Generalized Linear Mixed Models

- GLM + Mixed effects
- Goal: Add random effects or correlations among observations to a model where observations arise from a distribution in the exponential-scale family (other than the normal)
- Why:
  - More than one source of variation (e.g. farm and animal within farm)
  - Account for temporal correlation
  - Provides another way to deal with overdispersion
- Take home message: Can be done, but a lot harder than a linear mixed effect model
- Because: both computation and interpretation issues
Another look at the canonical LME: \( Y = X\beta + Zu + \epsilon \)

Consider each level of variation separately.

A hierarchical or multi-level model

\[
\eta = X\beta + Zu \\
\sim N(X\beta, ZGZ')
\]

\[
Y|\eta = \eta + \epsilon \\
\sim N(\eta, R)
\]

\[
Y|u = X\beta + Zu + \epsilon \\
\sim N(X\beta + Zu, R)
\]

Above specifies the conditional distribution of \( Y \) given \( \eta \) or equivalently \( u \)
To write down a likelihood, need the marginal pdf of $Y$

\[ f(Y, u) = f(Y|u)f(u) \]

\[ f(Y) = \int_{u} f(Y, u)du = \int_{u} f(Y|u)f(u)du \]

When $u \sim N()$ and $\epsilon \sim N()$, that integral has a closed form solution

\[ Y \sim N(X\beta, ZGZ' + R) \]

Extend to GLMs by changing conditional distribution of $Y|u$

- Logistic: $f(Y_i|u) \sim Binomial(m_i, \pi_i(u))$
- Poisson: $f(Y_i|u) \sim Poisson(\lambda_i(u))$
Big problem: Usually no analytic solutions to $f(Y)$

No closed form solution to the integral

Some exceptions:

- $Y | \eta \sim \text{Binomial}(m, \eta)$, $\eta \sim \beta(\alpha, \beta)$
- $Y \sim \text{BetaBinomial}$
- $Y | \eta \sim \text{Poisson}(\eta)$, $\eta \sim \Gamma(\alpha, \beta)$
- $Y \sim \text{NegativeBinomial}$

Ok for one level of additional variability, but difficult (if not impossible) to extend to multiple random effects

Normal distributions are very very nice:

- Easy to model multiple random effects:
  - the sum of Normals is Normal

- Easy to model correlations among observations

Want a way to fit a model like:

$$\mu = g^{-1}(X\beta + Zu), \ u \sim N(0, G)$$

$$Y|\mu = f(\mu)$$
Example: probability of red deer infection by the parasitic nematode *E. cervi*

Expected to vary by deer size (length)

Sampling scheme:

1. 24 farms in Spain. Consider only male deer. 2 farms excluded because no male deer.
2. From 3 to 83 deer per farm. Total of 447 deer.

Response is 1: deer infected with parasite, 0: not

Goals:

1. describe the relationship between length and \( P[\text{infect}] \)
2. predict \( P[\text{infect}] \) for a deer of a specified length

Consider the model \( i \in \{1, 2, \ldots, 447\} \) indexes deer

\[
Y_i \sim \text{Bernoulli}(\pi_i) \\
\text{logit } \pi_i = \mu + \beta l_i,
\]

where \( Y_i \) is infection status \((0/1)\) and \( l_i \) is the length of the deer
Problem: deer not sampled randomly from one population
Two stages: farms, then deer within farm.
Farms are likely to differ.
Consider the model, \( i \in \{1, 2, \ldots, 24\} \) indexes farms, 
\( j \in \{1, 2, \ldots n_i\} \) indexes deer within farm:

\[
Y_{ij} \sim \text{Bernoulli}(\pi_{ij})
\]

\[
\text{logit} \; \pi_{ij} = \mu + \alpha_i + \beta l_{ij}
\]

<table>
<thead>
<tr>
<th>Term</th>
<th>Deviance</th>
<th>( \Delta ) Dev.</th>
<th>df</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>549.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Farm</td>
<td>394.25</td>
<td>155.05</td>
<td>21</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>Length</td>
<td>363.53</td>
<td>30.72</td>
<td>1</td>
<td>&lt; 0.0001</td>
</tr>
</tbody>
</table>
\( \hat{\beta} \) for Length is 0.0391. Each additional 10cm of length multiplies odds of infection by 
\[ e^{10 \times 0.0391} = 1.47 \]

when compared to other length deer on the same farm

Model provides estimates of \( P[\text{infect} | \text{length}] \) for these 24 farms

You need to know the Farm effect to estimate \( P[\text{infect}] \)

Can we say anything about Farms not in the data set?

Yes, if we can assumes that the 24 study Farms are a simple random sample from a population of farms (e.g. in all of Spain)

Consider farm a random effect

\[
Y_{ij} \sim \text{Bernoulli}(\pi_{ij})
\]

\[
\text{logit } \pi_{ij} = \mu + \alpha_i + \beta l_{ij}
\]

\[
\alpha_i \sim N(0, \sigma_F^2)
\]

where \( i \in \{1, 2, \ldots, 24\} \) indexes farms, \( j \in \{1, 2, \ldots, n_i\} \) indexes deer within farm:
Three general approaches to fitting this model

1. GLMM by maximum likelihood
2. GLMM using Bayesian methods, particularly MCMC

The likelihood approach (regular ML, not REML)

Evaluate that untractable integral \( \int_u f(Y|u)f(u)du \) by numerical approximation

1. Gaussian quadrature: intelligent version of the trapezoid rule
2. Laplace approximation: Gaussian quadrature with 1 point

Or avoid the integral by quasilikelihood

1. Penalized Quasi-likelihood: Taylor expansion of \( g^{-1}(X\beta + Zu) \)
2. Pseudolikelihood: similar

Inference about \( b \) conditional on \( \Sigma \)
Bayesian methods

- Evaluate that integral by Markov-Chain Monte-Carlo methods
- Require specifying appropriate prior distributions for parameters
- Hierarchical structure to the model very appropriate for Bayesian methods
- Provides marginal inference about $\boldsymbol{b}$
  i.e., includes the uncertainty associated with estimation of $\Sigma$
Generalized Estimating Equations

- Avoid the integral by ignoring (temporarily) the random effects
- Assume a convenient “working correlation matrix”.
  e.g. independence
- Estimate parameters using the working correl. matrix
  - Estimates are not as efficient as those from model with the correct variance structure
  - But loss of efficiency often not too large
  - And estimates can be computed much more easily if assume independence
- Real problem is the \( \text{Var}_W \hat{\beta} \) computed from the working correl. matrix: usually badly biased
A better estimator of \( \text{Var} \hat{b} \):

Remember \( \text{Var} \hat{\beta} \) when \( \Sigma \) misspecified:

\[
\text{Var} \hat{b} = (X'X)^{-1} X' \Sigma X (X'X)^{-1} = \frac{\text{Var}_w \hat{b}}{\sigma^2} X' \Sigma X \frac{\text{Var}_w \hat{b}}{\sigma^2}
\]

Imagine there is an estimate of \( \Sigma \), call it \( C \), usually computed from replicate data.

Use the mis-specified variance estimator to patch-up \( \text{Var} \hat{\beta} \):

\[
\text{Var} \hat{b} = \frac{\text{Var}_w \hat{b}}{\sigma^2} X' C X \frac{\text{Var}_w \hat{b}}{\sigma^2}
\]

Sometimes called the Sandwich estimator (bread, filling, bread)

Same idea, but many more details and different equations for GLMM
There is a major, important difference between the model fit by GEE and the model fit by GLMM.

\[
\text{GLMM} \quad \mathbb{E} Y | u = g^{-1}(X\beta + Zu) \quad (1)
\]
\[
\text{GEE} \quad \mathbb{E} Y = g^{-1}(X\beta) \quad (2)
\]

(1) models the conditional mean of \( Y \) given the random effects.

Influence of length of deer randomly selected within a farm.

(2) models the marginal mean of \( Y \).

Influence of length on deer randomly selected from the population.

These are the same for identity link, \( g^{-1}(x) = x \), usually used with normal distributions.

Not the same for other link functions (logit, log).
## Results from various estimation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Intercept estimate</th>
<th>Intercept se</th>
<th>Slope estimate</th>
<th>Slope se</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regr.</td>
<td>-3.30</td>
<td>0.946</td>
<td>0.0245</td>
<td>0.0056</td>
</tr>
<tr>
<td>GEE (naive)</td>
<td>-3.90</td>
<td>0.920</td>
<td>0.0288</td>
<td>0.0056</td>
</tr>
<tr>
<td>GEE (fixed)</td>
<td></td>
<td></td>
<td>1.132</td>
<td>0.0071</td>
</tr>
<tr>
<td>LR w/farm</td>
<td></td>
<td></td>
<td>0.0391</td>
<td>0.0076</td>
</tr>
<tr>
<td>GLMM (Laplace)</td>
<td>-5.03</td>
<td>1.273</td>
<td>0.0374</td>
<td>0.0072</td>
</tr>
<tr>
<td>GLMM (Gauss Q)</td>
<td>-5.03</td>
<td>1.273</td>
<td>0.0374</td>
<td>0.0072</td>
</tr>
<tr>
<td>GLMM (Resid PL)</td>
<td>-4.87</td>
<td>1.246</td>
<td>0.0357</td>
<td>0.0071</td>
</tr>
</tbody>
</table>

- Big difference is between marginal and conditional models
So which is the right approach?
My answer is that it depends on goal of study.
Sometimes called population averaged and subject-specific models
This helps identify most appropriate method for specific goal
Example: influence of cholesterol on P[heart attack]
Data are observations on individuals made every 3 months:
(Chl at start of period, Heart attack during period?)
Two slightly different questions
1. If I change my diet and reduce my cholesterol from 230 to 170, how much will I reduce my probability of a heart attack?
   want conditional = subject specific estimate of log odds
2. Public health official: If we implement a nationwide program to reduce cholesterol from 230 to 170, how much will we reduce the number of heart attacks?
   # heart attacks = P[heart attack] \times \text{population size}
   want marginal = population averaged estimate of log odds
Computing for GLMM’s

- Only code for fitting a GLMM is included here
- All the code to produce the plots in this section is in deer2.r on the class web site

```r
deer <- read.csv('deer.csv', as.is=T)
deer$farm.f <- factor(deer$Farm)

library(lme4)
# use glmer() to fit GLMM.
# Farm is a unique identifier for a cluster;
# does not need to be a factor
deer.glmm <- glmer(infect~Length+(1|Farm),
                  data=deer,family=binomial)

# default in glmer() is ML estimation.
```
Computing for GLMM’s

# have all the lmer() helper functions
# coef(), fixef(), vcov()
# summary(), print()
# anova()
# full list found in ?mer (look for Methods)

# default is Laplace approximation
# shift to Gaussian quadrature by specifying
# nAGQ = # quadrature points

deer.glmm2 <- glmer(infect~Length+(1|Farm),
                     data=deer,family=binomial, nAGQ = 5)
Computing for GEE’s

# GEE is in the gee library
# arguments are formula, data, and family as in glm()
# id=variable has a unique value for each cluster
# DATA must be sorted by this variable
# help file implies any type of variable will work,
# but my experience is that this needs to be a
# number or a factor

deer <- deer[order(deer$Farm),]

deer.gee <- gee(infect~Length, id=farm.f, data=deer,
  family=binomial, corstr='exchangeable')
# then the working correlation matrix, as corstr =
# I used exchangeable = Compound symmetry to get the
# results shown in lecture
even though the lecture material focused on independence. Results are not quite the same
General advice about GEE is to use a working correlation close to the suspected true correlation model, that’s exchangeable here

summary() produces a lot of output here because it prints the working correl matrix for the largest cluster. That’s 83x83 for the deer data.
just get the coefficients part
summary(deer.gee)$coeff
Nonlinear Models

- So far the models we have studied this semester have been linear in the sense that our model for the mean has been a linear function of the parameters.
- We have assumed $E(y) = X\beta$
- $f(X_i, \beta) = X'_i\beta$ is said to be linear in the parameters of $\beta$ because $X'_i\beta = X_{i1}\beta_1 + X_{i2}\beta_2 + \ldots + X_{ip}\beta_p$ is a linear combination of $\beta_1, \beta_2, \ldots, \beta_p$.
- $f(X_i, \beta) = X'_i\beta$ is linear in $\beta$ even if the predictor variables, the $X'$s are nonlinear functions of other variables.
For example, if
\[ X_{i1} = 1 \]
\[ X_{i2} = \text{Amount of fertilizer applied to plot } i \]
\[ X_{i3} = (\text{Amount of fertilizer applied to plot } i)^2 \]
\[ X_{i4} = \log(\text{Concentration of fungicide on plot } i) \]

\[ f(X_i, \beta) = X_i'\beta = X_{i1}\beta_1 + X_{i2}\beta_2 + X_{i3}\beta_3 + X_{i4}\beta_4 \]
\[ = \beta_1 + \text{fert}_i\beta_2 + \text{fert}_i^2\beta_3 + \log((fung)_i)\beta_4 \]

is still linear in the parameters \( \beta_1, \beta_2, \beta_3, \beta_4 \).

Now, we consider nonlinear models for the mean \( E(y_i) \).

These are models where \( f(X_i, \beta) \) cannot be written as a linear combination of \( \beta_1, \beta_2, \ldots, \beta_p \).

Small digression: What about models that can be transformed to be linear in the parameters?
linearizing a non-linear model

- Example: Michaelis-Menton enzyme kinetics model

\[ v_s = \frac{v_m S}{S + K_m} \]

- \( S \) is concentration of substrate, \( v_s \) is reaction rate at \( S \)
- \( v_m \) is maximum reaction rate,
- \( K_m \) is enzyme affinity= \( S \) at which \( v_s = v_m/2 \)

- Function is mathematically equivalent to:
  - Lineweaver-Burke:
    \[ \frac{1}{v_s} = \frac{1}{v_m} + \frac{K_m}{v_m} \frac{1}{S} \]
    Linear regression of \( Y = 1/v_s \) on \( X = 1/S \)
  - Hanes-Woolf:
    \[ \frac{S}{v_s} = \frac{K_m}{v_m} + \frac{1}{v_m} S \]
    Linear regression of \( Y = S/v_s \) on \( X = S \)

- Both are linear regressions
The graph shows the relationship between substrate concentration and reaction velocity. The x-axis represents the substrate concentration, ranging from 0 to 100, while the y-axis represents the reaction velocity, ranging from 0.5 to 2.0. The data points suggest a positive correlation between the two variables.
However, the estimators of $v_m$ and $K_m$ derived from each model are not the same.

Illustrate numerically: LS estimates from each model

<table>
<thead>
<tr>
<th>Model</th>
<th>$\hat{\beta}_0$</th>
<th>$\hat{\beta}_1$</th>
<th>$v_m$</th>
<th>$\hat{v}_m$</th>
<th>$K_m$</th>
<th>$\hat{K}_m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>nonlin</td>
<td>2.05</td>
<td>9.12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L-B</td>
<td>0.377</td>
<td>5.64</td>
<td>$1/\beta_0$</td>
<td>2.65</td>
<td>$\beta_1/\beta_0$</td>
<td>14.96</td>
</tr>
<tr>
<td>H-W</td>
<td>4.74</td>
<td>0.482</td>
<td>$1/\beta_1$</td>
<td>2.07</td>
<td>$\beta_0/\beta_1$</td>
<td>9.83</td>
</tr>
</tbody>
</table>

Why?

Because the statistical model adds a specification of variability to the mathematical model, e.g.

$$v_i = \frac{v_m S_i}{S_i + K_m} + \varepsilon_i, \quad \varepsilon_i \sim (0, \sigma^2)$$
And

\[ v_i = \frac{\nu_m S_i}{S_i + K_m} + \varepsilon_i, \quad \varepsilon_i \sim (0, \sigma_1^2) \]  \hspace{1cm} (3)

is not the same as

\[ \frac{1}{\nu_1} = \frac{1}{\nu_m} + \frac{K_m}{\nu_m S_i} + \varepsilon_i, \quad \varepsilon_i \sim (0, \sigma_2^2) \] \hspace{1cm} (4)

If you work out all the details, (2) is equivalent to (1) with unequal variances.

The statistical models for MM, L-B, and H-W are different.

Estimates differ because

- Different variance models
- Leverage of specific observations is not the same
linearizing a non-linear model: 2nd example

- Exponential growth model
  $$Y_i = \beta_0 e^{\beta_1 T_i}$$

- Nonlinear form, constant variance:
  $$Y_i = \beta_0 e^{\beta_1 T_i} + \epsilon_i, \quad \epsilon_i \sim (0, \sigma_1^2)$$

- Linearized form, constant variance, normal dist.:
  $$Y^*_i = \log Y_i = \log \beta_0 + rT_i + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma_2^2)$$

- Statistically equivalent to
  $$Y_i = \beta_0 e^{\beta_1 T_i} \times e^{\epsilon_i}, \quad \epsilon_i \sim N(0, \sigma_2^2)$$

- i.e., errors are multiplicative log normal with constant lognormal variance
Three classes of models:

1. Linear
2. Transformable to linear, e.g. MM or exp. growth
3. Intrinsically nonlinear, e.g. 
   \[ Y(t) = N_1 e^{-r_1 T} + N_2 e^{-r_2 T} \]

Why would we want to consider a nonlinear model?

Pinheiro and Bates (2000) give some reasons:

1. Mechanistic - based on theoretical considerations about the mechanism producing the response.
2. Often interpretable and parsimonious
3. Can be valid beyond the range of the observed data.

I add: because the implied variance model (usually constant variance for untransformed observations) may be more appropriate for the data.
Geometry of nonlinear least squares

- Remember the geometry of LS for a linear model
Add $\beta$ values to $C(X)$
Example 3: A 1 parameter nonlinear model

Classic data set, “Rumford” data: how quickly does a cannon cool?

15’th - 19’th century cannons made by forging a big piece of metal, then boring out the tube in the middle.

Boring generates a lot of heat. Doesn’t work if the cannon gets too hot. Have to stop and wait for cannon to cool

Count Rumford: how long does this take? Developed the physics leading to:

\[ Y_i = T_{env} + (T_{init} - T_{env})e^{-rX_i} \]

\( T_{env} \) and \( T_{init} \) are temp in the environment and cannon’s initial temperature, \( Y_i \) is temp at time \( X_i \)

Collected data to see if this model was appropriate. Cannon heated to 130 F. Environment is 60 F. Measured temp at set times.

\[ Y_i = 60 + 70e^{-rX_i} \]
Assume errors in temperature measurement have constant variance

\[ Y_i = 60 + 70e^{-rX_i} + \epsilon_i, \quad \epsilon_i \sim (0, \sigma^2) \]

Equation is non-linear in the parameter, \( r \)

But, least squares is still a reasonable way to define an estimator

Estimate \( r \) by finding the \( r \) that minimizes

\[
L(r) = (Y_i - \hat{Y}(r)_i)^2 = \left( Y_i - (60 + 70e^{-rX_i}) \right)^2
\]

Consider fitting this model to two data points: (15,118), (60, 102)
The graph demonstrates the relationship between time and temperature for different rates (r).

- For $r=0.001$, the temperature decreases rapidly at the beginning and then slows down.
- For $r=0.01$, the decrease is slower compared to $r=0.001$ but still noticeable in the early stages.
- For $r=0.1$, the temperature decreases at a moderate rate throughout the observed time.

As time increases, the temperature for each rate shows a consistent downward trend.
The expectation surface, \((\hat{Y}(r)_{X=15}, \hat{Y}(r)_{X=60})\),

1. Expectation surface is curved
2. Points not equally spaced (considered as function of \(r\))
Can write the same nonlinear model using different parameters.

\[ Y_i = 60 + 70e^{-e^\theta X_i} + \epsilon_i, \quad \epsilon_i \sim (0, \sigma^2), \quad \theta = \log r \]
The expectation surface, \( (\hat{Y}(\theta)_{X=15}, \hat{Y}(\theta)_{X=60}) \),

1. Expectation surface is same manifold
2. But spacing of points not the same (more evenly spaced for \( \theta \))
Y is still the closest point on the expectation surface.
LS estimate of r is the parameter corresponding to that point.
But the geometry is (or can be) very different
  - May be more than one closest point.
  - Residual vector may not be perpendicular to (the tangent line) to the expectation surface, e.g., (15,135), (60,132)

Advanced discussions on nonlinear regression consider consequences of two types of curvature
  - Parameter effect curvature: deviation from equal spacing along expectation surface
    Can reduce by reparameterizing model
  - Intrinsic curvature: curvature of expectation surface
    Characteristic of model
Example 4: Logistic Growth Model

- $y_i$ is the height of a tree at age $X_i (i = 1, \ldots, n)$
Want a model in which:
- trees grow slowly, then quickly, then slowly
- trees have constant final height
- the final height needs to be estimated

One (of many) asymptotic growth models is the 3 parameter logistic

\[ E(y_i) = f(X_i, \beta) = \frac{\beta_1}{1 + e^{-(X_i - \beta_2)/\beta_3}} \]

Interpretation of parameters:
- \( \beta_1 \) is final height
- \( \beta_2 \) is age at which height is \( \beta_1/2 \)
- \( \beta_3 \) is the growth rate,
  # years to grow from 0.5\( \beta_1 \) to \( \beta_1/(1 + e^{-1}) \approx 0.73\beta_1 \)

Statistical model:

\[ y_i = f(X_i, \beta) + \epsilon_i, \ E(\epsilon_i) = 0, \ Var(\epsilon_i) = \sigma^2, \ i = 1, \ldots, n \]
Least Squares Estimation $y_i = f(X_i, \beta) + \epsilon_i \quad i = 1, \ldots, n$

Find $\hat{\beta}$ that minimizes $g(b) = \sum_{i=1}^{n} [y_i - f(X_i, b)]^2$
• Candidate $\hat{\beta}$ is the solution to the estimating equations:

$$\frac{\partial g(b)}{\partial b} = 0$$

• These are:

$$\frac{\partial g(b)}{\partial b_1} = 2 \sum_{i=1}^{n} [y_i - f(X_i, b)] \frac{\partial f(X_i, b)}{\partial b_1}$$

$$\vdots$$

$$\frac{\partial g(b)}{\partial b_p} = 2 \sum_{i=1}^{n} [y_i - f(X_i, b)] \frac{\partial f(X_i, b)}{\partial b_p}$$
Can write as a matrix equation

\[ f(X, b) = \begin{bmatrix} f(X_1, b) \\ \vdots \\ f(X_n, b) \end{bmatrix} \]

Define \( f(X, b) = \)

And

\[ D' = \begin{bmatrix} \frac{\partial f(X_1, b)}{\partial b_1} & \ldots & \frac{\partial f(X_n, b)}{\partial b_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f(X_1, b)}{\partial b_p} & \ldots & \frac{\partial f(X_n, b)}{\partial b_p} \end{bmatrix} \]

Then \( \frac{\partial g(b)}{\partial b} = 0 \) is equivalent to \( D'[y - f(X, b)] = 0 \)

In the linear case, \( D' = X' \) and \( D'[y - f(X, b)] = 0 \) becomes \( X'[y - Xb] = 0 \) \( \Rightarrow \) \( X'Xb = X'y \)
In the nonlinear case, \( D' \) depends on \( \beta \) so that the equation \( D'[y - f(X, b)] = 0 \) has (usually) no analytic solution for \( b \).

For example, for the logistic model,

\[
\begin{align*}
\frac{\partial f(X_i, \beta)}{\partial \beta_1} &= \frac{1}{1 + \exp\left\{-(X_i - \beta_2)/\beta_3\right\}}
\end{align*}
\]

\[
\begin{align*}
\frac{\partial f(X_i, \beta)}{\partial \beta_2} &= -\beta_1 \exp\left\{-(X_i - \beta_2)/\beta_3\right\}
\end{align*}
\]

\[
\begin{align*}
\frac{\partial f(X_i, \beta)}{\partial \beta_3} &= -\beta_1 \exp\left\{-(X_i - \beta_2)/\beta_3\right\} \frac{(X_i - \beta_2)}{[1 + \exp\left\{-(X_i - \beta_2)/\beta_3\right\}]^2 \beta_3}
\end{align*}
\]
Various algorithms to find minimum analytically

Very common one for nonlinear regression is the Gauss-Newton algorithm

Taylor’s theorem:

\[ f(x_i, b) \approx f(x_i, b^*) + \left[ \frac{\partial f(x_i, b)}{\partial b} \right]_{b=b^*} (b - b^*) \]

So \( E = f(X, \beta) \) can be approximated by

\[ f(X, b) \approx f(X, b^*) + \hat{D}(b - b^*) \]

where \( \hat{D} \) is \( D \) evaluated at \( b = b^* \).

Notice this is a linear regression where \( X \) is \( \hat{D} \)

\[ f(X, b) \approx f(X, b^*) - \hat{D}b^* + \hat{D}b \\
\approx \text{constant} + \hat{D}b \]
Gauss-Newton algorithm

1. Choose a starting value, $b_0$
2. Calculate $D$ for $b = b_0$
3. Estimate $\hat{b}$ using approximating linear model
4. Call this $b_1$
5. Calculate $D$ for $b = b_1$
6. Repeat steps 3-5 until convergence

Various ways to define convergence

1. Little to no change in $b$ after an iteration
   “not making progress” convergence
2. Little to no change in SSE, $g(b)$, after an iteration
   “not making progress” convergence
3. $\frac{\partial g(b)}{\partial b}$ evaluated at $b_i$ is sufficiently close to 0
   “close to goal” convergence
Choice of starting value can really matter

Nice to have a starting value close to the overall minimizer
  - Taylor expansion is a close approximation to the nonlinear function, so convergence will be quick
  - less likely to get stuck at some local minimum.

Good idea to try multiple starting values.

Would like to get to same solution from each starting value

Often implementations of the G-N algorithm impose a maximum number of iterations. Often 50 or 100.

If doesn’t converge, try different starting value or increase the number of iterations

Relaxing the convergence criterion is something to be used only if really desperate.
Reported “solution” may be close, but probably not.
Continue iterating until some convergence criterion is met.

Possible convergence criteria:

- \[ \sum |\beta^r - \beta^{r-1}| < \text{small constant} \]
- \[ \max_{j=1,\ldots,p} \frac{|b_j^r - b_j^{r-1}|}{|b_j^{r-1}| + \epsilon} < \text{small constant} \]
- \[ g(b^{r-1}) - g(b^r) < \text{small constant} \]
- \[ \sum \text{abs} \left( \frac{\partial g(b)}{\partial b} \bigg|_{b=b^r} \right) < \text{small constant} \]
Add assumption of normal distribution to our error model

The model is now:

\[ y_i = f(x, \beta) + \epsilon_i, \ i = 1, \ldots, n, \ \epsilon_1, \ldots, \epsilon_n \text{ i.i.d. } \sim N(0, \sigma^2) \]

Let \( \hat{\beta} \) be the least squares estimate of \( \beta \)

If \( n \) sufficiently large,

\[ \hat{\beta} \sim N(\beta, \sigma^2(\hat{D}'\hat{D})^{-1}) \]

where \( \hat{D} \) is \( D \) evaluated at \( \hat{\beta} \)

because if \( n \) large, \( f(x, b) \approx \text{constant} + \hat{D}b \)

where \( \hat{D} \) is \( D \) evaluated at \( b \)

\( \sigma^2(\hat{D}'\hat{D})^{-1} \) can be estimated by \( \text{MSE}(\hat{D}'\hat{D})^{-1} \), where \( \hat{D} \) is \( D \) evaluated at \( \hat{\beta} \)
MSE is estimated in the obvious way

- Define \( p \) = number of parameters
- \( MSE = \frac{SSE}{n-p} \).
- \( SSE = g(\hat{\beta}) = \sum_{i=1}^{n}[y_i - f(X_i, \hat{\beta})]^2 \).

For \( n \) sufficiently large, \( \frac{(n-p)MSE}{\sigma^2} = \frac{SSE}{\sigma^2} \sim \chi^2(n-p) \)

All the linear model inference follows, using \( \hat{D} \) as the “\( X \)” matrix

An approximate F-test \( H_0: C\beta = 0 \) rejects \( H_0 \) at level \( \alpha \) if and only if
\[
F = \frac{\hat{\beta}'C'[C(\hat{D}'\hat{D})^{-1}C']^{-1}C\hat{\beta}}{MSE} \geq F_{\alpha}^{(\alpha)}
\]
with \( q = \text{rank}(c) \) = number of rows of \( C \).

An approximate 100(1 – \( \alpha \))% confidence interval for \( C'\beta \) is
\[
C'\hat{\beta} \pm t_{n-p}^{\alpha} \sqrt{MSE \ C'(\hat{D}'\hat{D})^{-1}C}
\]
We also have approximate F tests for reduced vs. full model comparisons:

\[
F = \frac{(SSE_{\text{reduced}} - SSE_{\text{full}})/(df_{\text{reduced}} - df_{\text{full}})}{SSE_{\text{full}}/df_{\text{full}}}
\]

\[
H_0 \sim F_{df_{\text{reduced}} - df_{\text{full}}, df_{\text{full}}}
\]

For example, consider a test of \( H_0 : \beta_1 = \beta_{10} \) vs. \( H_1 : \beta_1 \neq \beta_{10} \) for some fixed \( \beta_{10} \). Let \( \beta_2 = \begin{bmatrix} \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} \).

Let \( f_0(x, \beta_2) = f(x, \begin{bmatrix} \beta_{10} \\ \beta_2 \end{bmatrix}) \).

Then the reduced model is

\[
y_i = f_0(x_i, \beta_2) + \epsilon_i \quad i = 1, \ldots, n \quad \epsilon_1, \ldots, \epsilon_n \overset{i.i.d.}{\sim} N(0, \sigma^2)
\]

Then \( F(\beta_{10}) \equiv \frac{SSE_{\text{reduced}} - SSE_{\text{full}}}{MSE_{\text{full}}} \sim F_{1, n-p} \).
Confidence intervals

- Two ways to get a confidence interval for $\beta_1$
  1. **Wald interval:**
     \[ \hat{\beta}_1 \pm t_{n-p}^{\alpha} \sqrt{\text{MSE} \ (\hat{D}' \hat{D})^{-1}} \]
  2. **“profile” interval:**
     - Consider all $\beta_{10}$. Include in $1 - \alpha$ confidence interval all those $\beta_{10}$ for which the F test accepts Ho: $\beta_1 = \beta_{10}$ at level $\alpha$.
     - The set \( \{ \beta_{10} : F(\beta_{10}) \leq F_{1,n-p}^{(\alpha)} \} \) is an approximate 100(1 − $\alpha$)% confidence set for $\beta_1$.

- Same interval for linear models
- **Not the same for a nonlinear model**
- Reparameterization of $\beta$, e.g. exp $\beta$, changes Wald interval.
- No effect on profile interval.
- Wald interval assumes SSE surface quadratic around estimate
- Wald intervals commonly used because they’re easier to compute.
  For careful work, use profile intervals.
Example: Confidence interval for Rumford temperature change

Model 1: \( \text{temp}_i = 60 + 70 \times \exp(-r \times \text{time}_i) + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2) \)

Fit to Rumford data: \( \hat{r} = 0.0094, \text{se} \hat{r} = 0.00042, \text{rMSE} = 1.918 \)

Model 2:

\( \text{temp}_i = 60 + 70 \times \exp(-\exp(t) \times \text{time}_i) + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2) \)

Fit to Rumford data: \( \hat{t} = -4.665, \text{se} \hat{t} = 0.044, \text{rMSE} = 1.918, \exp(-4.665) = 0.0094 \)

<table>
<thead>
<tr>
<th>Data</th>
<th>Model</th>
<th>Wald interval</th>
<th>Profile interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumford</td>
<td>1</td>
<td>(0.0085, 0.0103)</td>
<td>(0.0085, 0.0103)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(-4.762, -4.568)</td>
<td>(-4.767, -4.571)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.0085, 0.0104)</td>
<td>(0.0085, 0.0103)</td>
</tr>
<tr>
<td>Noisy</td>
<td>1</td>
<td>(0.0084, 0.0168)</td>
<td>(0.0087, 0.0171)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(-4.702, -4.044)</td>
<td>(-4.746, -4.071)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.0091, 0.0175)</td>
<td>(0.0087, 0.0171)</td>
</tr>
</tbody>
</table>
Profile SS Error for $r$ parameterization

![Graph showing the profile SS Error for $r$ parameterization. The x-axis represents $r$ with values ranging from 0.006 to 0.014, and the y-axis represents SS Error with values ranging from 100 to 600. The graph shows a curve with a minimum SS Error at approximately $r = 0.010$.](image-url)
Profile SS Error for $t = \exp(r)$ parameterization
A useful property of nonlinear models

- Consider a model: \( \mathbb{E} Y = f(X, \beta) \)
- \( \hat{\beta} \) satisfies the normal equations: \( D' [Y - f(X, \beta)] = 0 \)
  where \( D' \) is the matrix of partial derivatives with respect to \( \beta \)
- And \( \text{Var} \beta = MSE(D'D)^{-1} \)
- The real interest is in a new set of parameters computed from \( \beta \): Call these \( \alpha \), where \( \alpha_i = g_i(\beta) \)
- Using invariance of MLE's: \( \hat{\alpha}_i = g_i(\hat{\beta}) \)
- How to obtain variance-covariance matrix of \( \hat{\alpha} \)?
- Define \( G \) as the matrix of partial derivatives of \( \alpha \) with respect to \( \beta \).

\[
G_{ij} = \frac{\partial \alpha_i}{\partial \beta_j}
\]
Two ways:

1. Delta method: \( \text{Var} \alpha = G \text{Var} \beta G' \)
2. Fit a model using the \( \alpha \) parameterization, i.e.
   \[
   y = f^*(X, \alpha) = f(X, g(\beta))
   \]

The variances are exactly the same. Can prove using chain rule.

One of the models may be linear, but usually at least one model is nonlinear.

Remember that inference either using the delta method or using nonlinear regression is only asymptotic.
Example: location of minimum/maximum of a quadratic function

\[ Y_i = \beta_0 + \beta_1 X_i + \beta_2^2 X_i^2 + \varepsilon_i \]

Estimated location of min/max is \( X_m = -\beta_1/(2\beta_2) \)

Can estimate \( X_m \) = location of min/max and its asymptotic variance directly by fitting the nonlinear model

\[ Y_i = \beta_0 + \beta_2 (X_i - X_m)^2 + \varepsilon_i \]

Wald confidence interval matches Delta method ci from linear regression

Profile confidence interval performs better
Change-point models

- Short detour through regression models with dummy variables
- We’ve seen indicator (0/1) variables used to represent group-specific means, group-specific intercepts, and groups-specific slopes
- They are also used in “change-point” problems.
- Suppose we are relating $Y$ and $x$ and expect a change in slope at $x = 100$. A possible model is

$$Y_i = \beta_0 + \beta_1 x_i + \beta_2 (x_i - 100) z_i + \epsilon_i$$

where $z_i = 1$ if $x_i > 100$ and 0 otherwise

$$E(Y|x) = \beta_0 + \beta_1 x \text{ (for } x \leq 100)$$
$$E(Y|x) = \beta_0 + \beta_1 x + \beta_2 (x - 100) \text{ (for } x > 100)$$
slope changes from $\beta_1$ to $\beta_1 + \beta_2$ at $x = 100$
if change point is unknown then can replace 100 by parameter $\tau$

$$Y_i = \beta_0 + \beta_1 x_i + \beta_2 (x_i - \tau) I(x_i > \tau) + \epsilon_i \quad (5)$$

$E Y_i \mid x_i$ is a non-linear function of $\tau$; need non-linear regression to estimate $\hat{\tau}$.

A common variation is “segmented” regression: second part is flat

$$EY_i = \begin{cases} 
\beta_0 + \beta_1 x_i & x_i \leq \tau \\
\beta_0 + \beta_1 \tau & x_i > \tau 
\end{cases} \quad (6)$$

$$EY_i = \beta_0 + \beta_1 x_i (1 - z_i) + \beta_1 \tau z_i$$

If $\tau$ unknown, need one of these two forms and NL regression

If $\tau$ known, replace all $x_i > \tau$ with $\tau$ and use OLS

Both (5) and (6) are continuous, but 1st derivative is not.
Quadratic variation has continuous first derivative:
Quadratic increase to maximum, then flat.
Easiest to write in non-linear form

\[ EY_i = \begin{cases} 
\beta_0 + \beta_1(\tau - x_i)^2 & x_i \leq \tau \\
\beta_0 & x_i > \tau 
\end{cases} \]
“Change-point” model: $E Y_i \mid x_i$ “jumps” at $\tau$.

Trivial to estimate (2 means) if $\tau$ known. Use NL regression if need to estimate it.

$$EY_i = \beta_0 + \beta_1 I(x_i < \tau)$$
Computing for nonlinear models

rumf <- read.table('rumford.txt', header=T)

# fit non-linear model
# the formula gives the model
# start is a list of name=constants
# the names are the parameter names in the model
# (here only one, r)
# the constants are the starting values
# can also specify a vector of possible starting
# values for each parameter
# every variable in the model needs to either be
# a parameter, i.e. in the start list
# or a variable, i.e. in the data frame

rumf.nls <- nls(temp~60 + 70*exp(-r*time), data=rumf, start=list(r=0.01))
Computing for nonlinear models

# many helper functions, including:
#  summary()
#  coef(), vcov()
#  logLik(), deviance(), df.residual()
#  predict(), residuals()
#  anova()
# each does the same thing as corresponding lm
#  helper function except for anova:
#  you need to provide the sequence of models
#  e.g: evaluate exponential quadratic in time

rumf.nls2 <- nls(temp~60 + 70*exp(-r*time-r2*time^2),
                 data=rumf, start=list(r=0.01,r2=-0.0001))

anova(rumf.nls, rumf.nls2)
# estimated coefficients

coeff(rumf.nls)

# profile ci’s on parameters

cconfint(rumf.nls)

# residual vs predicted values plot

plot(predict(rumf.nls), resid(rumf.nls))
plot(predict(rumf.nls2), resid(rumf.nls2))

rumf.nls3 <- nls(temp~init + delta*exp(-r*time), data=rumf, start=list(r=0.01, init=60, delta=70))
Nonlinear mixed models

- Can add additional random variation to Nonlinear models
- Easy version: use additive random effects to model correlated observations
  \[ Y_{ij} = f(X_i, \beta) + u_i + \varepsilon_{ij} \]
- More flexible: values of \( \beta \) depend on subject
- First order compartment model with absorption
  e.g. swallow a pill with dose \( D \), absorbed into blood, removed by kidneys
- Two compartments: stomach, blood
  \( A_s \): amount in stomach, \( A_b \): amount in blood
  \[
  \frac{dA_s}{dt} = -k_a A_s \\
  \frac{dA_b}{dt} = k_a A_s - k_e A_b
  \]
Solution gives blood concentration, $C(t)$, at time $t$:

$$C(t) = \frac{k_ak_eD}{V_c} \frac{(e^{-k_a t} - e^{-k_e t})}{k_e - k_a}$$

Picture on next slide: $D = 100$, $k_a = 0.1$, $k_e = 0.02$, $V_c = 1$
Nonlinear mixed models

- Such models fit to data collected on one or more individuals over time
- Allow the parameters to vary among individuals
- Permits inference to unobserved individuals

\[ E \ C(t) \mid [k_{ai}, k_{ei}, k_{vi}]' = \frac{k_{ai} k_{ei} D}{V_{ci}} \left( e^{-k_{ai} t} - e^{-k_{ei} t} \right) \]

\[
\begin{bmatrix}
k_{ai} \\ k_{ei} \\ V_{ci}
\end{bmatrix} \sim N\left(\begin{bmatrix}
k_a \\ k_e \\ V_c
\end{bmatrix}, \begin{bmatrix}
\sigma_a^2 & \sigma_{ae} & \sigma_{av} \\
\sigma_{ae} & \sigma_e^2 & \sigma_{ev} \\
\sigma_{av} & \sigma_{ev} & \sigma_V^2
\end{bmatrix}\right)
\]

- Often, parameters “better behaved” if modeled on log scale
Same computational issues as with GLMM’s

No analytic marginal distribution for observations

Same sorts of computational solutions:

- Linearize the model (Pseudolikelihood approaches)
- Approximate the likelihood (Laplace approx. or Gaussian quadrature)
- Bayesian MCMC

ASA webinar on these models and their use in Pharmacokinetic/Pharmacodynamic modeling

# Fit NLME to Theophylline data
# The Theoph object preloaded in R has all sorts of additional data associated with it. Here, I show you how to set up things from a raw data file

theoph <- read.csv('Theoph.csv',as.is=T)

# There are a variety of "Self-starting" pre-defined nonlinear functions.
# they simplify fitting non-linear models
# SSfol() is the one-compartment with clearance model
# uses log scale parameterization of all parameters
# the advantages of a self-start, is that
# 1) you do not need to provide starting values
#    when you use nls(), but you do with nlme()
# 2) they calculate the gradient analytically
ls('package:stats',patt='SS')
# will list all the R self-start functions
# nlme requires the data frame to be a 'groupedData' object. This indicates the groups of
# independent observations
theoph.grp <- groupedData(conc ~ time | Subject,
                           data=theoph)
# you need to indicate Y and the 'primary' X variable
# and most importantly the grouping variable as
# | Subject
# estimate parameters for one subject to
# get an approx. of starting values for the pop.
theoph.1 <- subset(theoph, Subject==1)
subj.1 <- nls(conc~SSfol(Dose, Time, lKe, lKa, lCl),
              data = theoph.1)
Computing for NLME's

\[
\text{theoph.m1} \leftarrow \text{nlme}( \\
\text{conc} \sim \text{SSfol} (\text{Dose}, \text{Time}, \text{lKe}, \text{lKa}, \text{lCl}), \\
data=\text{theoph.grp}, \\
\text{fixed} = \text{lKe} + \text{lKa} + \text{lCl} \sim 1, \\
\text{random} = \text{lKe} + \text{lKa} + \text{lCl} \sim 1, \\
\text{start} = \text{coef(subj.1)} )
\]

# If the parameters differed by (e.g.) sex, you
# would change to fixed = \text{lKe} + \text{lKa} + \text{lCl} \sim \text{sex},
# If the variance/covariance matrix varied by
# sex, use random = \text{lKe} + \text{lKa} + \text{lCl} \sim \text{sex},

\text{summary(\text{theoph.m1})}

# other helper functions are fitted(), predict()
# random.effects(), residuals()
# nlme is extremely powerful. You can also fit
# models for correlation among observations using
# corClasses and model heterogeneity in
# variances (see varClasses and varPower)

# There are also a variety of interesting/useful
# plots for grouped data. See library(help=nlme)
Nonparametric regression using smoothing splines

- Smoothing is fitting a smooth curve to data in a scatterplot
- Will focus on two variable problems: $Y$ and one $X$
- Our model:
  \[ y_i = f(x_i) + \varepsilon_i, \]
  where $\varepsilon_1, \varepsilon_1, \ldots \varepsilon_n$ are independent with mean 0
- $f$ is some unknown smooth function
- Up to now $f$ has a specified form with unknown parameters
  - $f$ could be linear or nonlinear in the parameters,
    - functional form always specified
- If $f$ not determined by the subject matter, we may prefer to let the data suggest a functional form
Why estimate \( f \)?

- can see features of the relationship between \( X \) and \( Y \) that are obscured by error variation
- summarizes the relationship between \( X \) and \( Y \)
- provide a diagnostic for a presumed parametric form

Example: Diabetes data set in Hastie and Tibshirani’s book *Generalized Additive Models*

Examine relationship between age of diagnosis of diabetes and log of the serum C-peptide concentration

Here’s what happens if we fit increasing orders of polynomial, then fit an estimated \( f \)
Penalized spline fit

Aga

log C−peptide concentration

-3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5

5 10 15

Aga

© Dept. Statistics ()
Stat 511 - part 5
Spring 2013 86 / 125
A slightly different way of thinking about Gauss-Markov Linear models:

- If we assume that \( f(x) \) is linear, then \( f(x) = \beta_0 + \beta_1 x \)
- In terms of the Gauss-Markov Linear Model \( y = X\beta + \epsilon \),

\[
X = \begin{bmatrix}
1 & x_1 \\
1 & x_2 \\
\vdots & \vdots \\
1 & x_n \\
\end{bmatrix}
\quad \text{and} \quad
\beta = \begin{bmatrix}
\beta_0 \\
\beta_1 \\
\end{bmatrix}
\]

The linear model approximates \( f(x) \) as a linear combination of two "basis" functions: \( b_0(x) = 1, \ b_1(x) = x \),

\[
f(x) = \beta_0 b_0(x) + \beta_1 b_1(x)
\]
If we assume that f(x) is quadratic, then \( f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 \).

In terms of the Gauss-Markov Linear Model \( y = x\beta + \epsilon \),

\[
X = \begin{bmatrix}
1 & x_1 & x_1^2 \\
1 & x_2 & x_2^2 \\
\vdots & \vdots & \vdots \\
1 & x_n & x_n^2 \\
\end{bmatrix}
\quad \text{and} \quad
\beta = \begin{bmatrix}
\beta_0 \\
\beta_1 \\
\beta_2 \\
\end{bmatrix}
\]

The quadratic model tries to approximate f(x) as a linear combination of three basis functions:

\( b_0(x) = 1, \ b_1(x) = x, \ b_2(x) = x^2 \)

\[
f(x) = \beta_0 b_0(x) + \beta_1 b_1(x) + \beta_2 b_2(x)
\]
Now consider replacing \( b_2(x) = x^2 \) with

\[
S_1(x) = (x - k_1)^+ \equiv \begin{cases} 
0 & \text{if } x \leq k_1 \\
 x - k_1 & \text{if } x > k_1
\end{cases}
\]

where \( k_1 \) is a specified real value.

\( f(x) \) is now approximated by \( \beta_0 b_0(x) + \beta_1 b_1(x) + u_1 S_1(x) \), where \( u_1 \) (like \( \beta_0 \) and \( \beta_1 \)) is an unknown parameter.
Note that $\beta_0 b_0(x) + \beta_1 b_1(x) + u_1 S_1(x) = \beta_0 + \beta_1 X + u_1 (x - k_1)^+$

$$= \begin{cases} 
\beta_0 + \beta_1 x & \text{if } x \leq k_1 \\
\beta_0 + \beta_1 x + u_1 (x - k_1) & \text{if } x > k_1 
\end{cases}$$

$$= \begin{cases} 
\beta_0 + \beta_1 x & \text{if } x \leq k_1 \\
\beta_0 - u_1 k_1 + (\beta_1 + u_1)x & \text{if } x > k_1 
\end{cases}$$

This is clearly a continuous function (because it is a linear combination of continuous functions), and it is piecewise linear.
The function $\beta_0 + \beta_1 x + u_1(x - k_1)^+$ is a simple example of a linear spline function.

The value $k_1$ is known as a knot.

As a Gauss-Markov Linear Model, $y = X\beta + \epsilon$,

$$X = \begin{bmatrix} 1 & x_1 & (x_1 - k_1)^+ \\ 1 & x_2 & (x_2 - k_1)^+ \\ \vdots & \vdots & \vdots \\ 1 & x_n & (x_n - k_1)^+ \end{bmatrix}$$

and $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ u_1 \end{bmatrix}$

We can make our linear spline function more flexible by adding more knots $k_1, \ldots, k_k$ so that $f(x)$ is approximated by $\beta_0 + \beta_1 x + \sum_{j=1}^k u_j s_j(x) = \beta_0 + \beta_1 x + \sum_{j=1}^k u_j(x - k_j)^+$
If we assume \( f(x) = \beta_0 + \beta_1 x + \sum_{j=1}^{k} u_j (x - k_j)^+ \), we can write our model as the Gauss-Markov Linear Model \( y = X\beta + \epsilon \), where

\[
X = \begin{bmatrix}
1 & x_1 & (x_1 - k_1)^+ & (x_1 - k_2)^+ & \ldots & (x_1 - k_k)^+ \\
1 & x_2 & (x_2 - k_1)^+ & (x_2 - k_2)^+ & \ldots & (x_2 - k_k)^+ \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n & (x_n - k_1)^+ & (x_n - k_2)^+ & \ldots & (x_n - k_k)^+
\end{bmatrix}
\]

and \( \beta = (\beta_0, \beta_1, u_1, u_2, \ldots, u_k)' \)

The OLS estimator of \( \beta \) is \( (x'x)^{-1}x'y \).
This is the BLUE of \( \beta \), but this can often result in an estimate of \( f(x) \) that is too “wiggly” or “non-smooth”.

A “wiggly” curve corresponds to values of \( u_1, u_2, \ldots u_k \) far from zero

<table>
<thead>
<tr>
<th>Curve</th>
<th>( \beta_1 )</th>
<th>( u_1 )</th>
<th>( u_2 )</th>
<th>( u_3 )</th>
<th>( \sum u_i^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoother</td>
<td>0.4</td>
<td>0.0</td>
<td>0.4</td>
<td>1.6</td>
<td>2.72</td>
</tr>
<tr>
<td>Wigglier</td>
<td>3.6</td>
<td>-6.4</td>
<td>4.8</td>
<td>-0.8</td>
<td>64.64</td>
</tr>
</tbody>
</table>
If we really believe the true \( f(x) \) is a linear spline function with knots at \( k_1, k_2, \ldots, k_k \), then \( \hat{\beta} = (x'x)^{-1}y \) is the best linear unbiased estimator of \( (\beta_0, \beta_1, u_1, \ldots, u_k)' \).

However, we usually think of our linear spline function as an approximation to the true \( f(x) \).

Prefer a smoother (less flexible) estimate of \( f(x) \).

This has \( u_i \) coefficients closer to 0

Use penalized least squares to estimate a smoother curve.

Find \( \beta = (\beta_0, \beta_1, u_1, \ldots, u_k)' \) that minimizes

\[
(y - x\beta)'(y - x\beta) + \lambda^2 \sum_{j=1}^{k} u_j^2,
\]

where \( \lambda^2 \) is the smoothing parameter, and

\[
\lambda^2 \sum_{j=1}^{k} u_j^2
\]

is the penalty for roughness (lack of smoothness).

Combines two ideas: fit (SSE) and smoothness (penalty for roughness)
Finding the penalized LS estimate of \((\beta_0, \beta_1, u_1, \ldots, u_k)\)'

- If we let \(D = \text{diag}(0,0,1,...,1)\) (k terms), then

\[
(y - x\beta)'(y - x\beta) + \lambda^2 \sum_{j=1}^{k} u_j^2 = (y - x\beta)'(y - x\beta) + \lambda^2 \beta' D \beta
\]

\[
= y'y - 2y'x\beta + \beta' x'x \beta + \lambda^2 \beta' D \beta
\]

\[
= y'y - 2y'x\beta + \beta'(x'x + \lambda^2 D) \beta
\]

- Set derivatives with respect to \(\beta\) equal to 0
- estimating equations: \((x'x + \lambda^2 D)\beta \equiv x'y\)
- solution: \(\hat{\beta}_{\lambda^2} = (x'x + \lambda^2 D)^{-1} x'y\) for any fixed \(\lambda^2 \geq 0\)
- predicted values: \(\hat{y}_{\lambda^2} \equiv x\hat{\beta}_{\lambda^2} = x(x'x + \lambda^2 D)^{-1} x'y\)
You choose $\lambda^2$ and the knots $k_1, \ldots, k_k$.

As $\lambda^2 \to 0$, $\hat{\beta}_{\lambda^2} \to \hat{\beta} = (x'x)^{-1}x'y$.

Small $\lambda^2$ results in non-smooth fit.

As $\lambda^2 \to \infty$, $\hat{\beta}_{\lambda^2} \to \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ 0 \end{bmatrix}$

In the limit, $\lambda^2 \to \infty$ results in the least squares fit.

When $f(x)$ is defined as $f(x) = \beta_0 + \beta_1 x + \sum_{j=1}^{k} u_j (x - k_j)^+$, the resulting function is continuous but the 1st and 2nd derivatives are not.

1st and 2nd derivatives are undefined at the knots.
A smoother smoother

- Next page: fitted penalized regression splines for 3 smoothing parameters: $\sim 0, 100, \text{ and } 5.7$
- $5.7$ is the “optimal” choice, to be discussed shortly
- “optimal” curve is a sequence of straight lines
- continuous, but 1st derivative is not continuous
- Smoothed fits look “smoother” if continuous in 1st derivative and in 2nd derivative
- Suggests joining together cubic pieces with appropriate constraints on the pieces so that the 1st and 2nd derivatives are continuous
- Many very slightly different approaches
  - cubic regression splines (cubic smoothing splines)
  - thin plate splines
We’ll talk about thin plate splines because they provide an easy to implement way to fit multiple $X$’s
$\mathbb{E} y = f(x_1, x_2)$ as well as $\mathbb{E} y = f(x_1) + f(x_2)$

The degree 3 thin plate spline with knots at $(k_1, k_2, \ldots, k_K)$

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \sum_{i=1}^{K} u_k |x - k_i|^5$$
How much to smooth?
  i.e. what $\lambda^2$? or what $u_k$’s
  reminder: $0 \Rightarrow$ no smoothing (linear or quadratic in tps)
  large $\Rightarrow$ close fit to data points

We’ll talk about three approaches:
1. Cross validation
2. Generalized cross validation
3. Mixed models
Cross validation

- General method to estimate “out of sample” prediction error
- Concept: Develop a model, want to assess how well it predicts
- Might use rMSEP $\sqrt{\sum(y_i - \hat{y}_i)^2}$ as a criterion.
- Problem: data used twice, once to develop model and again to assess prediction accuracy
- rMSEP systematically underestimates $\sqrt{\sum(y_i^* - \hat{y}_i^*)^2}$, where $y^*$ are new observations, not used in model development
- Training/test set approach: split data in two parts
  - Training data: used to develop model, usually 50%, 80% or 90% of data set
  - Test set: used to assess prediction accuracy
- Want a large training data set (to get a good model) and a large test set (to get a precise estimate of rMSEP)
Cross validation gets the best of both.

- leave-one-out cv: fit model without obs $i$, use that model to compute $\hat{y}_i$
- 10-fold cv: same idea, blocks of $N/10$ observations

Can be used to choose a smoothing parameter

Find $\lambda^2$ that minimizes cv prediction error

$$CV(\lambda^2) = \sum_{i=1}^{n} \left\{ y_i - \hat{f}_{-i}(x_i; \lambda^2) \right\}^2,$$

where $\hat{f}_{-i}(x_i; \lambda^2)$ is the predicted value of $y_i$ using a penalized linear spline function estimated with smoothing parameter $\lambda^2$ from the data set that excludes the $i^{th}$ observation.

Find $\lambda^2$ value that minimizes $CV(\lambda^2)$. Perhaps compute $CV(\lambda^2)$ for a grid of $\lambda^2$ values

Requires a LOT of computing (each obs, many $\lambda^2$)
Approximation to $CV(\lambda^2)$

$$CV(\lambda^2) \approx \sum_{i=1}^{n} \left\{ \frac{y_i - \hat{f}(x_i; \lambda^2)}{1 - S_{\lambda^2,ii}} \right\}^2$$

where $S_{\lambda^2,ii}$ is the $i^{th}$ diagonal element of the smoother matrix $S_{\lambda^2,ii} = x(x'x + \lambda^2 D)^{-1} x'$.

Remember that $\hat{y} = x(x'x + \lambda^2 D)^{-1} x' y = S_{\lambda^2,ii} y$

OLS: $\hat{y} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \mathbf{X}' y = P_{\mathbf{X}} y$

The smoother matrix $S_{\lambda^2}$ is analogous to the “hat” or projection matrix, $P_{\mathbf{X}}$ in a Gauss-Markov model.
Stat 500: discussed "deleted residuals" \( y_i - \hat{y}_{-i} \), where \( \hat{y}_{-i} \) is the prediction of \( y_i \) when model fit without observation \( i \).

Can compute with refitting the model \( N \) times

\[
y_i - \hat{y}_{-i} = \frac{y_i - \hat{y}_i}{1 - h_{ii}},
\]

where \( h_{ii} \) is the \( i^{th} \) diagonal element of the "hat" matrix

\[
H = P_x = x(x'x)^{-1}x'.
\]

\( h_{ii} \) = "leverage" of observation \( i \)

Thus, the approximation \( CV(\lambda^2) \approx \sum_{i=1}^{n} \left\{ \frac{y_i - \hat{f}(x_i; \lambda^2)}{1 - S_{\lambda^2,ii}} \right\}^2 \) is analogous to the PRESS statistic

\[
\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left( \frac{y_i - \hat{y}_i}{1 - h_{ii}} \right)^2
\]

used in multiple regression.
2. Generalized Cross-Validation (GCV)

- GCV is an approximation to CV obtained as follows:

\[
GCV(\lambda^2) \equiv \sum_{i=1}^{n} \left\{ \frac{y_i - \hat{f}(x_i; \lambda^2)}{1 - \frac{1}{n} \text{trace}(S_{\lambda^2})} \right\}^2
\]

- Since \( \text{trace}(S_{\lambda^2}) = \sum_{i=1}^{n} S_{\lambda^2,ii} \), GCV is \( CV(\lambda^2) \) using the average \( \frac{1}{n} \sum_{i=1}^{n} S_{\lambda^2,ii} \) instead of each specific element.
- Used same way: find \( \lambda^2 \) minimizes \( GCV(\lambda^2) \).
- GCV is not a generalization of CV.
- Originally proposed because faster to compute.
- In some situations, seems to work better than CV, see Wahba, G. (1990). *Spline Models for Observational Data* for details.
- And in very complicated situations, cannot compute \( H \) but can estimate \( \text{trace}(H) \), so can’t use CV but can use GCV.
3. The Linear Mixed Effects Model Approach

- Recall that for our linear spline approach, we assume the model
\[ y_i = \beta_0 + \beta_1 x_i + \sum_{j=1}^{k} u_j (x_i - k_j)^+ + \epsilon_i \] for \( i = 1, \ldots, n \), where \( \epsilon_1, \ldots, \epsilon_n \) i.i.d. \( N(0, \sigma^2) \)

- Suppose we add the following assumptions: \( u_1, \ldots, u_k \) i.i.d. \( N(0, \sigma^2_u) \) independent of \( \epsilon_1, \ldots, \epsilon_n \) i.i.d. \( N(0, \sigma^2_e) \). \( \sigma^2_u \equiv \sigma^2 \)

- Then we may write our model as \( y = X\beta + Zu + \epsilon \), where

\[
X = \begin{bmatrix}
1 & x_1 \\
1 & x_2 \\
\vdots & \vdots \\
1 & x_n \\
\end{bmatrix}, \quad \beta = \begin{bmatrix}
\beta_0 \\
\beta_1 \\
\end{bmatrix}, \quad Z = \begin{bmatrix}
(x_1 - k_1)^+ & \ldots & (x_1 - k_k)^+ \\
(x_2 - k_1)^+ & \ldots & (x_2 - k_k)^+ \\
\vdots & \ddots & \vdots \\
(x_n - k_1)^+ & \ldots & (x_n - k_k)^+ \\
\end{bmatrix}.
\]
This is a linear mixed effects model!
It can be shown that the BLUP of $X\beta + Zu$ is equal to
$$w(w'w + \frac{\sigma_e^2}{\sigma_u^2}D)^{-1}w'y$$
where $w = [x, z]$.

Thus, the BLUP of $X\beta + Zu$ is equal to
$$S_{\frac{\sigma_e^2}{\sigma_u^2}}y = \text{(Fitted values of linear spline smoother for } \lambda^2 = \frac{\sigma_e^2}{\sigma_u^2})$$

Thus, we can use either ML or REML to estimate $\sigma_u^2$ and $\sigma_e^2$.
(Denote estimates by $\hat{\sigma}_u^2$ and $\hat{\sigma}_e^2$.)

Then we can estimate $\beta$ by
$$\hat{\beta}_{\hat{\Sigma}} = (x'\hat{\Sigma}^{-1}x)^{-1}x'\hat{\Sigma}y$$
and predict $u$ by
$$\hat{u}_{\hat{\Sigma}} = \hat{G}Z'\hat{\Sigma}^{-1}(y - x\hat{\beta}_{\hat{\Sigma}}) = \hat{\sigma}_u^2Z'\hat{\Sigma}^{-1}(y - x\hat{\beta}_{\hat{\Sigma}})$$
where
$$\hat{\Sigma} = \hat{\sigma}_u^2ZZ' + \hat{\sigma}_e^2I$$

The resulting coefficients $\begin{bmatrix} \hat{\beta}_{\hat{\Sigma}} \\ \hat{u}_{\hat{\Sigma}} \end{bmatrix}$ will be equal to the estimate obtained using penalized least squares with smoothing parameter
$$\lambda^2 = \frac{\hat{\sigma}_e^2}{\hat{\sigma}_u^2}$$
A scatter plot showing the relationship between log C-peptide concentration and age of diagnosis. The plot includes data points and fitted curves for Linear spline and Thin Plate spine models.
Still need to choose number of knots \((k)\) and their locations \(k_1, \ldots, k_k\).

Ruppert, Wand and Carroll (2003) recommend 20-40 knots maximum, located so that there are roughly 4-5 unique \(x\) values between each pair of knots.

Most software automatically chooses knots using a strategy consistent (roughly) with this recommendation.

Knot choice is not usually as important as choice of smoothing parameter.

- As long as there are enough knots, a good fit can usually be obtained.
- Penalization prevents a fit that is too rough even when there are many knots.
Towards inference with a penalized spline

- If we want a confidence or prediction interval around the predicted line, need to know df for error.
- If we want to compare models (e.g. $Ey = \beta_0 + \beta_1 x$ vs $Ey = f(x)$), need to know df for penalized spline fit.
- Can do this test because
  - $Ey = \beta_0 + \beta_1 x$ is nested in $Ey = f(x)$ fit as a linear spline.
  - $Ey = \beta_0 + \beta_1 x + \beta_2 x^2$ is nested in $Ey = f(x)$ fit as a thin plate spline.
- If we use a penalized linear spline, how many parameters are we using to estimate the mean function?
  - It may seem like we have $k+2$ parameters $\beta_0, \beta_1, u_1, u_2, ..., u_k$. 
However, \( u_1, u_2, \ldots, u_k \) are not completely free parameters because of penalization.

The effective number of parameters is lower than \( k+2 \) and depends on the value of the smoothing parameter \( \lambda^2 \).

Recall that our estimates of \( \beta_0, \beta_1, u_1, u_2, \ldots, u_k \) minimize

\[
\sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i - \sum_{j=1}^{k} u_j (x_i - k_j)^+)^2 + \lambda^2 \sum_{j=1}^{k} u_j^2
\]

A larger \( \lambda^2 \) means less freedom to choose values for \( u_1, \ldots, u_k \) for from 0.

Thus, the number of effective parameters should decrease as \( \lambda^2 \) increases.

In the Gauss-Markov framework with no penalization, the number of free parameters used to estimate the mean of \( y(x|\beta) \) is

\[
\text{rank}(x) = \text{rank}(P_x) = \text{trace}(P_x)
\]
For a smoother, the smoother matrix $S$ plays the role of $P_x$.

For penalized linear splines, the smoother matrix is

$$S_{\lambda^2} = x(x'x + \lambda^2 D)^{-1} x'$$

where

$$X = \begin{bmatrix}
1 & x_1 & (x_1 - k_1)^+ & \ldots & (x_1 - k_k)^+ \\
1 & x_2 & (x_2 - k_1)^+ & \ldots & (x_2 - k_k)^+ \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n & (x_n - k_1)^+ & \ldots & (x_n - k_k)^+
\end{bmatrix}$$

$$D = \begin{bmatrix} 2 \times 2 \quad 0 \\ 0 \quad k \times k \end{bmatrix}$$

Thus, we define the effective number of parameter (or the degrees of freedom) used when estimating $f(x)$ to be

$$tr(S_{\lambda^2}) = tr[x(x'x + \lambda^2 D)^{-1} x'] = tr[(x'x + \lambda^2 D)^{-1} x'x]$$
df model = 2, df error = 41

log C−peptide concentration vs. Age of diagnosis

df model = 2, df error = 41
df model = 10, df error = 33

![Graph showing relationship between age of diagnosis and log C-peptide concentration with a model of df = 10 and df error = 33.](image)
df model = 3.59, df error = 38.76
Recall that our basic model is $y_i = f(x_i) + \epsilon_i$ \hspace{1pt} $(i = 1, \ldots, n)$ where $\epsilon_1, \ldots, \epsilon_n \overset{i.i.d.}{\sim} (0, \sigma^2)$.

How should we estimate $\sigma^2$?

A natural estimator would be $MSE \equiv \frac{\sum_{i=1}^{n} \{y_i - \hat{f}(x_i; \lambda^2)\}^2}{df_{ERROR}}$

$df_{ERROR}$ is usually defined to be $n - 2tr(S_{\lambda^2}) + tr(S_{\lambda^2}S'_{\lambda^2})$.

To see where this comes from, recall that for $w$ random and $A$ fixed $E(w'Aw) = E(w)'A\text{E}(w) + tr(A\text{Var}(w))$

Let $f = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix}$ and $\hat{f}_{\lambda^2} = \begin{bmatrix} \hat{f}(x_1; \lambda^2) \\ \hat{f}(x_2; \lambda^2) \\ \vdots \\ \hat{f}(x_n; \lambda^2) \end{bmatrix} = S_{\lambda^2}y$
Then, $E[\sum_{i=1}^{n} \left\{ y_i - \hat{f}(x_i; \lambda^2) \right\}^2 ]$

$$= E[(y - \hat{f})'(y - \hat{f})]$$
$$= E[\|y - \hat{f}\|^2] = E[\|(I - S_{\lambda^2})y\|^2]$$
$$= E[y'(I - S_{\lambda^2})'(I - S_{\lambda^2})y]$$
$$= f'(I - S_{\lambda^2})'(I - S_{\lambda^2})f + tr([(I - S_{\lambda^2})'(I - S_{\lambda^2})\sigma^2 I]$$
$$= \|f - S_{\lambda^2}f\|^2 + \sigma^2[tr(I) - 2tr(S_{\lambda^2}) + tr(S'_{\lambda^2}S_{\lambda^2})]$$
$$\approx \sigma^2[n - 2tr(S_{\lambda^2}) + tr(S'_{\lambda^2}S_{\lambda^2})]$$

Thus, if we define
$$df_{ERROR} = n - 2tr(S_{\lambda^2}) + tr(S'_{\lambda^2}S_{\lambda^2}), E(MSE) \approx \sigma^2$$
The Standard Error of \( \hat{f}(x; \sigma^2) \):

\[
\hat{f}(x; \lambda^2) = \hat{\beta}_0 + \hat{\beta}_1 x + \sum_{j=1}^{k} \hat{u}_j (x - k_j)^+
\]

\[
= [1, x, (x - k_1)^+, ..., (x - k_k)^+] \begin{bmatrix}
\hat{\beta}_0 \\
\hat{\beta}_1 \\
\hat{u}_1 \\
\vdots \\
\hat{u}_k
\end{bmatrix}
\]

\[
= [1, x, (x - k_1)^+, ..., (x - k_k)^+] (x' x + \lambda^2 D)^{-1} x' y = C' y
\]

If \( \lambda^2 \) and the knots, \( k_i \), are fixed and not chosen as a function of the data, \( C \) is just a fixed (nonrandom) vector.
Thus, $\text{Var}[\hat{f}(x; \lambda^2)] = \text{Var}(C'y) = C'\sigma^2 I C = \sigma^2 C'C$

It follows that the standard error for $\hat{f}(x; \lambda^2)$ is

$SE[\hat{f}(x; \lambda^2)] = \sqrt{MSE \; C'C}$

If $\lambda^2$ and/or the knots are selected based on the data (as is usually the case), $\sqrt{MSE \; C'C}$ is still used as an approximate standard error.

However, that approximate standard error may be smaller than it should be because it does not account for variation in the $C$ vector itself.

Ruppert, Wand, and Carroll (2003) suggest other strategies that use the linear mixed effects model framework.

Calculate pointwise $1 - \alpha$ confidence intervals for $\hat{f}(x_i)$ by

$t_{1-\alpha/2,dfe} \sqrt{\text{Var}[\hat{f}(x; \lambda^2)]},$

where $dfe$ is the $df_{ERROR}$ defined a few pages ago.
Extensions of penalized splines

- More than one \( X \) variable
  - Can fit either as a thin plate spline, \( f(X_1, X_2) \)
  - or as additive effects: \( f_1(X_1) + f_2(X_2) \)
  - Can combine parametric and nonparametric forms:
    \[
    \beta_0 + \beta_1 X_1 + f(X_2)
    \]

- Additive effects models sometimes called Generalized Additive Models (GAM’s)

- Penalized splines provide a model for \( E_y \)

- Our discussion has only considered \( y_i \sim N(Ey_i, \sigma^2) \)

- Can combine with GLM ideas, e.g.:
  \[
  y_i \sim Poisson(f(x_i)) \text{ or } Binomial(f(x_i))
  \]
Computing splines

This is a compressed version of diabetes.r. The version on the class web site has more extensive comments.

# these can be fit by at least three packages:
#    gam() in mgcv, spm() in SemiPar, and fda

# I’ve used gam() before. spm() has some peculiarities
# Previous instructors of 511 used spm()
# the results are slightly different and I haven’t had time to track down why.
# To replicate lecture results, this code demonstrates

library(SemiPar)
Computing splines

diabetes <- read.csv('diabetes.csv')

plot(diabetes$age,diabetes$y, pch=19,col=4, 
     xlab='Age at diagnosis', ylab='log C-peptide concentration')

# a couple of peculiarities
# 1) formula interface to spm() does not 
# accept data= argument.
# 2) to use predict.spm(), cannot use diabetes$age.
# I use attach to avoid problems.

# basic call to spm
attach(diabetes)
diab.spm <- spm(y ~ f(age));
Computing splines

plot(diab.spm)
#   Bands are pointwise 95% ci's for f hat(x)

diab.pred <- predict(diab.spm,
    newdata=data.frame(age=seq(1,16,0.5)))
lines(seq(1,16,0.5), diab.pred,lwd=2)

# default is normal d’n.  can use binomial
#  or Poisson, by specifying family=binomial
#  or family=poisson

# warning: remember to specify f() to get a smooth
temp <- spm(diabetes$y ˜ age)
# gives you the linear regression fit
# is useful for more than one X, some of which are to
# others by a smooth.
# a third pecularity: lots of useful values have be extracted by hand
print(diab.spm)     # not very informative
summary(diab.spm)   # a little better

# info on where to find various potentially
#    useful numbers is in the version on the web site
# also how to change the basis functions, amount
#    of smoothing, and est. derivatives