determine a possible model
  - use knowledge of mechanism, or appearance of data
  - Simpler is better because:
  - Must have more data than unknown parameters
- Assume an error model
  - use knowledge of mechanism, or appearance of data
- Get initial guesses for the unknown parameters
- Minimize the objective function, J (see next slide)
  - sum of squares of residuals weighted by the error model
- Now have: Parameters, weighted residuals, and J

Weighting Schemes: σ is:

- Constant coefficient of variation (CV)
  - error in the data = constant percent = CV% of the data value
  - σ is proportional to to measurement or model value
- Constant standard deviation (SD)
  - error in the data = constant = SD
  - σ is independent of size of measurement and model
- General Variance Model
  - error in the data = f[A+Bf(b)C]
    - function of the model prediction
    - A, B and C are constants

Using MATLAB's fminsearch



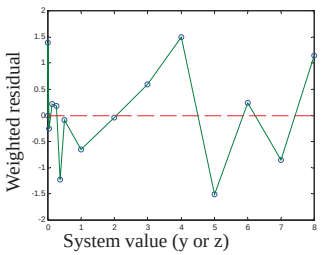
fminsearch
- starts with p = guesses.
- calls userOBJ(p) to get J(p)
- iterates to find ∂J/∂p and move along gradient until at minimum.
- returns p as 'estimates' and J.

userOBJ defined by user:
- calls odesolver, which calls userODE to calculate model.
- calculate y(i,p) from model
- calculates J from model y(i,p), data z(i), and error model w(i).

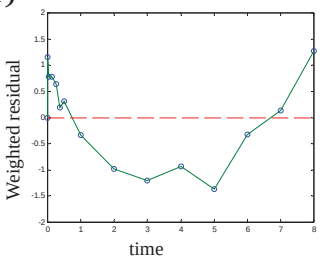
Determining Weights: Goodness of Fit

Weighted residuals reflects the hypotheses about measurement error statistics. *Check*: are the weighted residuals normally distributed and independent?

$$\text{wres(i)} = \frac{z(i) - y(i,p)}{\sigma(i)}, i = 1,\ldots,N$$



OK - normal distributed, independent of value



No – weighted residuals are not independent

Parameter Precisions
(often called Standard Errors)

$$\text{Cov}[p] = \left[\frac{\partial y(p)}{\partial p}^T W \frac{\partial y(p)}{\partial p}\right]^{-1}$$

- Y is the vector of model value for each data point given the parameters (length = N)
- p are the parameters (length = M)
- Remember the notation: ∂y/∂p is the Jacobian again. This time it is how the model values vary as the parameters change. (N by M, so transpose is M x N)
- W is a diagonal matrix of the vector of weights. (N x N)
- Thus Cov(p) is a M x M matrix
- So this reflects how the model behavior is sensitive to variation in each parameter (diagonals) and in pairs of parameters (rest).
- Is accurate only if the weights *reflect* measurement error
- The precisions are a *function* of the richness of the experiment/model

Parameter Precisions

$$\mathrm{Cov}[p] = \left[\frac{\partial y(p)}{\partial p}^T W \frac{\partial y(p)}{\partial p}\right]^{-1} = \left(\begin{bmatrix} \frac{\partial y_1}{\partial p_1} & \cdots & \frac{\partial y_N}{\partial p_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial p_M} & \cdots & \frac{\partial y_N}{\partial p_M} \end{bmatrix} \begin{bmatrix} w_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & w_N \end{bmatrix} \begin{bmatrix} \frac{\partial y_1}{\partial p_1} & \cdots & \frac{\partial y_1}{\partial p_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_N}{\partial p_1} & \cdots & \frac{\partial y_N}{\partial p_M} \end{bmatrix} \right)^{-1}$$

(MxN)(NxN)(NxM)

(Mx1)(NxM)

$$\left(\begin{bmatrix} w_1 \frac{\partial y_1}{\partial p_1} & \cdots & w_N \frac{\partial y_N}{\partial p_1} \\ \vdots & w_c \frac{\partial y_c}{\partial p_r} & \vdots \\ w_1 \frac{\partial y_1}{\partial p_M} & \cdots & w_N \frac{\partial y_N}{\partial p_M} \end{bmatrix} \begin{bmatrix} \frac{\partial y_1}{\partial p_1} & \cdots & \frac{\partial y_1}{\partial p_M} \\ \vdots & \frac{\partial y_r}{\partial p_c} & \vdots \\ \frac{\partial y_N}{\partial p_1} & \cdots & \frac{\partial y_N}{\partial p_M} \end{bmatrix} \right)^{-1}$$

(MxM)

$$Cov(p) = \begin{bmatrix} \sum_{n=1}^N w_n \left(\frac{\partial y_n}{\partial p_1}\right)^2 & \sum_{n=1}^N w_n \frac{\partial y_n}{\partial p_r}\frac{\partial y_n}{\partial p_c} & \\ \vdots & \sum_{n=1}^N w_n \left(\frac{\partial y_n}{\partial p_r}\right)^2 & \vdots \\ & \cdots & \sum_{n=1}^N w_n \left(\frac{\partial y_n}{\partial p_r}\right)^2 \end{bmatrix}^{-1}$$

mth diagonal value is the weighted sum of the squares of how each data prediction varies with the parameter m

Units in Parameter Precisions

Consider individual terms in the matrix prior to inverting:

$\sum_{n=1}^N w_n \frac{\partial y_n}{\partial p_r}\frac{\partial y_n}{\partial p_c}$, with $w_n = \left(\frac{1}{\sigma_n}\right)^2$.

The units of ∂y_n and σ_n are the same, so cancel,

and units are simply inverse of $\partial p_r \partial p_c$.

If one asks change with respect to % error in all parameters,

then units are in $1/\%^2$, and after inversion, are in $\%^2$.

The diagonal elements are the variance (in %) for each parameter

(so square root is the standard deviation)

The off - diagonal elements show the covariance (in %) for each pair.

positive covariance means two parameters work together

negative covarience means two parameters work against each other

Parameter Precisions

In MATLAB

$$\mathrm{Cov}[p] = \left[\left(\frac{\partial y}{\partial p}\right)^T W \frac{\partial y}{\partial p}\right]^{-1}$$

- You don't have an analytic expression for the Jacobian, since you don't have an equation for y(p) in general.
- But, you can calculate it numerically
  - Calculate the y vector (recall this is the model prediction that corresponds to each data point, at the final parameter values p.
    - typically a subset of the variables at a discrete set of time points.
    - Put these all in a single vector, even if they include more than one variable.
  - Increase one parameter a tiny amount (0.1%?) and recalculate the y-vector. Calculate (ynew-y(p))/0.1% to get the derivative.
    - You have the first column of the Jacobian, delydelp.
    - Repeat for each other parameter, keeping all the rest at the original p values.
  - Calculate the diagonal matrix W from the weighting scheme.
  - Perform matrix multiplication and invert the result
  - Consider the diagonal elements to get parameters precisions.
  - Check your units!

Normalized Parameter Precisions

$$\mathrm{CV[p]} = \frac{\sqrt{\mathrm{Var}[p]}}{\mathrm{p}} \times 100 = \frac{\mathrm{SD}[p]}{\mathrm{p}} \times 100$$

- Essentially, this says how much each parameter can vary and not affect the model beyond the assumed errors in the data.
- Usually they are also called %CV (percent coefficient of variation) or %FSD (fractional standard deviations)
- Bad precisions (CV>100%) are often a symptom of a model that is *too complex*, i.e. has too much detail to be resolvable.
- Often, a model will result in bad precisions on one or a few parameters only, meaning they are not well resolved by the model.

Comparing Models

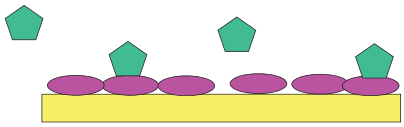
- Now have: Parameters, weighted residuals, and J
- Try different ***guesses for parameters***
  - May get same P, wres, J or different due to local mins
  - If different, go with P and wres that make lower J.
- Try different error/statistical models
  - Better model has more uniformly distributed wres.
  - Cannot use J to compare error models!
- Try different system models. If same # params and same error model, then…
  - Better model has more uniformly distributed wres, and lower J

Comparing Models of Different Complexity

- If different numbers of parameters:
  - If more parameters does not improve J or uniformity of wres, clearly not better
  - But how much better does J need to be to warrant the more complex model?
- Various criteria can be used to weight the number of parameters.
  - Beyond the scope of this class but you can read more..
    - Akaike information criterion (AIC)
    - Bayesian information criterion (BIC)

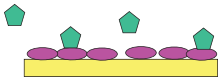
Example 1: Surface Plasmon Resonance (SPR)

- Receptor on gold Surface
- Analyte flows through
- SPR signal reflects mass of material on surface, so *change* in SPR reflects binding.
- Error is constant, since ligand and receptor are only small part of mass on surface.



SPR Example: Equations

- Variables:
  - B(t)= surface concentration of Bound complex
  - S(t) = signal read by SPR
- Parameters:
  - C_0 = Solution concentration of Analyte (assume constant)
  - R_0 = Total surface concentration of Receptor
  - S_0 = sensitivity of SPR to receptor binding
- Equation:

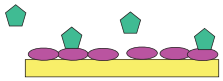


$$\frac{dB}{dt} = k_{on} C_0 (R_0 - B(t)) - k_{off} B(t)$$
$$S(t) = S_0 B(t)$$

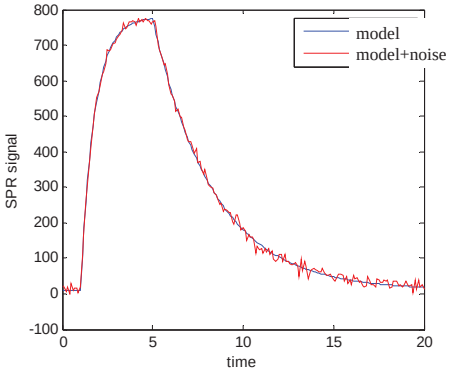
Input for SPR:

- Concentration of analyte is a time-dependent parameter, C(t):
- Injection followed by wash:
  - At time T_{inject}, C(t) jumps from 0 to C_0.
  - At time T_{wash}, C(t) jumps from C_0 to 0.

$$\frac{dB}{dt} = k_{on} C(t)(R_0 - B(t)) - k_{off} B(t) \qquad C(t) = \begin{cases} C_0 \text{ for } T_{inject} < t < T_{wash} \\ 0 \text{ otherwise} \end{cases}$$
$$S(t) = S_0 B(t)$$



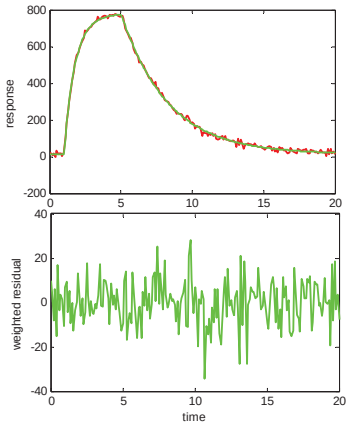
Simulate SPR:



Fit model to data, constant SD:

Parameter estimates:

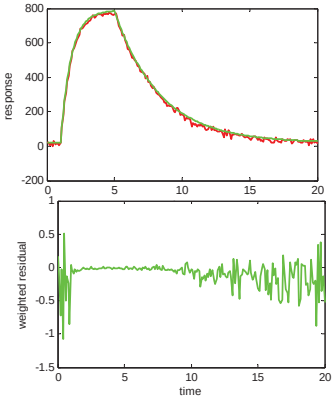
	$k_{on}S_0$	k_{off}
used	.1	0.3
Est.	.099	0.307
error	0.7%	-2.2%



Fit model to data, constant CV:

Parameter estimates:

	$k_{on}S_0$	k_{off}
used	0.1	0.3
Est.	0.098	0.322
error	1.9%	7.4%



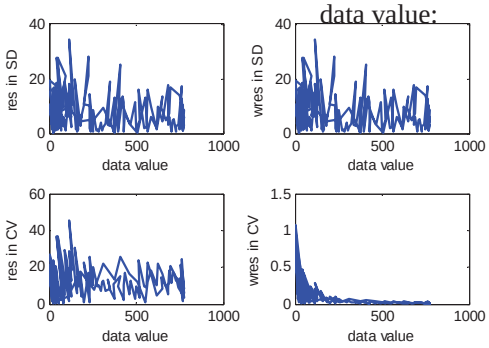
Evaluate Error Model:

SD: assumed error constant

CV: assumed error proportional

Plot the residuals against data value:

Plot the weighted residuals against data value:



Compare to assumptions

How to use Objective Function (J)?

Different Guesses:
Often get different estimates due to local minima
How to decide which is better?
The lower Objective Function

Different models or weighting schemes:
Cannot, in general, compare Objective Functions to pick best.
Why not?

Scale Factor affects J and scale of the weighted residuals, but not the parameters, the evaluation of residuals, or the fit.

Error Modeling Issues:

Recall: $$J(p) = \sum_{i=1}^{N} \left[\frac{z_i - y_i(p)}{\sigma_i} \right]^2$$

For CV, (% error is important; error scales with the value)

Use data value?

$$\sigma_i = CV \cdot z_i$$

$$J(p) = \sum_{i=1}^{N} \left[\frac{z_i - y_i(p)}{CV \cdot z_i} \right]^2$$

Use model value?

$$\sigma_i = CV \cdot y_i(p)$$

$$J(p) = \sum_{i=1}^{N} \left[\frac{z_i - y_i(p)}{CV \cdot y_i(p)} \right]^2$$

•Dividing by model risks biasing to parameters that increase the model value, rather than decrease the difference.
• Dividing by data is more stable.
•But, if we have a zero or near-zero value for the data, this can be unstable.

Why don't we use the standard deviation or SEM of the measurements?

- The standard deviations are only an estimate of the error
- The fewer the points, the worse the estimate
- If two duplicates happen to be very close, the standard deviation undertstates the error at this data value, and the model is too constrained to this point.

Is the Problem Well-Determined?

Do you have enough of the right type of data to determine the parameters?

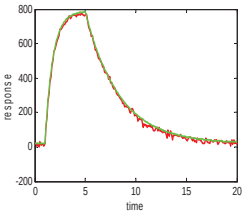
If not:
1. Sample more compartments?
2. Simplify model? (combined parameters?)
3. Create alternate experiment to independently determine a subset of the problem

SPR identifiability:

- Usually don't know receptor concentration R_0 or response scaling, S_0.
- But C_0 is known.
- So 4 unknowns:
  - k_{on}
  - k_{off}
  - R_0
  - S_0

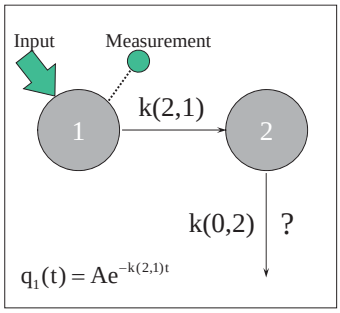
$$\frac{dB}{dt} = k_{on} C_0 (R_0 - B(t)) - k_{off} B(t)$$

$$S(t) = S_0 B(t)$$

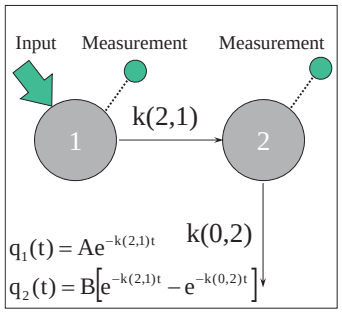


- However, the complexity of our data only justifies three parameters. (rate of approach, rate of decay, and steady state value. If time is too short to estimate steady state, then only two parameters: rate of increase and rate of decay.
- To fix this, must run multiple concentrations.

Model Identifiability Properties
(Must Be Assessed Before Estimation)

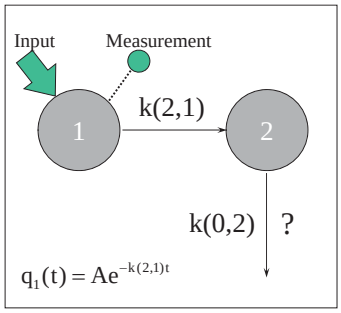


Not identifiable

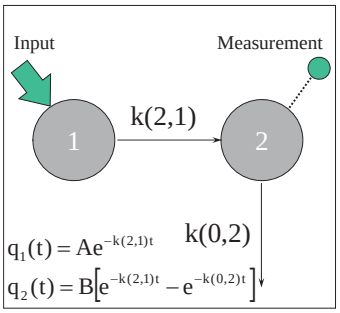


Identifiable

Model Identifiability Properties
(Sometimes Unexpected)



Not identifiable



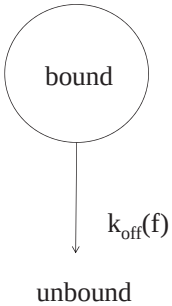
Identifiable, but not uniquely!

Lab Example: bond strength under force

- Simplest model for a bond:
- Compartmental 1-state model
- Unbinding rate = k_{off} is a function of force.

$$k_{off}(f) = k \cdot \exp(f / f_S)$$

- B(t) = number of bonds still formed at time t.
- Simple ODE: $\frac{dB}{dt} = -k_{off}(f) \cdot B(t)$



constant force

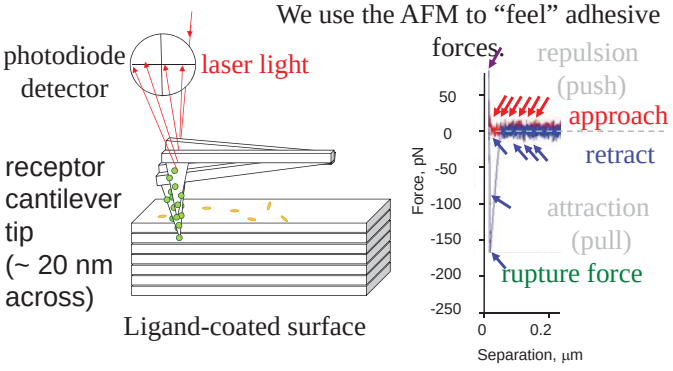
- At constant force, linear ODE with solution:

$$B(t) = B_0 \cdot \exp(-k_{off}(f) \cdot t)$$

- B0 is the number of bonds at the start. Note that bonds are not all there at once – each is tested one at a time, but then we look at all data together. So B(t) means how many bonds are left at time t after the test for each bond starts.
- The rate of exponential decay depends on the force.

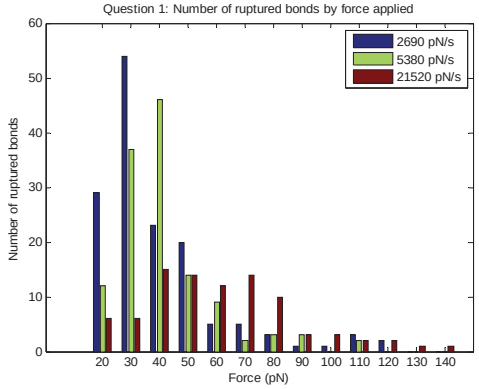
Atomic Force Microscope (AFM)

We use the AFM to "feel" adhesive forces.



Method of single molecule force spectroscopy

Data: Histogram of Bond Rupture Forces in AFM for von Willebrand – GPIb bond



force increases with time

- f(t) = rt, where r is loading rate

$$k_{off}(t) = k_{off}(f(t))$$
$$= k \cdot \exp(-rt / f_S)$$

- ODE is: $\frac{dB}{dt} = -k_0 \exp(-rt / f_S) \cdot B(t)$
- IC B(0)

Comparing Model to Data:

- Data shows number of ***bond ruptures*** as function of ***force***
- Model gives number of ***bonds remaining*** as a function of ***time***.

- Switch time in model to force with f = rt.
- Experimentalist bin the data (eg, cutoffs at 20, 30, 40,.. 140 pN). Number of ruptures in bin is difference between bonds remaining at F(i) vs previous point, F(i-1).
- Also: have three loading rates, r, so have three ODEs.

Match model to data

- ODE is: $\frac{dB}{dt} = -k_0 \exp(-rt / f_S) \cdot B(t)$
- Change to ODE in force

$$\frac{dB(f)}{df} = \frac{dt}{df} \cdot \frac{dB(f)}{dt} = -\frac{k}{r} \cdot \exp(f / f_S) \cdot B(f)$$

- have 3 ODEs: $\frac{dB_i(f)}{df} = -\frac{k}{r_i} \cdot \exp(f / f_S) \cdot B_i(f)$

Parameters

Parameters to be fit

- The amount of force needed to increase rupture rate, $\boldsymbol{f_S}$
- The rupture rate without force, ***k***
- The initial numbers of bonds in each conditions (***B*1(0), *B*2(0), *B*3(0)**. Sum of ruptures includes some nonspecific bonds, and misses some that broke before 10 pN)

Known parameters:

- The three loading rates: $\boldsymbol{r_1, r_2, r_3}$.

Need to fit $\boldsymbol{f_S}$ and ***k*** for all three conditions simultaneously, since they relate to the bond properties, which are the same regardless of loading rate.