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: $\boldsymbol{Y} = \boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u} + \epsilon$
- Consider each level of variation separately.
  A hierarchical or multi-level model

$$\begin{aligned}
\boldsymbol{\eta} &= \boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u} \\
&\sim N(\boldsymbol{X}\beta,\ \boldsymbol{ZGZ}') \\
\boldsymbol{Y}|\boldsymbol{\eta} &= \boldsymbol{\eta} + \epsilon \\
&\sim N(\eta,\ \boldsymbol{R}) \\
\boldsymbol{Y}|\boldsymbol{u} &= \boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u} + \epsilon \\
&\sim N(\boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u},\ \boldsymbol{R})
\end{aligned}$$

- Above specifies the conditional distribution of $\boldsymbol{Y}$ given $\boldsymbol{\eta}$ or equivalently $\boldsymbol{u}$

- To write down a likelihood, need the marginal pdf of $\boldsymbol{Y}$

$$\begin{aligned} f(\boldsymbol{Y}, \boldsymbol{u}) &= f(\boldsymbol{Y}|\boldsymbol{u}) f(\boldsymbol{u}) \\ f(\boldsymbol{Y}) &= \int_{\boldsymbol{u}} f(\boldsymbol{Y}, \boldsymbol{u}) d\boldsymbol{u} \\ &= \int_{\boldsymbol{u}} f(\boldsymbol{Y}|\boldsymbol{u}) f(\boldsymbol{u}) d\boldsymbol{u} \end{aligned}$$

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

$$\boldsymbol{Y} \sim N(\boldsymbol{X}\beta, \ \boldsymbol{ZGZ}' + \boldsymbol{R})$$

- Extend to GLMs by changing conditional distribution of $\boldsymbol{Y}|\boldsymbol{u}$
  - Logistic: $f(Y_i|\boldsymbol{u}) \sim Binomial(m_i, \pi_i(\boldsymbol{u}))$
  - Poisson: $f(Y_i|\boldsymbol{u}) \sim Poisson(\lambda_i(\boldsymbol{u}))$

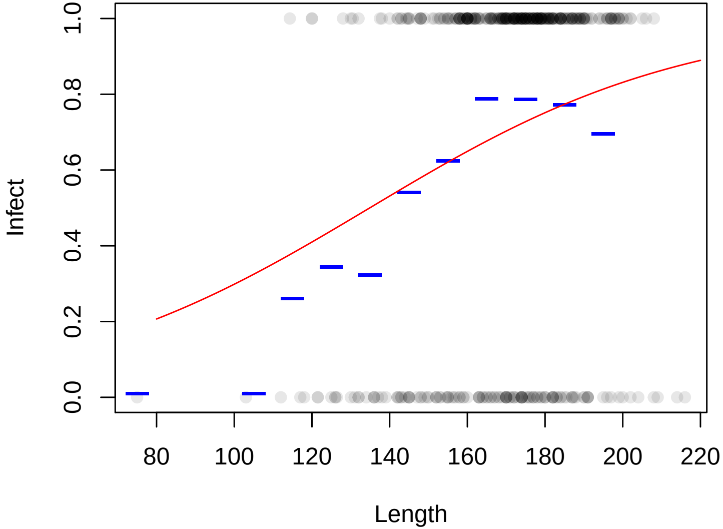
- **Big problem**: Usually no analytic solutions to $f(\boldsymbol{Y})$
  No closed form solution to the integral
- Some exceptions:
  - $\boldsymbol{Y}|\boldsymbol{\eta} \sim \textit{Binomial}(\boldsymbol{m}, \boldsymbol{\eta})$, $\boldsymbol{\eta} \sim \beta(\alpha, \beta)$
    $\boldsymbol{Y} \sim \textit{BetaBinomial}$
  - $\boldsymbol{Y}|\eta \sim \textit{Poisson}(\eta)$, $\boldsymbol{\eta} \sim \Gamma(\alpha, \beta)$
    $\boldsymbol{Y} \sim \textit{NegativeBinomial}$
- 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:

$$\begin{aligned}\boldsymbol{\mu} &= g^{-1}(\boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u}), \;\; \boldsymbol{u} \sim N(\mathbf{0}, \boldsymbol{G}) \\ \boldsymbol{Y}|\boldsymbol{\mu} &= f(\boldsymbol{\mu})\end{aligned}$$

- 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[infect]
    2. predict P[infect] for a deer of a specified length
- Consider the model $i \in \{1, 2, \ldots, 447\}$ indexes deer

$$\begin{aligned} Y_i &\sim \mathit{Bernoulii}(\pi_i) \\ \text{logit}\, \pi_i &= \mu + \beta\, l_i, \end{aligned}$$

where Y_i is infection status (0/1) and l_i is the length of the deer

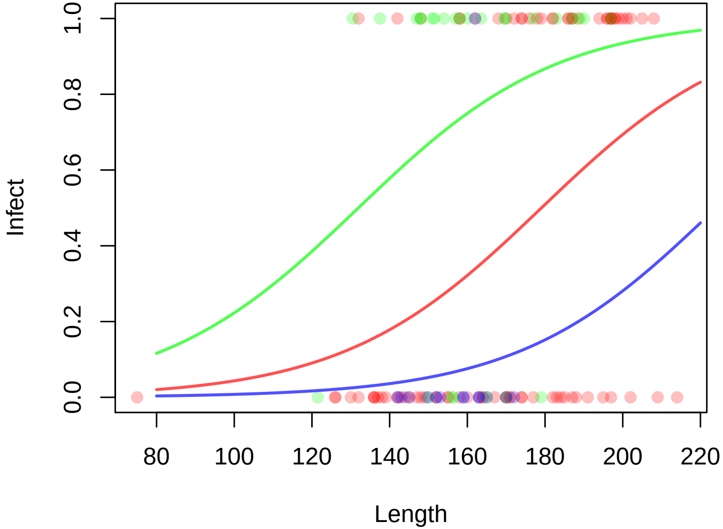

Infect
0.0
0.2
0.4
0.6
0.8
1.0
80
100
120
140
160
180
200
220
Length

- 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:

$$\begin{aligned} Y_{ij} &\sim Bernoulii(\pi_{ij}) \\ \text{logit}\, \pi_{ij} &= \mu + \alpha_i + \beta\, l_{ij} \end{aligned}$$

-

Term	Deviance	Δ Dev.	df	p value
NULL	549.2			
Farm	394.25	155.05	21	< 0.0001
Length	363.53	30.72	1	< 0.0001


Infect
0.0
0.2
0.4
0.6
0.8
1.0
80
100
120
140
160
180
200
220
Length

- $\hat{\beta}$ 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[infect| length] for these 24 farms
- You need to know the Farm effect to estimate P[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

$$\begin{aligned} Y_{ij} &\sim Bernoulii(\pi_{ij}) \\ \text{logit}\, \pi_{ij} &= \mu + \alpha_i + \beta\, l_{ij} \\ \alpha_i &\sim N(0, \sigma^2_F) \end{aligned}$$

- 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
    3. Generalized Estimating Equations
- 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 $g^{-1}(\boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u})$
        2. Pseudolikelihood: similar
    - Inference about $\boldsymbol{b}$ conditional on $\boldsymbol{\Sigma}$

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 $\boldsymbol{\Sigma}$

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 Var $\hat{\boldsymbol{b}}$:
- Remember Var $\hat{\beta}$ when $\boldsymbol{\Sigma}$ misspecified:

$$\begin{aligned}\mathrm{Var}\;\hat{\boldsymbol{b}} &= (\boldsymbol{X}'\boldsymbol{X})^{-}\boldsymbol{X}'\boldsymbol{\Sigma}\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-} \\ &= \frac{\mathrm{Var_W}\,\hat{\boldsymbol{b}}}{\sigma^2}\boldsymbol{X}'\boldsymbol{\Sigma}\boldsymbol{X}\frac{\mathrm{Var_W}\,\hat{\boldsymbol{b}}}{\sigma^2}\end{aligned}$$

- Imagine there is an estimate of $\boldsymbol{\Sigma}$, call it C,
  usually computed from replicate data
- Use the mis-specified variance estimator to patch-up Var $\hat{\beta}$:

$$\mathrm{Var}\;\hat{\boldsymbol{b}} = \frac{\mathrm{Var_W}\;\hat{\boldsymbol{b}}}{\sigma^2}\boldsymbol{X}'\boldsymbol{C}\boldsymbol{X}\frac{\mathrm{Var_W}\;\hat{\boldsymbol{b}}}{\sigma^2}$$

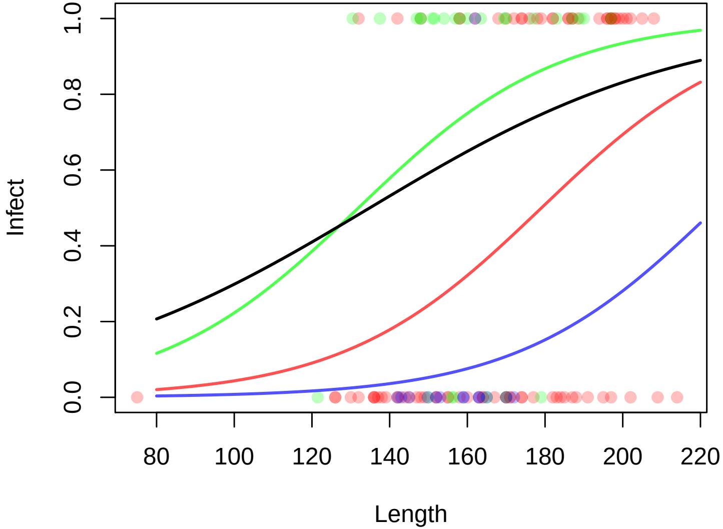
- Sometimes called the Sandwich estimator (bread, filling, bread)
- Same idea, but many more details and different equations for GLMM

Marginal or conditional inference

- There is a major, important difference between the model fit by GEE and the model fit by GLMM

$$\begin{aligned} \text{GLMM} \quad \mathrm{E}\boldsymbol{Y}|\boldsymbol{u} = &\quad g^{-1}(\boldsymbol{X}\beta + \boldsymbol{Z}\boldsymbol{u}) \qquad (1) \\ \text{GEE} \quad \mathrm{E}\boldsymbol{Y} = &\quad g^{-1}(\boldsymbol{X}\beta) \qquad (2) \end{aligned}$$

- (1) models the conditional mean of Y given the random effects
  Influence of length 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, $g^{-}(x) = x$, usually used with normal distributions
- Not the same for other link functions (logit, log)

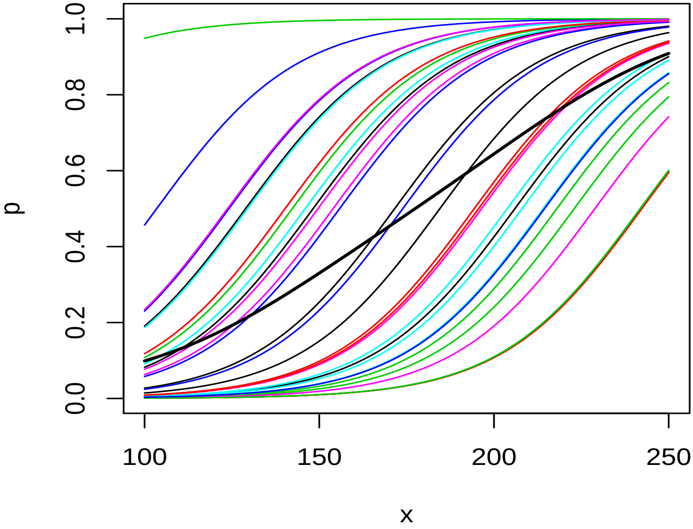

Infect
0.0
0.2
0.4
0.6
0.8
1.0
80
100
120
140
160
180
200
220
Length

- Results from various estimation methods

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

- 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] × population size
     want marginal = population averaged estimate of log odds


p
0.0
0.2
0.4
0.6
0.8
1.0
100
150
200
250
x

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

```
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
```

Computing for GEE’s

```
#  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 $E(\boldsymbol{y}) = X\beta$
- $f(\boldsymbol{X}_i, \beta) = \boldsymbol{X}_i'\beta$ is said to be linear in the parameters of β because $\boldsymbol{X}_i'\boldsymbol{\beta} = X_{i1}\boldsymbol{\beta}_1 + X_{i2}\boldsymbol{\beta}_2 + \ldots + X_{ip}\boldsymbol{\beta}_p$ is a linear combination of $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, ..., \boldsymbol{\beta}_p$.
- $f(\boldsymbol{X}_i, \beta) = \boldsymbol{X}_i'\beta$ is linear in β even if the predictor variables, the $\boldsymbol{X}'s$ are nonlinear functions of other variables.

- For example, if
  $X_{i1} = 1$
  $X_{i2} =$ Amount of fertilizer applied to plot i
  $X_{i3} =$ (Amount of fetrtilizer applied to plot i)^2
  $X_{i4} =$ log(Concentration of fungicide on plot i)
- $f(\boldsymbol{X}_i, \beta) = \boldsymbol{X}_i' \beta = X_{i1}\beta_1 + X_{i2}\beta_2 + X_{i3}\beta_3 + X_{i4}\beta_4$
  $= \beta_1 + \text{fert}_i\beta_2 + \text{fert}_i^2\beta_3 + \log((fung)_i)\beta_4$ 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 $f(\boldsymbol{X}_i, \beta)$ cannot be written as a linear combination of $\beta_1, \beta_2, .., \beta_p$
- 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

$$v_s = \frac{v_m S}{S + K_m}$$

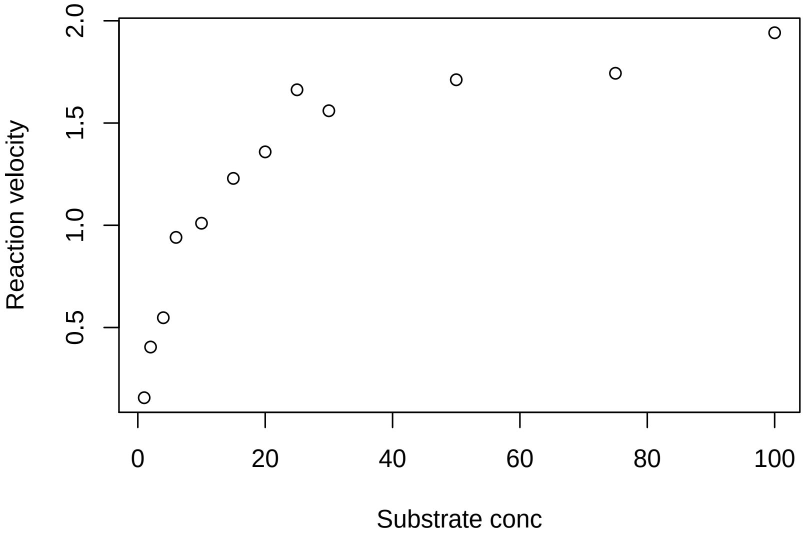
- S is concentration of substrate, v_s is reaction rate at S
  v_m is maximum reaction rate,
  K_m is enzyme affinity= S at which $v_s = v_m/2$
- Function is mathematically equivalent to:
  - Lineweaver-Burke:

    $$\frac{1}{v_s} = \frac{1}{v_m} + \frac{K_m}{v_m}\frac{1}{S}$$

    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


Reaction velocity
0.5
1.0
1.5
2.0
0
20
40
60
80
100
Substrate conc

- However, the estimators of v_m and K_m derived from each model are not the same
- Illustrate numerically: LS estimates from each model

Model	$\hat{\beta}_0$	$\hat{\beta}_1$	v_m	$\hat{v}_m$	K_m	$\hat{K}_m$
nonlin				2.05		9.12
L-B	0.377	5.64	$1/\beta_0$	2.65	β_1/β_0	14.96
H-W:	4.74	0.482	$1/\beta_1$	2.07	β_0/β_1	9.83

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

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

- And

$$v_i = \frac{v_m S_i}{S_i + K_m} + \varepsilon_i, \qquad \varepsilon_i \sim (0, \sigma_1^2) \tag{3}$$

- is not the same as

$$\frac{1}{v_1} = \frac{1}{v_m} + \frac{K_m}{v_m}\frac{1}{S_i} + \epsilon_i, \qquad \epsilon_i \sim (0, \sigma_2^2) \tag{4}$$

- 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:

$$Y_i = \beta_0 e^{\beta_1 T_i} + \varepsilon_i, \qquad \varepsilon_i \sim (0, \sigma_1^2)$$

- Linearized form, constant variance, normal dist.:

$$Y_i^* = \log Y_i = \log \beta_0 + rT_i + \epsilon_i, \qquad \epsilon_i \sim N(0, \sigma_2^2)$$

- Statistically equivalent to

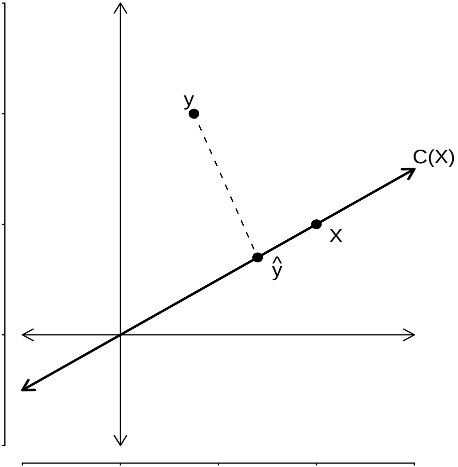
$$Y_i = \beta_0 e^{\beta_1 T_i} \times e^{\epsilon_i}, \qquad \epsilon_i \sim N(0, \sigma_2^2)$$

- i.e., errors are multiplicative log normal with constant lognormal variance

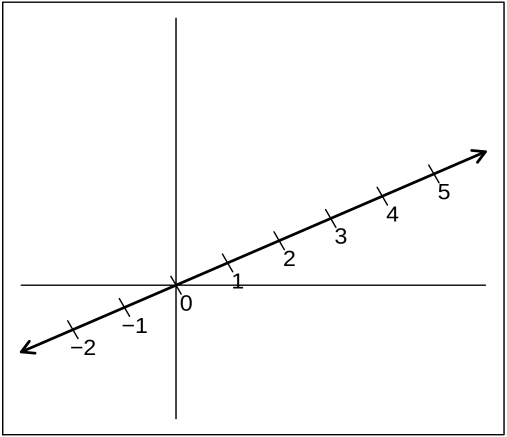
- Three classes of models:
  1. Linear
  2. Transformable to linear, e.g. MM or exp. growth
  3. Intrinsically nonlinear, e.g. $Y(t) = N_1 e^{-r_1 T} + N_2 e^{-r_2 T}$
- Why would we want to consider a nonlinear model?
- Pinheiro and Bates (2000) give some reasons:
  1. mechanistic - based on theoretical considerations about the mechanism producing the response.
  2. often interpretable and parsimonions
  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 β values to $\boldsymbol{C}(X)$



- 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.

$$Y_i = 60 + 70e^{-rX_i}$$

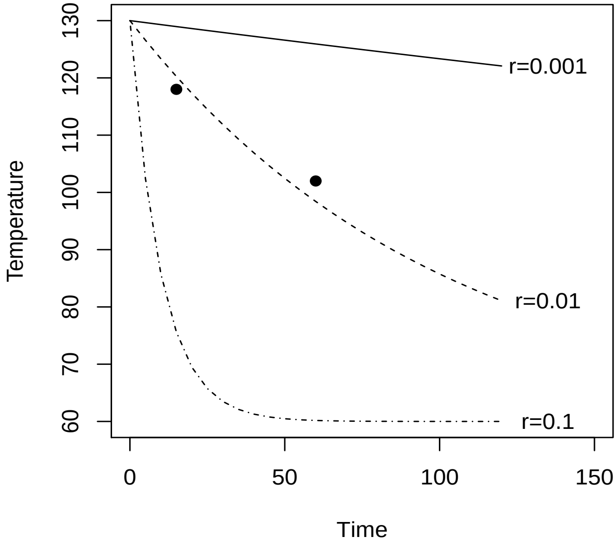
- Assume errors in temperature measurement have constant variance

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

- 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

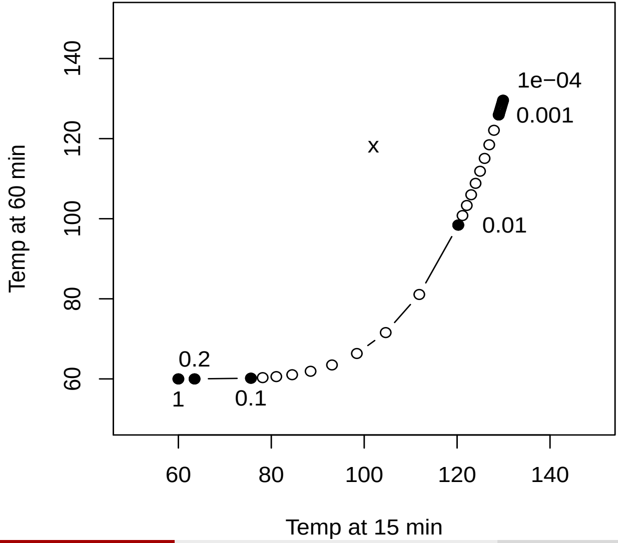
$$L(r) = (Y_i - \hat{Y}(r)_i)^2 = \left(Y_i - (60 + 70e^{-rX_i})\right)^2$$

- Consider fitting this model to two data points: (15,118), (60, 102)


Temperature
130
120
110
100
90
80
70
60
0
50
100
150
Time
r=0.001
r=0.01
r=0.1

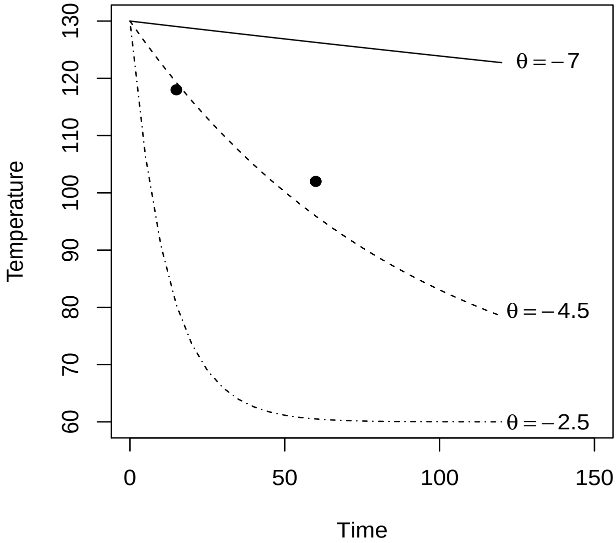
The expectation surface, $(\hat{Y}(r)_{X=15}, \hat{Y}(r)_{X=60})$,

1. Expectation surface is curved
2. Points not equally spaced (considered as function of r)



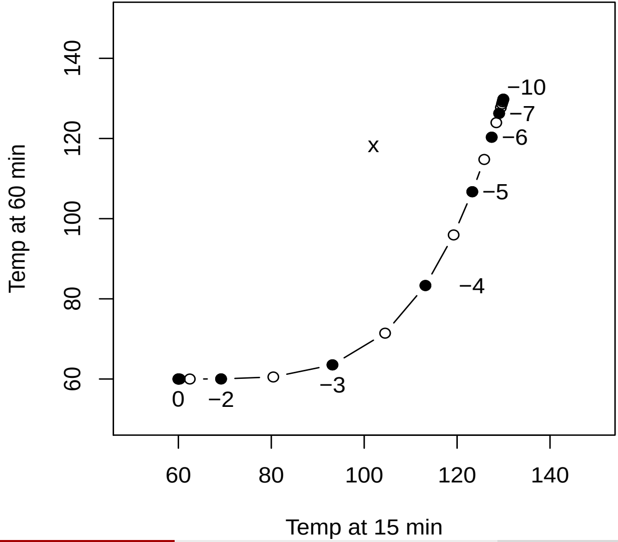
- Can write the same nonlinear model using different parameters.

$$Y_i = 60 + 70e^{-e^{\theta} X_i} + \epsilon_i,\ \epsilon_i \sim (0, \sigma^2), \qquad \theta = \log r$$



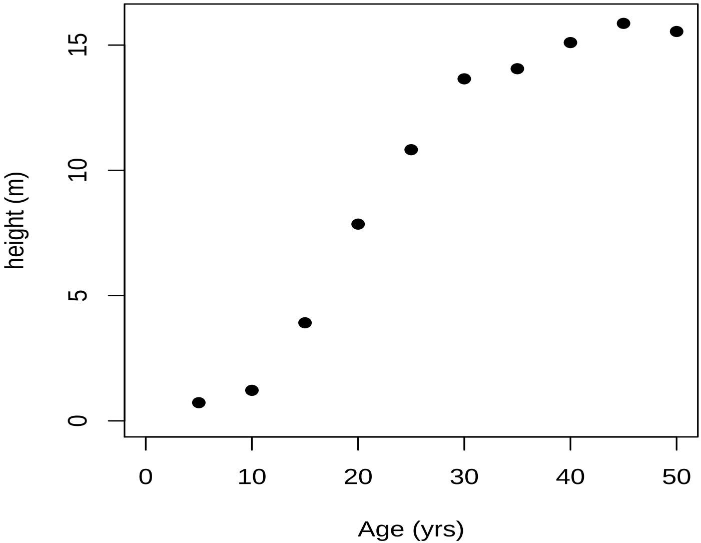
The expectation surface, $(\hat{Y}(\theta)_{X=15}, \hat{Y}(\theta)_{X=60})$,

1. Expectation surface is same manifold
2. But spacing of points not the same (more evenly spaced for θ)



- $\hat{Y}$ 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 $X_i (i = 1, ..., n)$

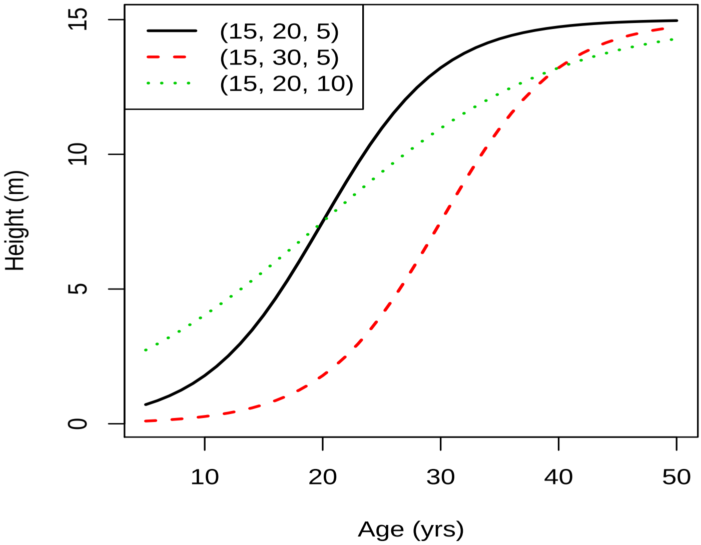


- Want a model in which:
  - trees grow slowly, then quickly, then slowly
  - trees have constant final height
  - the final height needs to be estimated
- One (of many) asymptotic growth models is the 3 parameter logistic

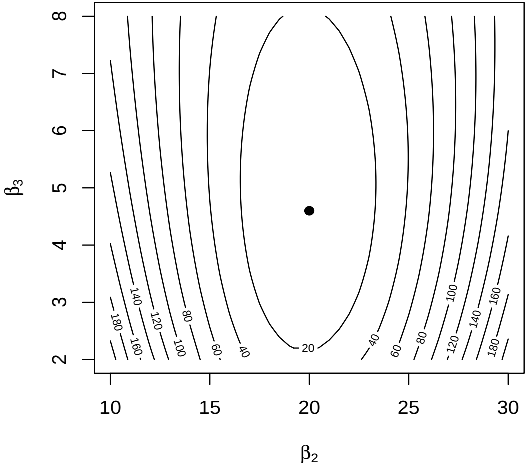
$$E(y_i) = f(X_i, \boldsymbol{\beta}) = \frac{\beta_1}{1 + e^{-(X_i - \beta_2)/\beta_3}}$$

- Interpretation of parameters:
  - β_1 is final height
  - β_2 is age at which height is $\beta_1/2$
  - β_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:

$$y_i = f(X_i, \boldsymbol{\beta}) + \epsilon_i,\ E(\epsilon_i) = 0,\ Var(\epsilon_i) = \sigma^2,\ i = 1, ..., n$$


(15, 20, 5)
(15, 30, 5)
(15, 20, 10)
Height (m)
0
5
10
15
10
20
30
40
50
Age (yrs)

- Least Squares Estimation $y_i = f(\boldsymbol{X}_i, \beta) + \epsilon_i \qquad i = 1, ..., n$
- Find $\hat{\beta}$ that minimizes $g(\boldsymbol{b}) = \sum_{i=1}^{n} [y_i - f(\boldsymbol{X}_i, \boldsymbol{b})]^2$



- Candidate $\hat{\beta}$ is the solution to the estimating equations:

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

- These are:

$$\begin{aligned}
\frac{\partial g(\boldsymbol{b})}{\partial \boldsymbol{b_1}} &= 2\sum_{i=1}^{n}[y_i - f(\boldsymbol{X}_i, \boldsymbol{b})]\frac{\partial f(\boldsymbol{X}_i, \boldsymbol{b})}{\partial b_1} \\
&\vdots \\
\frac{\partial g(\boldsymbol{b})}{\partial \boldsymbol{b_p}} &= 2\sum_{i=1}^{n}[y_i - f(\boldsymbol{X}_i, \boldsymbol{b})]\frac{\partial f(\boldsymbol{X}_i, \boldsymbol{b})}{\partial b_p}
\end{aligned}$$

- Can write as a matrix equation
- Define $f(\boldsymbol{X}, \boldsymbol{b}) = \begin{bmatrix} f(\boldsymbol{X}_1, \boldsymbol{b}) \\ \vdots \\ f(\boldsymbol{X}_n, \boldsymbol{b}) \end{bmatrix}$
- And

$$D' = \begin{bmatrix} \frac{\partial f(\boldsymbol{X}_1, \boldsymbol{b})}{\partial b_1} & \cdots & \frac{\partial f(\boldsymbol{X}_n, \boldsymbol{b})}{\partial b_1} \\ \vdots & & \vdots \\ \frac{\partial f(\boldsymbol{X}_1, \boldsymbol{b})}{\partial b_p} & \cdots & \frac{\partial f(\boldsymbol{X}_n, \boldsymbol{b})}{\partial b_p} \end{bmatrix}$$

  Then $\frac{\partial g(\boldsymbol{b})}{\partial \boldsymbol{b}} = \mathbf{0}$ is equivalent to $D'[\boldsymbol{y} - f(X, \boldsymbol{b})] = \mathbf{0}$
- In the linear case, $D' = X'$ and $D'[\boldsymbol{y} - f(X, \boldsymbol{b})] = \mathbf{0}$ becomes $X'[\boldsymbol{y} - X\boldsymbol{b}] = \mathbf{0} \Rightarrow X'X\boldsymbol{b} = X'\boldsymbol{y}$

- In the nonlinear case, D' depends on β so that the equation $D'[\boldsymbol{y} - f(X, \boldsymbol{b})] = \mathbf{0}$ has (usually) no analytic solution for $\boldsymbol{b}$
- For example, for the logistic model,

$$
\begin{aligned}
\frac{\partial f(\boldsymbol{X}_i, \beta)}{\partial \beta_1} &= \frac{1}{1 + \exp\{-(X_i - \beta_2)/\beta_3\}} \\
\frac{\partial f(\boldsymbol{X}_i, \beta)}{\partial \beta_2} &= \frac{-\beta_1 \exp\{-(X_i - \beta_2)/\beta_3\}}{[1 + \exp\{-(X_i - \beta_2)/\beta_3\}]^2 \beta_3} \\
\frac{\partial f(\boldsymbol{X}_i, \beta)}{\partial \beta_3} &= \frac{-\beta_1 \exp\{-(X_i - \beta_2)/\beta_3\}(X_i - \beta_2)}{[1 + \exp\{-(X_i - \beta_2)/\beta_3\}]^2 \beta_3^2}
\end{aligned}
$$

- Various algorithms to find minimum analytically
- Very common one for nonlinear regression is the Gauss-Newton algorithm
- Taylor's theorem:

$$f(\boldsymbol{x}_i, \boldsymbol{b}) \approx f(\boldsymbol{x}_i, \boldsymbol{b}^*) + \left[\frac{\partial f(\boldsymbol{x}_i, \boldsymbol{b})}{\partial \boldsymbol{b}} |_{\boldsymbol{b}=\boldsymbol{b}^*} \right] (\boldsymbol{b} - \boldsymbol{b}^*)$$

- So E $\boldsymbol{Y} = f(\boldsymbol{X}, \beta)$ can be approximated by

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

  where $\hat{D}$ is D evaluated at $\boldsymbol{b} = \boldsymbol{b}^*$.
- Notice this is a linear regression where $\boldsymbol{X}$ is $\hat{D}$

$$\begin{aligned} f(X, \boldsymbol{b}) &\approx f(X, \boldsymbol{b}^*) - \hat{D}\boldsymbol{b}^* + \hat{D}\boldsymbol{b} \\ &\approx \text{constant} + \hat{D}\boldsymbol{b} \end{aligned}$$

- Gauss-Newton algorithm
  1. Choose a starting value, $\boldsymbol{b}_0$
  2. Calculate D for $\boldsymbol{b} = \boldsymbol{b}_0$
  3. Estimate $\hat{\boldsymbol{b}}$ using approximating linear model
  4. Call this $\boldsymbol{b}_1$
  5. Calculate D for $\boldsymbol{b} = \boldsymbol{b}_1$
  6. Repeat steps 3-5 until convergence
- Various ways to define convergence
  1. Little to no change in $\boldsymbol{b}$ after an iteration
     “not making progress” convergence
  2. Little to no change in SSE, $g(\boldsymbol{b})$, after an iteration
     “not making progress” convergence
  3. $\frac{\partial g(\boldsymbol{b})}{\partial \boldsymbol{b}}$ evaluated at $\boldsymbol{b}_i$ 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:
  - $\sum |\beta^r - \beta^{r-1}| <$ small constant
  - $\max_{j=1,\ldots,p} \frac{|b_j^{(r)} - b_j^{(r-1)}|}{|b_j^{(r-1)} + \epsilon|} <$ small constant.
  - $g(\boldsymbol{b}^{(r-1)}) - g(\boldsymbol{b}^{(r)}) <$ small constant
  - $\sum \text{abs}\left(\frac{\partial g(\boldsymbol{b})}{\partial \boldsymbol{b}} \mid_{\boldsymbol{b}=\boldsymbol{b}^{(r)}}\right) <$ small constant

Normal Theory Inference

- Add assumption of normal distribution to our error model
- The model is now:

$$y_i = f(\boldsymbol{x}, \beta) + \epsilon_i,\ i = 1, ..., n,\ \epsilon_1, ..., \epsilon_n \overset{i.i.d.}{\sim} N(0, \sigma^2)$$

- Let $\hat{\beta}$ be the least squares estimate of β
- 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, $f(\boldsymbol{x}, \boldsymbol{b}) \approx \text{constant} + \hat{D}\boldsymbol{b}$ where $\hat{D}$ is D evaluated at $\boldsymbol{b}$
- $\sigma^2(\hat{D}'\hat{D})^{-1}$ can be estimated by 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
    - $MSE = \frac{SSE}{n-p}$.
    - $SSE = g(\hat{\beta}) = \sum_{i=1}^{n}[y_i - f(\boldsymbol{X}_i, \hat{\beta})]^2$.
- For n sufficiently large, $\frac{(n-p)\text{MSE}}{\sigma^2} = \frac{\text{SSE}}{\sigma^2} \sim X^2_{(n-p)}$
- All the linear model inference follows, using $\hat{D}$ as the "$\boldsymbol{X}$" matrix
- An approximate F-test $H_0 : C\beta = \mathbf{0}$ rejects H_0 at level α if and only if $F = \frac{\hat{\beta}' C'[C(\hat{D}'\hat{D})^{-1}C']^{-1}C\hat{\beta}/q}{\text{MSE}} \geq F^{(\alpha)}_{q,n-p}$ where $q = \text{rank}(c) =$ number of rows of C.
- An approximate $100(1-\alpha)\%$ confidence interval for $\boldsymbol{C}'\beta$ is

$$\boldsymbol{C}'\hat{\beta} \pm t^{\alpha}_{n-p}\sqrt{\text{MSE}\ \boldsymbol{C}'(\hat{D}'\hat{D})^{-1}\boldsymbol{C}}$$

- We also have approximate F tests for reduced vs. full model comparisons:
  $F = \frac{(SSE_{reduced} - SSE_{full})/(df_{reduced} - df_{full})}{SSE_{full}/df_{full}} \overset{H_0}{\sim} F_{df_{reduced} - df_{full}, df_{full}}$
- For example, consider a test of $H_0 : \boldsymbol{\beta}_1 = \boldsymbol{\beta}_{10}$ vs. $H_A : \boldsymbol{\beta}_1 \neq \boldsymbol{\beta}_1 0$ for some fixed $\boldsymbol{\beta}_{10}$. Let $\boldsymbol{\beta}_2 = \begin{bmatrix} \boldsymbol{\beta}_2 \\ \vdots \\ \boldsymbol{\beta}_p \end{bmatrix}$. Let
  $f_0(\boldsymbol{X}, \boldsymbol{\beta}_2) = f(\boldsymbol{X}, \begin{bmatrix} \boldsymbol{\beta}_{10} \\ \boldsymbol{\beta}_2 \end{bmatrix})$
- Then the reduced model is
  $y_i = f_0(\boldsymbol{X}_i, \boldsymbol{\beta}_2) + \epsilon_i \; i = 1, ..., n \; \epsilon_1, ..., \epsilon_n \overset{i.i.d.}{\sim} N(0, \sigma^2)$
- Then $F(\boldsymbol{\beta}_{10}) \equiv \frac{SSE_{reduced} - SSE_{full}}{MSE_{full}} \overset{H_0}{\sim} F_{1,n-p}$

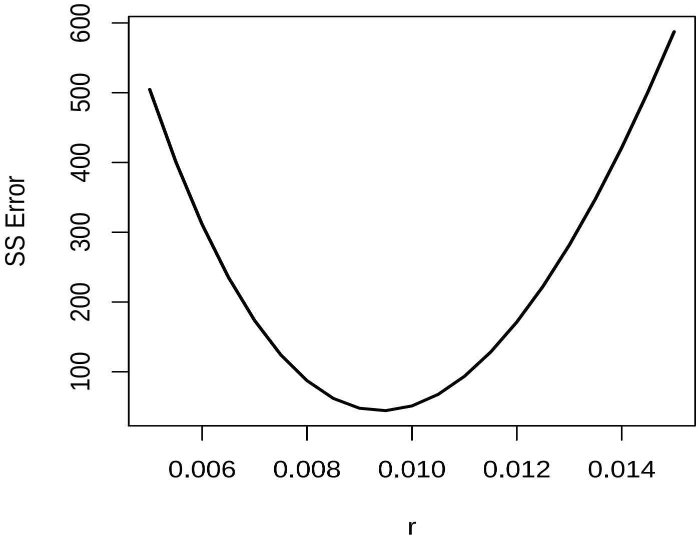
Confidence intervals

- Two ways to get a confidence interval for β_1
  1. Wald interval:
  $$\hat{\beta}_1 \pm t^{\alpha}_{n-p}\sqrt{\text{MSE}\ (\hat{D}'\hat{D})^{-1}}$$
  2. “profile” interval:
     - Consider all β_{10}. Include in $1-\alpha$ confidence interval all those β_{10} for which the F test accepts Ho: $\beta_1 = \beta_{10}$ at level α.
     - The set $\left\{\boldsymbol{\beta}_{10} : F_{(\boldsymbol{\beta}_{10})} \leq F^{(\alpha)}_{1,n-p}\right\}$ is an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{\beta}_1$.
- Same interval for linear models
- **Not the same for a nonlinear model**
- Reparameterization of β, 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.

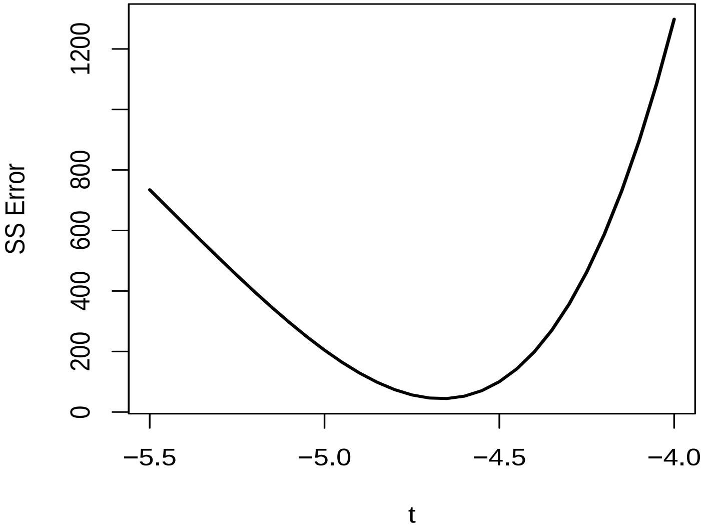
- Example: Confidence interval for Rumford temperature change
- Model 1: $temp_i = 60 + 70 \times \exp(-r * time_i) + \epsilon_i, \quad , \epsilon_i \sim N(0, \sigma^2)$
- Fit to Rumford data: $\hat{r} = 0.0094$, se $\hat{r} = 0.00042$, rMSE = 1.918
- Model 2: $temp_i = 60 + 70 \times \exp(-exp(t) * time_i) + \epsilon_i, \quad , \epsilon_i \sim N(0, \sigma^2)$
- Fit to Rumford data: $\hat{t} = -4.665$, se $\hat{t} = 0.044$, rMSE = 1.918, $\exp(-4.665) = 0.0094$

Data	Model	Wald interval	Profile interval
Rumford	1	(0.0085, 0.0103)	(0.0085, 0.0103)
	2	(-4.762, -4.568)	(-4.767, -4.571)
		(0.0085, 0.0104)	(0.0085, 0.0103)
Noisy	1	(0.0084, 0.0168)	(0.0087, 0.0171)
	2	(-4.702, -4.044)	(-4.746, -4.071)
		(0.0091, 0.0175)	(0.0087, 0.0171)

Profile SS Error for *r* parameterization



Profile SS Error for $t = \exp(r)$ parameterization



A useful property of nonlinear models

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

$$G_{ij} = \frac{\partial \alpha_i}{\partial \beta_j}$$

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

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

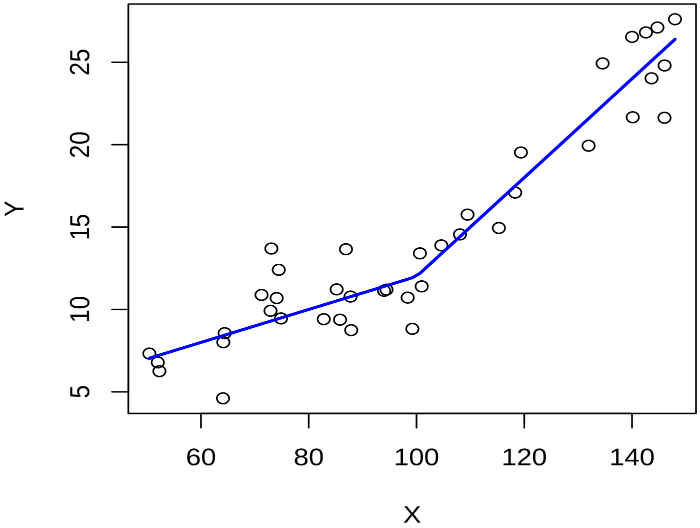
- Wald confidence interval matches Delta method ci from linear regression
- Profile confidence interval performs better

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
- $E(Y|x) = \beta_0 + \beta_1 x$ (for $x \leq 100$)
  $E(Y|x) = \beta_0 + \beta_1 x + \beta_2 (x - 100)$ (for $x > 100$)
  slope changes from β_1 to $\beta_1 + \beta_2$ at $x = 100$


Y
5
10
15
20
25
60
80
100
120
140
X

- if change point is unknown then can replace 100 by parameter τ

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

- E $Y_i \mid x_i$ is a non-linear function of τ; need non-linear regression to estimate $\hat{\tau}$.
- A common variation is “segmented” regression: second part is flat

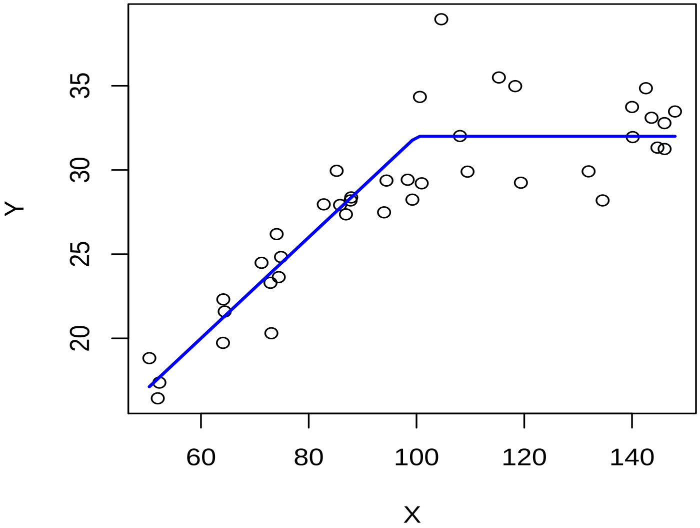
$$EY_i = \begin{cases} \beta_0 + \beta_1 x_i & x_i \le \tau \\ \beta_0 + \beta_1 \tau & x_i > \tau \end{cases} \tag{6}$$

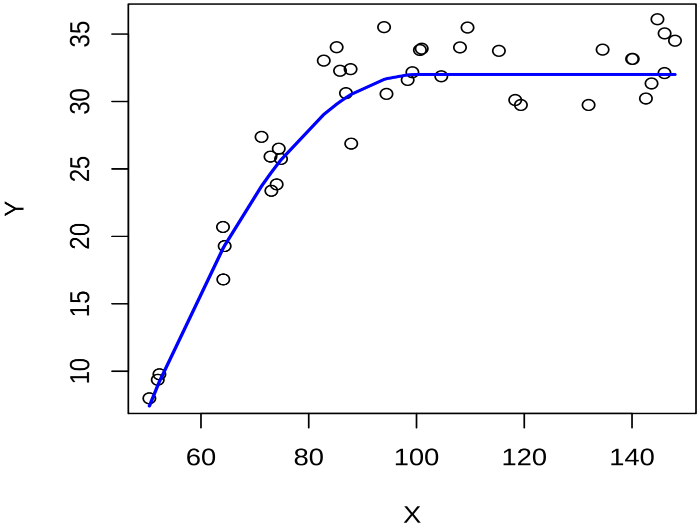
$$EY_i = \beta_0 + \beta_1 x_i (1 - z_i) + \beta_1 \tau z_i$$

- If τ unknown, need one of these two forms and NL regression
- If τ known, replace all $x_i > \tau$ with τ 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

$$EY_i = \begin{cases} \beta_0 + \beta_1(\tau - x_i)^2 & x_i \leq \tau \\ \beta_0 & x_i > \tau \end{cases}$$

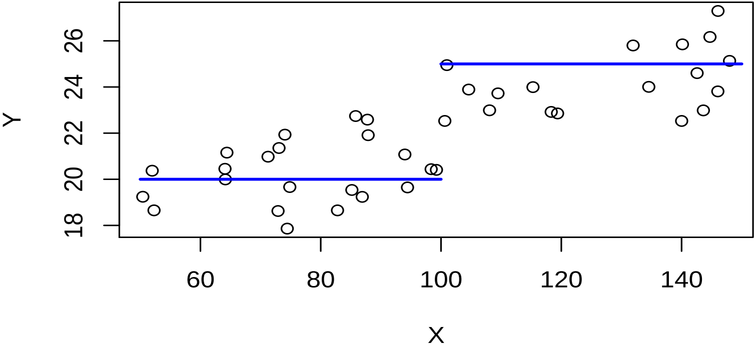

Y
20
25
30
35
60
80
100
120
140
X



Y
10
15
20
25
30
35
60
80
100
120
140
X

- “Change-point” model: E $Y_i \mid x_i$ “jumps” at τ.
- Trivial to estimate (2 means) if τ 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⋆timeˆ2),
  data=rumf, start=list(r=0.01,r2=-0.0001))

anova(rumf.nls,rumf.nls2)
```

Computing for nonlinear models

```
# estimated coefficients
coef(rumf.nls)

# profile ci’s on parameters
confint(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

$$Y_{ij} = f(X_i, \beta) + u_i + \varepsilon_{ij}$$

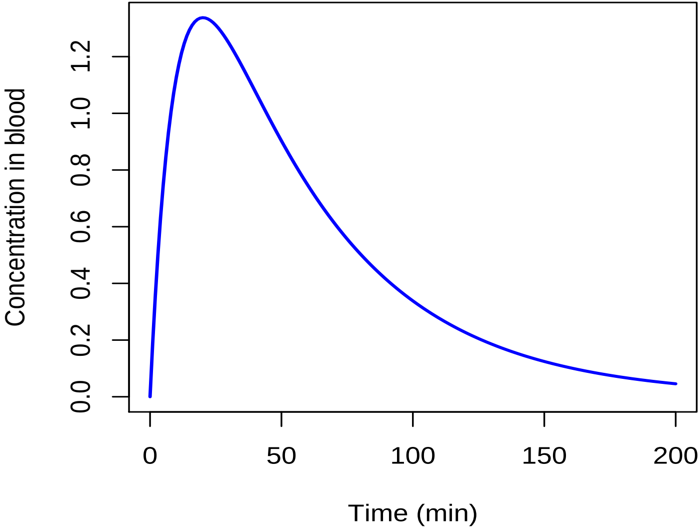
- More flexible: values of β depend on subject
- First order compartment model with absorbtion
  e.g. swallow a pill with dose D, absorbed into blood, removed by kidneys
- Two compartments: stomach, blood
  A_s: amount in stomach, A_b: amount in blood

$$\begin{aligned} \frac{d\ A_s}{dt} &= -k_a A_s \\ \frac{d\ A_b}{dt} &= k_a A_s - k_e A_b \end{aligned}$$

- Solution gives blood concentration, $C(t)$, at time t:

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

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


Concentration in blood
0.0
0.2
0.4
0.6
0.8
1.0
1.2
0
50
100
150
200
Time (min)

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

$$
\begin{aligned}
E\,C(t) \mid [k_{ai},\ k_{ei},\ k_{Vi}]' &= \frac{k_{ai} k_{ei} D}{V_{ci}} \frac{\left(e^{-k_{ai}\,t} - e^{-k_{ei}\,t}\right)}{k_{ei} - k_{ai}} \\
\begin{bmatrix} k_{ai} \\ k_{ei} \\ V_{ci} \end{bmatrix} &\sim N\left(\begin{bmatrix} k_a \\ k_e \\ V_c \end{bmatrix}, \begin{bmatrix} \sigma_a^2 & \sigma_{ae} & \sigma_{aV} \\ \sigma_{ae} & \sigma_e^2 & \sigma_{eV} \\ \sigma_{aV} & \sigma_{eV} & \sigma_V^2 \end{bmatrix} \right)
\end{aligned}
$$

- 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

  `http://www.amstat.org/sections/sbiop/webinars/WebinarSlidesBW11-08-12.pdf`

Computing for NLME’s

```
#  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
```

Computing for NLME’s

```
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

```
theoph.m1 <- nlme(
  conc ~ SSfol(Dose, Time, lKe, lKa, lCl),
  data=theoph.grp,
  fixed = lKe + lKa + lCl ~ 1,
  random = lKe + lKa + lCl ~ 1,
  start=coef(subj.1) )
# If the parameters differed by (e.g.) sex, you
#   would change to fixed = lKe + lKa + lCl ~ sex,
# If the variance/covariance matrix varied by
#   sex, use random = lKe + lKa + lCl ~ sex,

summary(theoph.m1)

# other helper functions are fitted(), predict()
#  random.effects(), residuals()
```

Computing for NLME's

```
# 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)
```

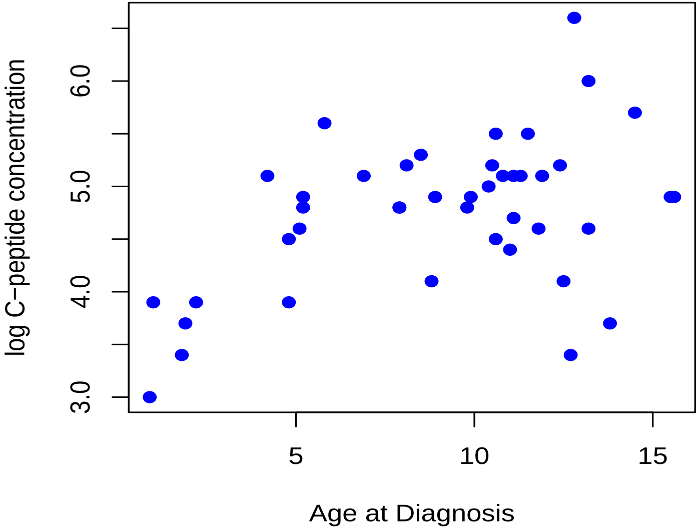
Nonparametric regression using smoothing splines

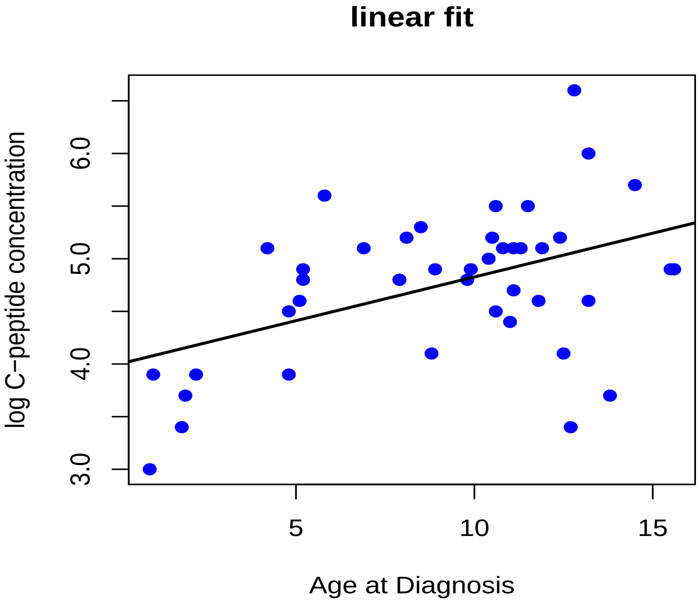
- Smoothing is fitting a smooth curve to data in a scatterplot
- Will focus on two variable problems: Y and one X
- Our model:

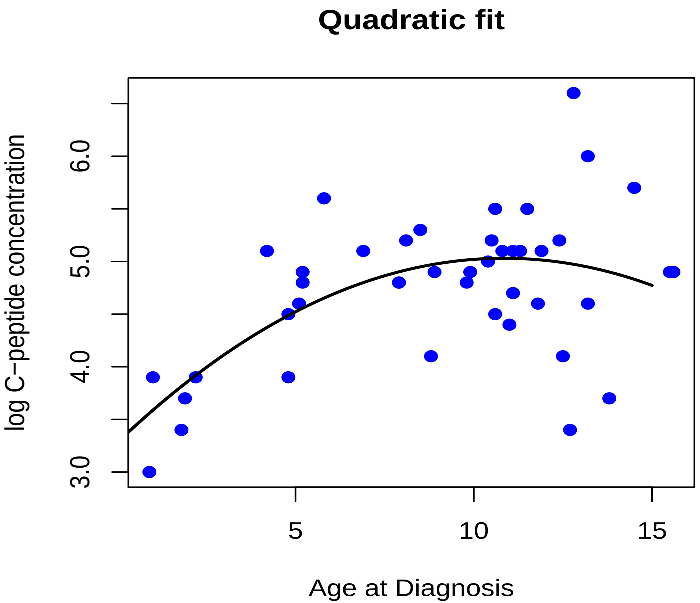
  $$y_i = f(x_i) + \varepsilon_i,$$

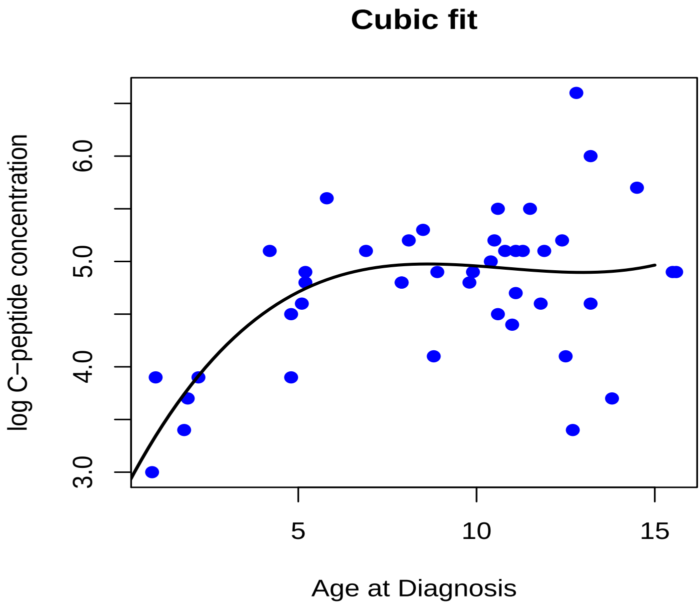
  where $\varepsilon_1, \varepsilon_1, \ldots \varepsilon_n$ are independent with mean 0
- f is some unknown smooth function
- Up to now f has a specified form with unknown parameters
  - f could be linear or nonlinear in the parameters,
  - functional form always specified
- If f not determined by the subject matter, we may prefer to let the data suggest a functional form

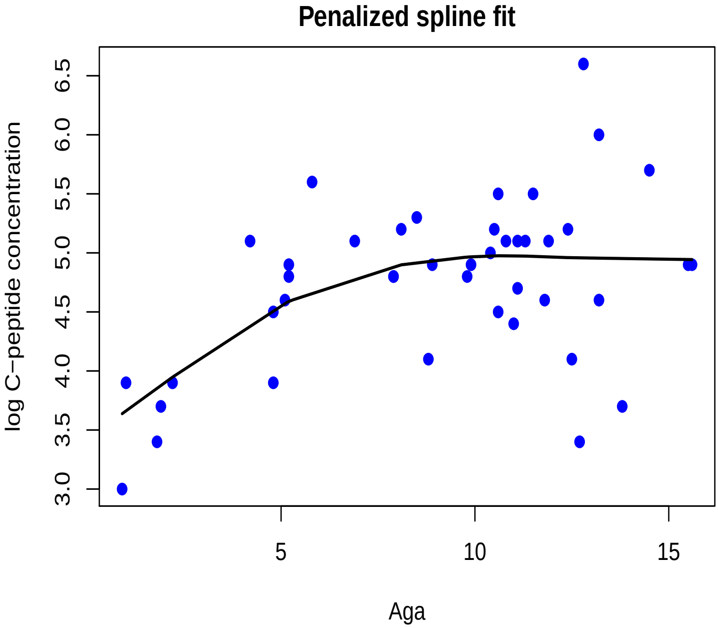
- Why estimate f?
  - can see features of the relationship between X and Y that are obscured by error variation
  - summarizes the relationship between X and Y
  - provide a diagnostic for a presumed parametric form
- Example: Diabetes data set in Hastie and Tibshirani's book *Generalized Additive Models*
- 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


log C−peptide concentration
3.0
4.0
5.0
6.0
5
10
15
Age at Diagnosis


linear fit
log C−peptide concentration
3.0
4.0
5.0
6.0
5
10
15
Age at Diagnosis


Quadratic fit
log C−peptide concentration
3.0
4.0
5.0
6.0
5
10
15
Age at Diagnosis


Cubic fit
log C−peptide concentration
3.0
4.0
5.0
6.0
5
10
15
Age at Diagnosis


Penalized spline fit
log C−peptide concentration
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
5
10
15
Aga

- A slightly different way of thinking about Gauss-Markov Linear models:
  - If we assume that f(x) is linear, then $f(x) = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 x$
  - In terms of the Gauss-Markov Linear Model $\boldsymbol{y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon}$,

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \text{ and } \boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_0 \\ \boldsymbol{\beta}_1 \end{bmatrix}$$

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

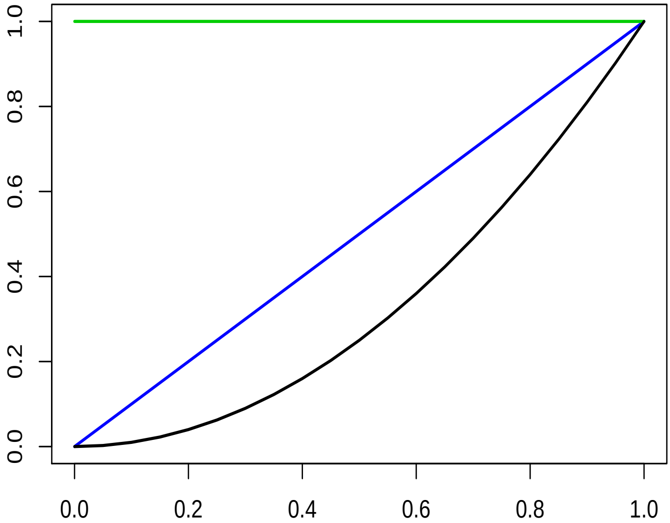
$$f(x) = \boldsymbol{\beta}_0 b_0(x) + \boldsymbol{\beta}_1 b_1(x)$$

- 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 $\boldsymbol{y} = x\boldsymbol{\beta} + \epsilon$,

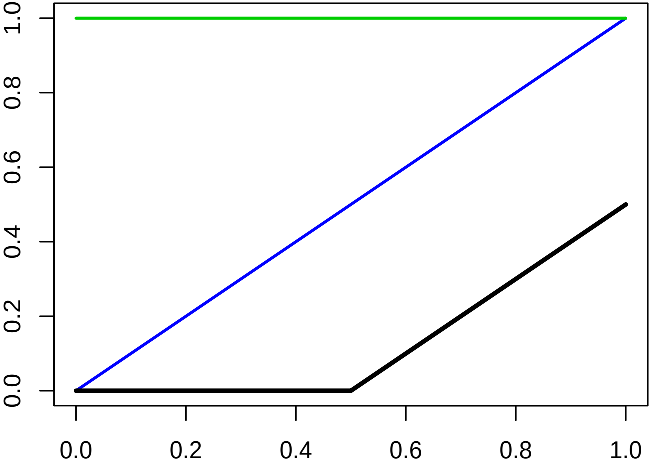
$$X = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \text{ and } \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

- The quadratic model tries to approximate f(x) as a linear combination of three basis functions:
  $b_0(x) = 1,\ b_1(x) = x,\ b_2(x) = x^2$

$$f(x) = \beta_0 b_0(x) + \beta_1 b_1(x) + \beta_2 b_2(x)$$


0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0

- 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 β_0 amd β_1) is an unknown parameter.

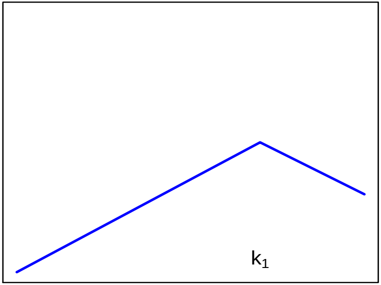

0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0

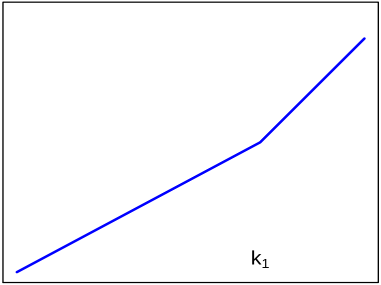
- Note that $\beta_0 b_0(x) + \beta_1 b_1(x) + u_1 S_1(x) = \beta_0 + \beta_1 X + u_1(x - k_1)^+$

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

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

- This is clearly a continuous function (because it is a linear combination of continuous functions), and it is piecewise linear.

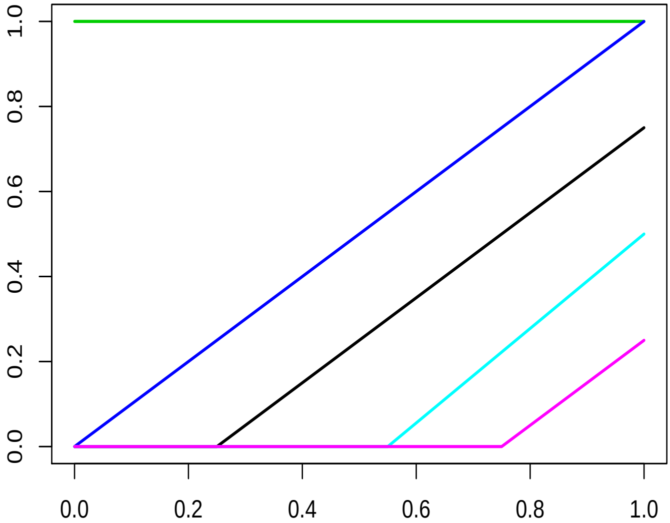


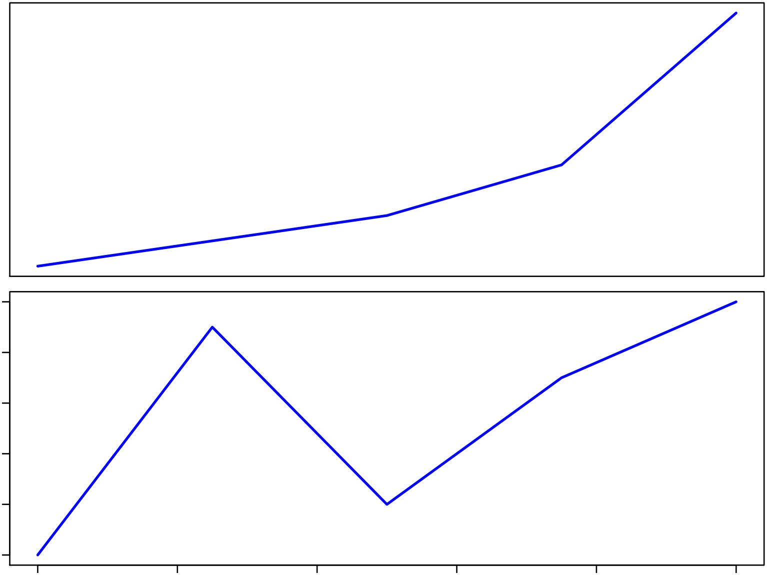


- The function $\beta_0 + \beta_1 x + u_1(x - k_1)^+$ is a simple example of a linear spline function.
- The value k_1 is known as a knot.
- As a Gauss-Markov Linear Model, $\boldsymbol{y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon}$,

$$X = \begin{bmatrix} 1 & x_1 & (x_1 - k_1)^+ \\ 1 & x_2 & (x_2 - k_1)^+ \\ \vdots & \vdots & \vdots \\ 1 & x_n & (x_n - k_1)^+ \end{bmatrix} \text{ and } \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ u_1 \end{bmatrix}$$

- We can make our linear spline function more flexible by adding more knots $k_1, ..., k_k$ so that f(x) is approximated by $\beta_0 + \beta_1 x + \sum_{j=1}^{k} u_j s_j(x) = \beta_0 + \beta_1 x + \sum_{j=1}^{k} u_j (x - k_j)^+$


0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0



- 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 $\boldsymbol{y} = X\beta + \epsilon$, where

$$X = \begin{bmatrix} 1 & x_1 & (x_1 - k_1)^+ & (x_1 - k_2)^+ ... (x_1 - k_k)^+ \\ 1 & x_2 & (x_2 - k_1)^+ & (x_2 - k_2)^+ ... (x_2 - k_k)^+ \\ \vdots & \vdots & \vdots & \vdots \qquad \vdots \\ 1 & x_n & (x_n - k_1)^+ & (x_n - k_2)^+ ... (x_n - k_k)^+ \end{bmatrix}$$

  and $\beta = (\beta_0, \beta_1, u_1, u_2, ..., u_k)'$
- The OLS estimator of β is $(x'x)^{-1} x' \boldsymbol{y}$.
  This is the BLUE of β, 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

Curve	β_1	u_1	u_2	u_3	$\sum u_i^2$
Smoother	0.4	0.0	0.4	1.6	2.72
Wigglier	3.6	-6.4	4.8	-0.8	64.64

- If we really believe the true $f(x)$ is a linear spline function with knots at $k_1, k_2, ..., k_k$, then $\hat{\boldsymbol{\beta}} = (x'x)^{-1}\boldsymbol{y}$ is the best linear unbiased estimator of $(\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, u_1, ..., u_k)'$.
- 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 coeffients closer to 0
- Use penalized least squares to estimate a smoother curve.
- Find $\boldsymbol{\beta} = (\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, u_1, ..., u_k)'$ that minimizes $(\boldsymbol{y} - x\boldsymbol{\beta})'(\boldsymbol{y} - x\boldsymbol{\beta}) + \lambda^2 \sum_{j=1}^{k} u_j^2$, where
  λ^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 = diag(0,0,1,...,1) (k terms), then

$$
\begin{aligned}
(\boldsymbol{y} - x\beta)'(\boldsymbol{y} - x\beta) + \lambda^2 \sum_{j=1}^{k} u_j^2 &= (\boldsymbol{y} - x\beta)'(\boldsymbol{y} - x\beta) + \lambda^2 \beta' D \beta \\
&= \boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{y}'x\boldsymbol{\beta} + \boldsymbol{\beta}'x'x\boldsymbol{\beta} + \lambda^2 \boldsymbol{\beta}' D \boldsymbol{\beta} \\
&= \boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{y}'x\boldsymbol{\beta} + \boldsymbol{\beta}'(x'x + \lambda^2 D)\boldsymbol{\beta}
\end{aligned}
$$

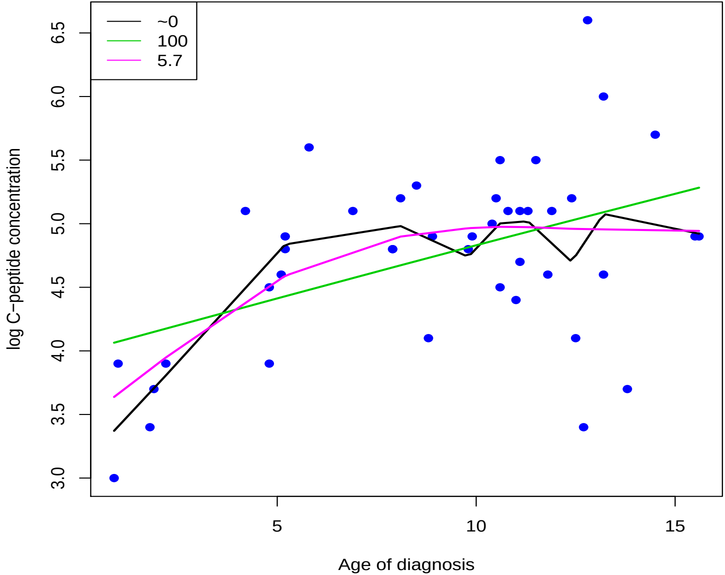
- Set derivatives with respect to β equal to **0**
- estimating equations: $(x'x + \lambda^2 D)\boldsymbol{\beta} \equiv x'\boldsymbol{y}$
- solution: $\hat{\boldsymbol{\beta}}_{\lambda^2} = (x'x + \lambda^2 D)^{-1} x'\boldsymbol{y}$ for any fixed $\lambda^2 \geq 0$
- predicted values: $\hat{\boldsymbol{y}}_{\lambda^2} \equiv x\hat{\boldsymbol{\beta}}_{\lambda^2} = x(x'x + \lambda^2 D)^{-1} x'\boldsymbol{y}$

- You choose λ^2 and the knots $k_1, ..., k_k$.
- As $\lambda^2 \rightarrow 0, \hat{\boldsymbol{\beta}}_{\lambda^2} \rightarrow \hat{\boldsymbol{\beta}} = (x'x)^{-1}x'\boldsymbol{y}$.
  Small λ^2 results in non-smooth fit.
- As $\lambda^2 \rightarrow \infty, \hat{\boldsymbol{\beta}}_{\lambda^2} \rightarrow \begin{bmatrix} \hat{\boldsymbol{\beta}}_0 \\ \hat{\boldsymbol{\beta}}_1 \\ \mathbf{0} \end{bmatrix}$

  In the limit, $\lambda^2 \rightarrow \infty$ results in the least squares fit
- When $f(x)$ is defined as $f(x) = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 x + \sum_{j=1}^{k} u_j (x - k_j)^+$, the resulting function is continuous but the 1st and 2nd derivatives are not.
- 1st and 2nd derivatives are undefined at the knots

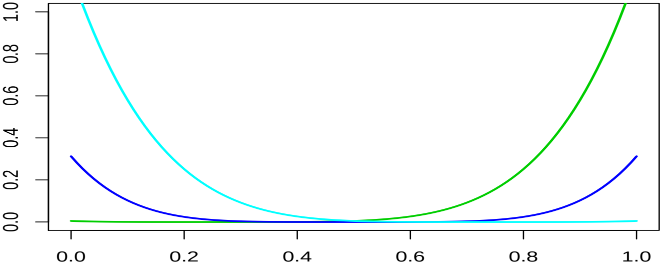
A smoother smoother

- Next page: fitted penalized regression splines for 3 smoothing parameters: $\sim$0, 100, 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


~0
100
5.7
log C−peptide concentration
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
5
10
15
Age of diagnosis

- We'll talk about thin plate splines because they provide an easy to implement way to fit multiple X's
  E $y = f(x_1, x_2)$ as well as 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 λ^2? or what u_k's
  - reminder: 0 ⇒ no smoothing (linear or quadratic in tps)
    large ⇒ 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 λ^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 λ^2 from the data set that excludes the i^{th} observation.
- Find λ^2 value that minimizes $CV(\lambda^2)$. Perhaps compute $CV(\lambda^2)$ for a grid of λ^2 values
- Requires a **LOT** of computing (each obs, many λ^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 $\hat{\boldsymbol{y}} = x(x'x + \lambda^2 D)^{-1} x' y = S_{\lambda^2,ii} y$
- OLS: $\hat{\boldsymbol{y}} = \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-}\boldsymbol{X}' y = P_{\boldsymbol{X}} y$
- The smoother matrix S_{λ^2} is analogous to the “hat” or projection matrix, $P_{\boldsymbol{X}}$ 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 $H = P_x = x(x'x)^{-}x'$.
- $h_{ii} =$ ”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 $\sum_{i=1}^{n}(y_i - \hat{y}_{-i})^2 = \sum_{i=1}^{n}(\frac{y_i - \hat{y}_i}{1 - h_{ii}})^2$ 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 trace(S_{λ^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 λ^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 $trace(H)$, 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 $i = 1, ..., n$; where $e_1, ..., e_n \overset{i.i.d.}{\sim} (0, \sigma^2)$
- Suppose we add the following assumptions: $u_1, ..., u_k \overset{i.i.d.}{\sim} N(0, \sigma_u^2)$ independent of $e_1, ..., e_n \overset{i.i.d.}{\sim} N(0, \sigma_e^2) . (\sigma_e^2 \equiv \sigma^2)$
- Then we may write our model as $\boldsymbol{y} = x\boldsymbol{\beta} + Z\boldsymbol{u} + \boldsymbol{\epsilon}$, where

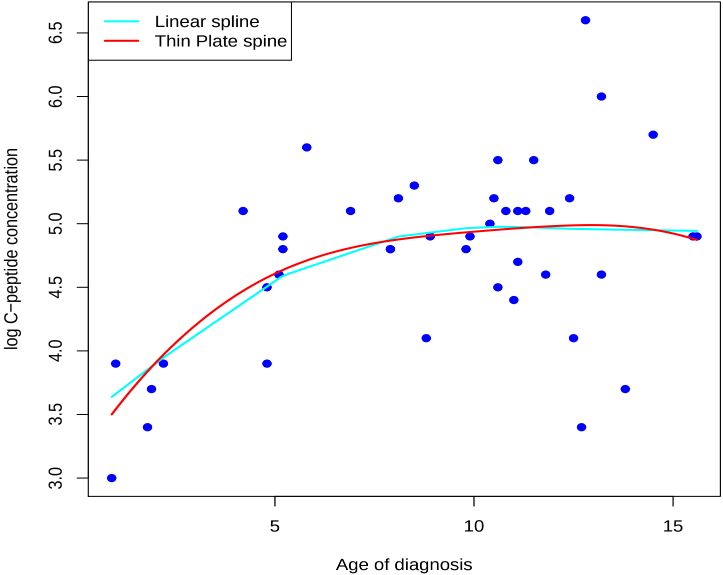
$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_0 \\ \boldsymbol{\beta}_1 \end{bmatrix} Z = \begin{bmatrix} (x_1 - k_1)^+ & . & . & . & (x_1 - k_k)^+ \\ (x_2 - k_1)^+ & . & . & . & (x_2 - k_k)^+ \\ \vdots & & & & \vdots \\ (x_n - k_1)^+ & . & . & . & (x_n - k_k)^+ \end{bmatrix}$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \end{bmatrix} \epsilon = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{u} \\ \epsilon \end{bmatrix} \sim N\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \sigma_u^2 I & 0 \\ 0 & \sigma_u^2 I \end{bmatrix} \right)$$

- This is a linear mixed effects model!

- It can be shown that the BLUP of $X\beta + Z\boldsymbol{u}$ is equal to $w(w'w + \frac{\sigma_e^2}{\sigma_u^2 D})^{-1} w'\boldsymbol{y}$ where $w = [x, z]$.
- Thus, the BLUP of $X\beta + Z\boldsymbol{u}$ is equal to $S_{\frac{\sigma_e^2}{\sigma_u^2}}\boldsymbol{y} =$ (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 σ_u^2 and σ_e^2. (Denote estimates by $\hat{\sigma}_u^2$ and $\hat{\sigma}_e^2$.)
- Then we can estimate β by $\hat{\beta}_{\hat{\boldsymbol{\Sigma}}} = (x'\hat{\boldsymbol{\Sigma}}^{-1}x)^{-1}x'\hat{\boldsymbol{\Sigma}}\boldsymbol{y}$ and predict $\boldsymbol{u}$ by $\hat{\boldsymbol{u}}_{\hat{\boldsymbol{\Sigma}}} = \hat{G}Z'\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{y} - x\hat{\beta}_{\hat{\boldsymbol{\Sigma}}}) = \hat{\sigma}_u^2 Z'\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{y} - x\hat{\beta}_{\hat{\boldsymbol{\Sigma}}})$ where $\hat{\boldsymbol{\Sigma}} = \hat{\sigma}_u^2 ZZ' + \hat{\sigma}_e^2 \boldsymbol{I}$
- The resulting coefficients $\begin{bmatrix} \hat{\beta}_{\hat{\boldsymbol{\Sigma}}} \\ \hat{\boldsymbol{u}}_{\hat{\boldsymbol{\Sigma}}} \end{bmatrix}$ will be equal to the estimate obtained using penalized least squares with smoothing parameter $\lambda^2 = \frac{\hat{\sigma}_e^2}{\hat{\sigma}_u^2}$


Linear spline
Thin Plate spine
log C−peptide concentration
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
5
10
15
Age of diagnosis

- Still need to choose number of knots (k) and their locations $k_1, ..., k_k$
- 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

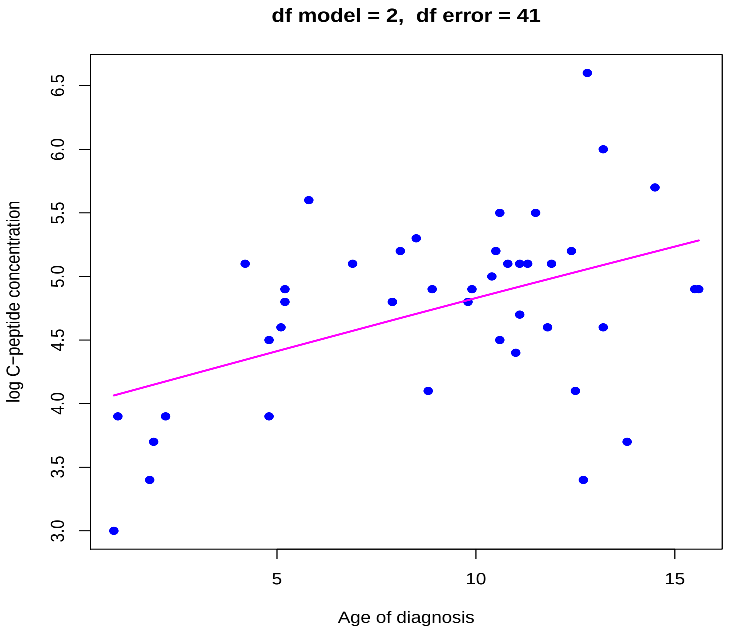
- If we want a confidence or prediction interval around the predicted line, need to know df for error.
- If we want to compare models (e.g. E$y = \beta_0 + \beta_1 x$ vs E$y = f(x)$), need to know df for penalized spline fit
- Can do this test because
  - E$y = \beta_0 + \beta_1 x$ is nested in E$y = f(x)$ fit as a linear spline
  - E$y = \beta_0 + \beta_1 x + \beta_2 x^2$ is nested in E$y = f(x)$ 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 $\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, u_1, u_2, ..., u_k$.

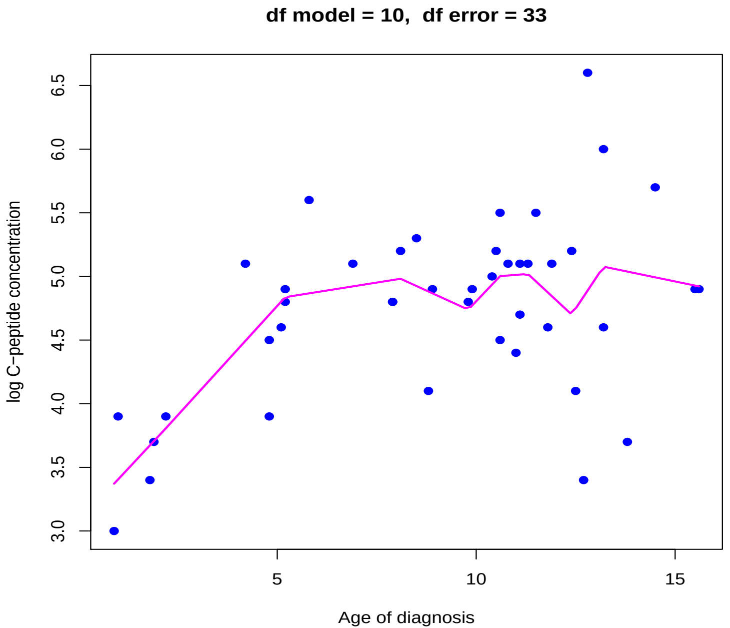
- However, $u_1, u_2, ..., u_k$ are not completely free parameters because of penalization.
- The effective number of parameters is lower than k+2 and depends on the value of the smoothing parameter λ^2.
- Recall that our estimates of $\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, u_1, u_2, ..., u_k$ minimize $\sum_{i=1}^{n}(y_i - \boldsymbol{\beta}_0 - \boldsymbol{\beta}_1 x_i - \sum_{j=1}^{k} u_j(x_i - k_j)^+)^2 + \lambda^2 \sum_{j=1}^{k} u_j^2$
- A larger λ^2 means less freedom to choose values for $u_1, ..., u_k$ for from 0.
- Thus, the number of effective parameters should decrease as λ^2 increases.
- In the Gauss-Markov framework with no penalization, the number of free parameters used to estimate the mean of $\boldsymbol{y}(x\beta)$ is $rank(x) = rank(P_x) = trace(P_x)$

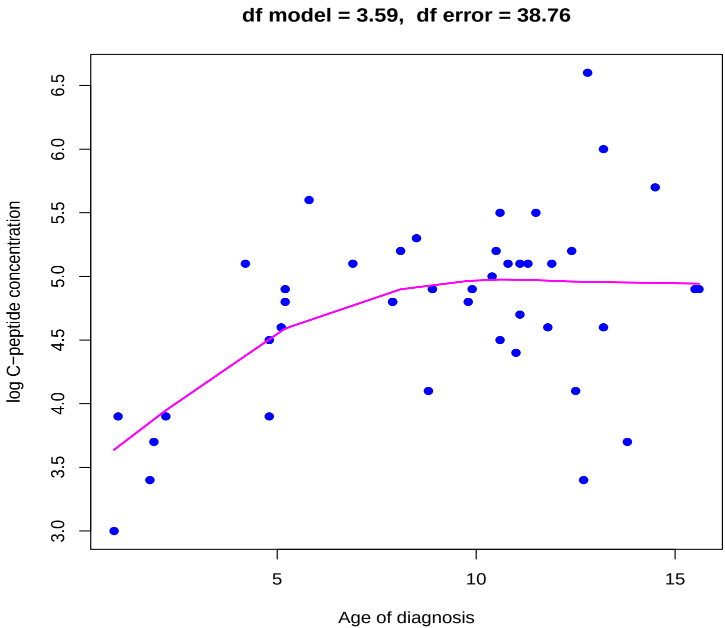
- For a smoother, the smoother matrix S plays the role of P_x.
- For penalized linear splines, the smoother matrix is $S_{\lambda^2} = x(x'x + \lambda^2 D)^{-1} x'$ where

$$X = \begin{bmatrix} 1 & x_1 & (x_1 - k_1)^+ ... (x_1 - k_k)^+ \\ 1 & x_2 & (x_2 - k_1)^+ ... (x_2 - k_k)^+ \\ . & . & . \\ . & . & . \\ . & . & . \\ 1 & x_n & (x_n - k_1)^+ ... (x_n - k_k)^+ \end{bmatrix} \quad D = \begin{bmatrix} \overset{0}{2 \times 2} & 0 \\ 0 & \overset{I}{k \times k} \end{bmatrix}$$

- 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
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
5
10
15
Age of diagnosis


df model = 10, df error = 33
log C−peptide concentration
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
5
10
15
Age of diagnosis


df model = 3.59, df error = 38.76
log C−peptide concentration
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
5
10
15
Age of diagnosis

- Recall that our basic model is $y_i = f(x_i) + \epsilon_i \quad (i = 1, ..., n)$ where $e_1, ..., e_n \overset{i.i.d.}{\sim} (0, \sigma^2)$.
- How should we estimate σ^2 ?
- A natural estimator would be $MSE \equiv \frac{\sum_{i=1}^{n} \left\{ y_i - \hat{f}(x_i, \lambda^2) \right\}^2}{df_{ERROR}}$
- 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 $\boldsymbol{w}$ random and A fixed $E(\boldsymbol{w}' A \boldsymbol{w}) = E(\boldsymbol{w})' A E(\boldsymbol{w}) + tr(AVar(\boldsymbol{w}))$

$$\text{Let } \boldsymbol{f} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix} \text{ and } \quad \hat{\boldsymbol{f}}_{\lambda^2} = \begin{bmatrix} \hat{f}(x_1; \lambda^2) \\ \hat{f}(x_2; \lambda^2) \\ \vdots \\ \hat{f}(x_n; \lambda^2) \end{bmatrix} = S_{\lambda^2} \boldsymbol{y}$$

- Then, $E[\sum_{i=1}^{n} \left\{ y_i - \hat{f}(x_i; \lambda^2) \right\}^2]$

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

- Thus, if we define $df_{ERROR} = n - 2\text{tr}(S_{\lambda^2}) + \text{tr}(S'_{\lambda^2} S_{\lambda^2})$, $E(MSE) \approx \sigma^2$

- The Standard Error of $\hat{f}(x;\sigma^2)$:

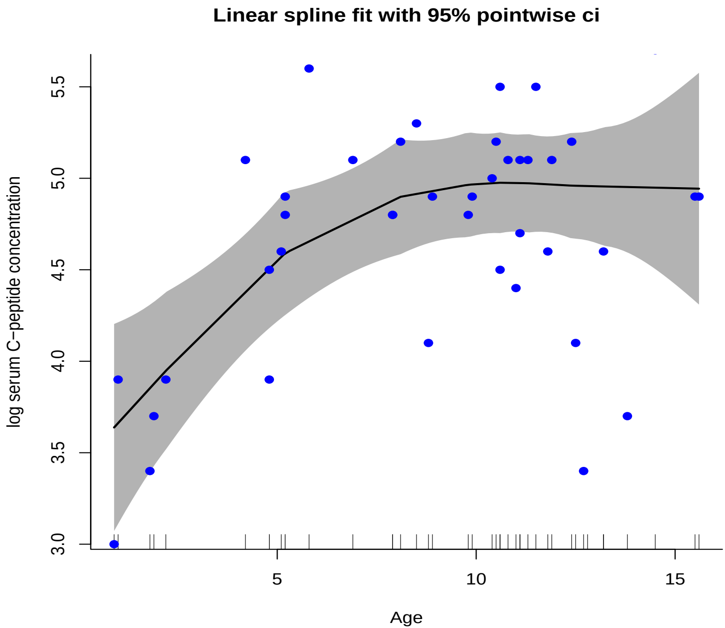
$$\hat{f}(x;\lambda^2) = \hat{\beta}_o + \hat{\beta}_1 x + \textstyle\sum_{j=1}^{k} \hat{u}_j (x - k_j)^+$$

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

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

- If λ^2 and the knots, k_i, are fixed and not chosen as a function of the data, $\boldsymbol{C}$ is just a fixed (nonrandom) vector.

- Thus, $Var[\hat{f}(x;\lambda^2)] = Var(\boldsymbol{C}'\boldsymbol{y}) = \boldsymbol{C}'\sigma^2 I\boldsymbol{C} = \sigma^2\boldsymbol{C}'\boldsymbol{C}$
- It follows that the standard error for $\hat{f}(x;\lambda^2)$ is $SE[\hat{f}(x;\lambda^2)] = \sqrt{MSE\ \boldsymbol{C}'\boldsymbol{C}}$
- If λ^2 and/or the knots are selected based on the data (as is usually the case), $\sqrt{MSE\ \boldsymbol{C}'\boldsymbol{C}}$ 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 $\boldsymbol{C}$ 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 $t_{1-\alpha/2,dfe}\sqrt{Var[\hat{f}(x;\lambda^2)]}$,
  where *dfe* is the df_{ERROR} defined a few pages ago


Linear spline fit with 95% pointwise ci
log serum C−peptide concentration
3.0
3.5
4.0
4.5
5.0
5.5
5
10
15
Age

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 Ey
- Our discussion has only considered $y_i \sim N(\mathrm{Ey_i}, \sigma^2)$
- Can combine with GLM ideas, e.g.: $y_i \sim \mathit{Poisson}(f(x_i))$ or $\mathit{Binomial}(f(x_i))$

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 pecularities
# Previous instructors of 511 used spm()
#  the results are slightly different and I haven't ha
#  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 concent

# a couple of pecularities
# 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.
```

Computing splines

```
# a third pecularity: lots of useful values have be ex
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
```