Bayesian Statistics for Genetics
Lecture 4: Linear Regression

Ken Rice

Swiss Institute in Statistical Genetics

September, 2017

Regression models: overview

How does an outcome Y vary as a function of $\boldsymbol{x} = \{x_1, \ldots, x_p\}$?

- What are the effect sizes?
- What is the effect of x_1, in observations that have the same $x_2, x_3, \ldots x_p$ (a.k.a. "keeping these covariates constant")?
- Can we predict Y as a function of $\boldsymbol{x}$?

These questions can be assessed via a **regression model** $p(y|\boldsymbol{x})$.

Regression models: overview

Parameters in a regression model can be estimated from data:

$$\begin{pmatrix} y_1 & x_{1,1} & \cdots & x_{1,p} \\ \vdots & \vdots & & \vdots \\ y_n & x_{n,1} & \cdots & x_{n,p} \end{pmatrix}$$

The data are often expressed in matrix/vector form:

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \quad \mathsf{X} = \begin{pmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \end{pmatrix} = \begin{pmatrix} x_{1,1} & \cdots & x_{1,p} \\ \vdots & & \vdots \\ x_{n,1} & \cdots & x_{n,p} \end{pmatrix}$$

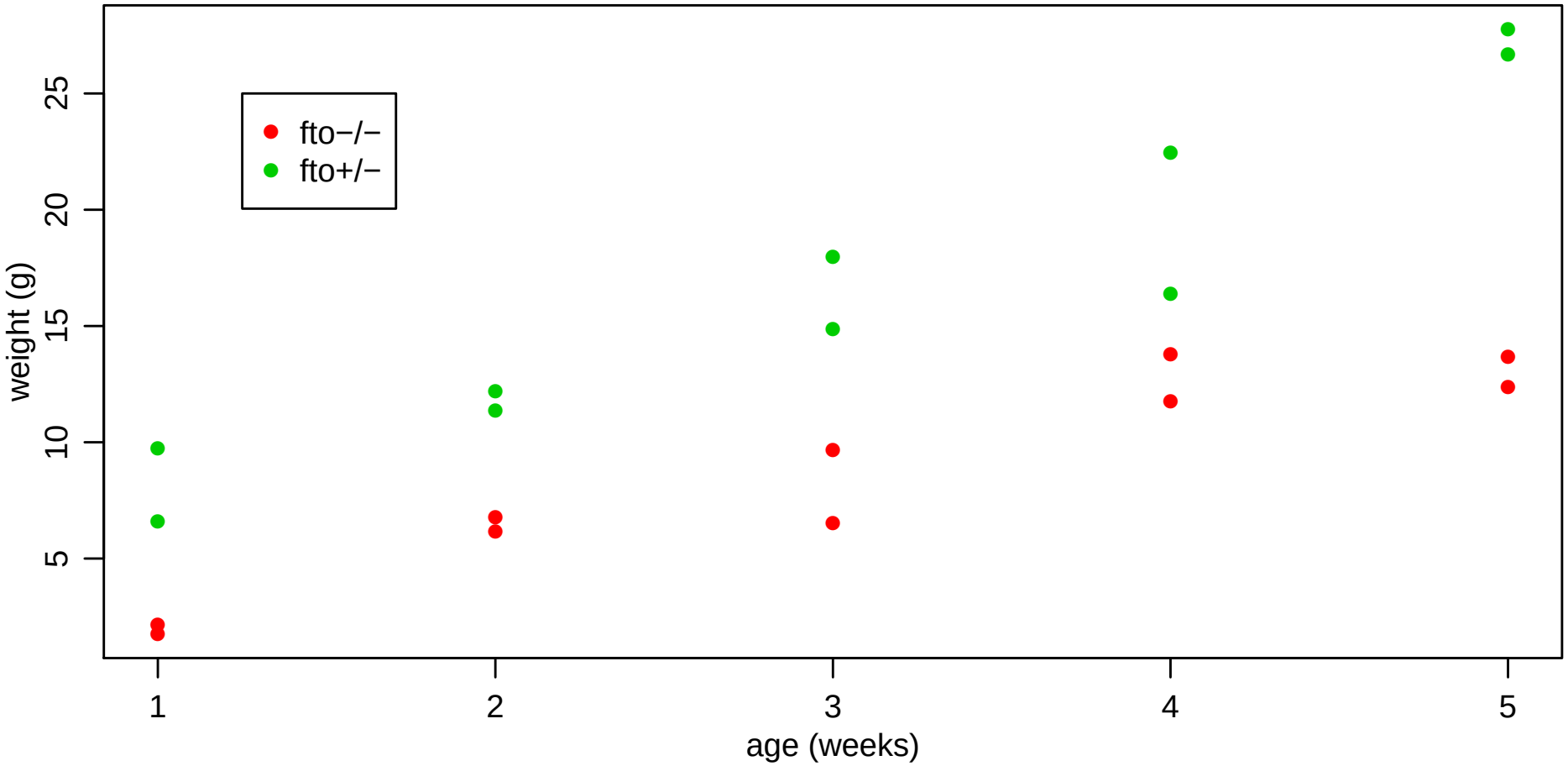
FTO example: design

FTO gene is hypothesized to be involved in growth and obesity.

Experimental design:

- 10 $fto+/-$ mice
- 10 $fto-/-$ mice
- Mice are sacrificed at the end of 1-5 weeks of age.
- Two mice in each group are sacrificed at each age.

FTO example: data



FTO example: analysis

- y = weight
- x_g = indicator of fto heterozygote $\in \{0, 1\}$ = number of "+" alleles
- x_a = age in weeks $\in \{1, 2, 3, 4, 5\}$

How can we estimate $p(y|x_g, x_a)$?

Cell means model:

$genotype$	age				
$-/-$	$\theta_{0,1}$	$\theta_{0,2}$	$\theta_{0,3}$	$\theta_{0,4}$	$\theta_{0,5}$
$+/-$	$\theta_{1,1}$	$\theta_{1,2}$	$\theta_{1,3}$	$\theta_{1,4}$	$\theta_{1,5}$

Problem: 10 parameters – only two observations per cell

Linear regression

Solution: Assume smoothness as a function of age. For each group,

$$y = \alpha_0 + \alpha_1 x_a + \epsilon.$$

This is a linear regression model. Linearity means "linear in the parameters", i.e. several covariates, each multiplied by corresponding α, and added.

A more complex model might assume e.g.

$$y = \alpha_0 + \alpha_1 x_a + \alpha_2 x_a^2 + \alpha_3 x_a^3 + \epsilon,$$

– but this is still a linear regression model, even with age^2, age^3 terms.

Multiple linear regression

With enough variables, we can describe the regressions for both groups simultaneously:

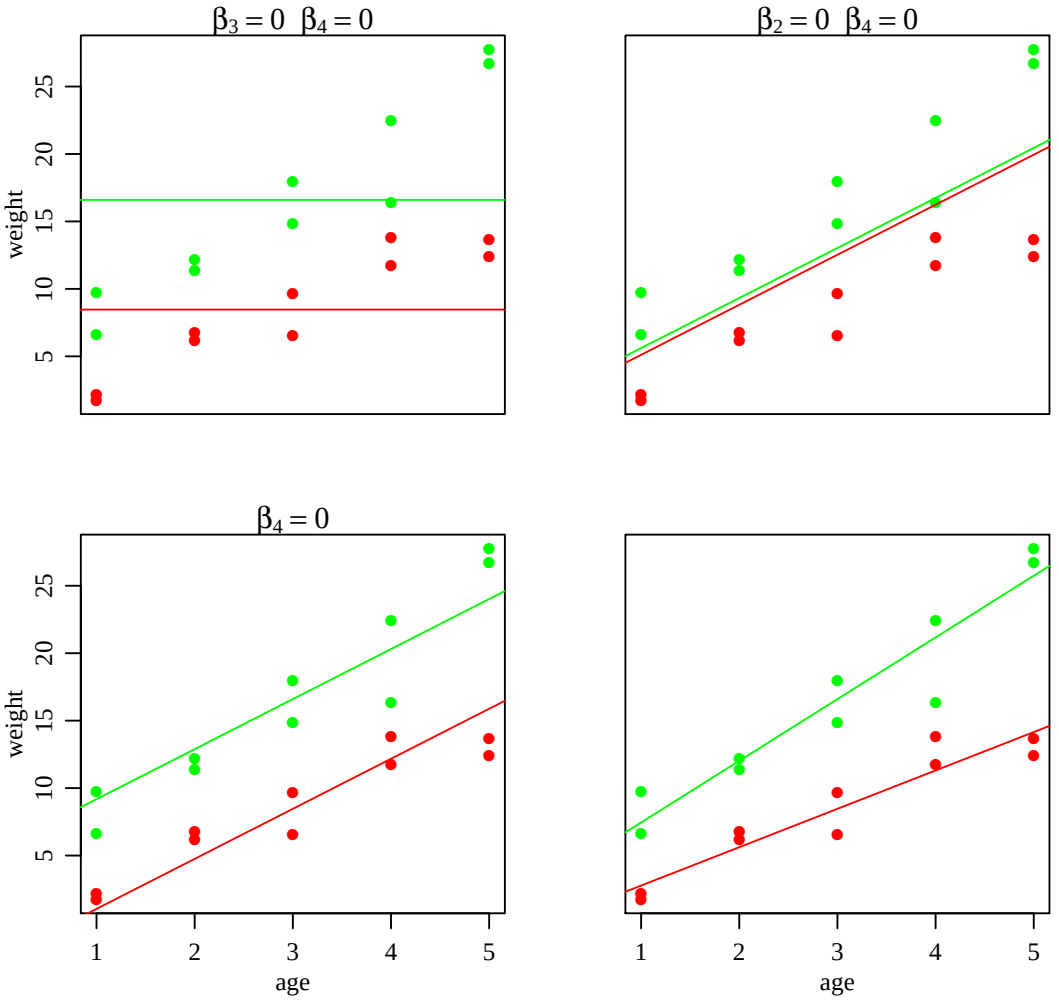
$$
\begin{aligned}
Y_i &= \beta_1 x_{i,1} + \beta_2 x_{i,2} + \beta_3 x_{i,3} + \beta_4 x_{i,4} + \epsilon_i \text{ , where} \\
x_{i,1} &= 1 \text{ for each subject } i \\
x_{i,2} &= 0 \text{ if subject } i \text{ is homozygous, 1 if heterozygous} \\
x_{i,3} &= \text{age of subject } i \\
x_{i,4} &= x_{i,2} \times x_{i,3}
\end{aligned}
$$

Note that under this model,

$$
\begin{aligned}
\mathbb{E}[\,Y|\boldsymbol{x},] &= \beta_1 + \beta_3 \times \text{age} \quad \text{if } x_2 = 0, \text{ and} \\
\mathbb{E}[\,Y|\boldsymbol{x}\,] &= (\beta_1 + \beta_2) + (\beta_3 + \beta_4) \times \text{age} \quad \text{if } x_2 = 1.
\end{aligned}
$$

Multiple linear regression

For graphical thinkers...



Normal linear regression

How does each Y_i vary around its mean $\mathbb{E}[\,Y_i|\boldsymbol{\beta}, \boldsymbol{x}_i\,]$, ?

$$\begin{aligned} Y_i &= \boldsymbol{\beta}^T \boldsymbol{x}_i + \epsilon_i \\ \epsilon_1, \ldots, \epsilon_n &\sim \text{i.i.d. normal}(0, \sigma^2). \end{aligned}$$

This assumption of Normal errors specifies the likelihood:

$$\begin{aligned} p(y_1, \ldots, y_n | \boldsymbol{x}_1, \ldots \boldsymbol{x}_n, \boldsymbol{\beta}, \sigma^2) &= \prod_{i=1}^{n} p(y_i | \boldsymbol{x}_i, \boldsymbol{\beta}, \sigma^2) \\ &= (2\pi\sigma^2)^{-n/2} \exp\{-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i)^2\}. \end{aligned}$$

Note: in large(r) sample sizes, analysis is "robust" to the Normality assumption—but here we **rely** on the mean being linear in the $\boldsymbol{x}$'s, and on the ϵ_i's variance being constant with respect to $\boldsymbol{x}$.

Matrix form

- Let $\boldsymbol{y}$ be the n-dimensional column vector $(y_1, \ldots, y_n)^T$;
- Let $\mathbf{X}$ be the $n \times p$ matrix whose ith row is $\boldsymbol{x}_i$

Then the normal regression model is that

$$\boldsymbol{y}|\mathbf{X}, \boldsymbol{\beta}, \sigma^2 \sim \text{ multivariate normal } (\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}),$$

where $\mathbf{I}$ is the $p \times p$ identity matrix and

$$\mathbf{X}\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{x}_1 \rightarrow \\ \boldsymbol{x}_2 \rightarrow \\ \vdots \\ \boldsymbol{x}_n \rightarrow \end{pmatrix} \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} = \begin{pmatrix} \beta_1 x_{1,1} + \cdots + \beta_p x_{1,p} \\ \vdots \\ \beta_1 x_{n,1} + \cdots + \beta_p x_{n,p} \end{pmatrix} = \begin{pmatrix} \mathbb{E}Y_1|\boldsymbol{\beta}, \boldsymbol{x}_1 \\ \vdots \\ \mathbb{E}Y_n|\boldsymbol{\beta}, \boldsymbol{x}_n \end{pmatrix}.$$

Ordinary least squares estimation

What values of $\boldsymbol{\beta}$ are consistent with our data $\boldsymbol{y}, \mathbf{X}$?

Recall

$$p(y_1, \ldots, y_n | \boldsymbol{x}_1, \ldots \boldsymbol{x}_n, \boldsymbol{\beta}, \sigma^2) = (2\pi\sigma^2)^{-n/2} \exp\{-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i)^2\}.$$

This is big when $\text{SSR}(\boldsymbol{\beta}) = \sum (y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i)^2$ is small – where we define

$$\begin{aligned} \text{SSR}(\boldsymbol{\beta}) &= \sum_{i=1}^{n} (y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i)^2 = (\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})^T (\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}) \\ &= \boldsymbol{y}^T \boldsymbol{y} - 2\boldsymbol{\beta}^T \mathbf{X}^T \boldsymbol{y} + \boldsymbol{\beta}^T \mathbf{X}^T \mathbf{X} \boldsymbol{\beta} \,. \end{aligned}$$

What value of $\boldsymbol{\beta}$ makes this the smallest, i.e. gives fitted values closest to the outcomes $y_1, y_2, \ldots y_n$?

OLS: with calculus

'Recall' that...

1. a minimum of a function $g(z)$ occurs at a value z such that $\frac{d}{dz}g(z) = 0$;
2. the derivative of $g(z) = az$ is a and the derivative of $g(z) = bz^2$ is $2bz$.

Doing this for SSR...

$$\begin{aligned} \frac{d}{d\boldsymbol{\beta}}\text{SSR}(\boldsymbol{\beta}) &= \frac{d}{d\boldsymbol{\beta}}\left(\boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{\beta}^T\mathbf{X}^T\boldsymbol{y} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}\right) \\ &= -2\mathbf{X}^T\boldsymbol{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}\ , \\ \text{and so } \frac{d}{d\boldsymbol{\beta}}\text{SSR}(\boldsymbol{\beta}) = 0 &\Leftrightarrow -2\mathbf{X}^T\boldsymbol{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = 0 \\ &\Leftrightarrow \mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^T\boldsymbol{y} \\ &\Leftrightarrow \boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y}\,. \end{aligned}$$

$\hat{\boldsymbol{\beta}}_{\text{ols}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y}$ is the Ordinary Least Squares (OLS) estimator of $\boldsymbol{\beta}$.

OLS: without calculus

The calculus-free, algebra-heavy version – which relies on knowing the answer in advance!

Writing $\Pi = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, and noting that $\mathbf{X} = \Pi\mathbf{X}$ and $\mathbf{X}\hat{\boldsymbol{\beta}}_{\text{ols}} = \Pi\boldsymbol{y}$;

$$\begin{aligned}(\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}) &= (\boldsymbol{y} - \Pi\boldsymbol{y} + \Pi\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \Pi\boldsymbol{y} + \Pi\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}) \\ &= ((I - \Pi)\boldsymbol{y} + \Pi(\hat{\boldsymbol{\beta}}_{\text{ols}} - \boldsymbol{\beta}))^T((I - \Pi)\boldsymbol{y} + \Pi(\hat{\boldsymbol{\beta}}_{\text{ols}} - \boldsymbol{\beta})) \\ &= \boldsymbol{y}^T(I - \Pi)\boldsymbol{y} + (\hat{\boldsymbol{\beta}}_{\text{ols}} - \boldsymbol{\beta})^T\Pi(\hat{\boldsymbol{\beta}}_{\text{ols}} - \boldsymbol{\beta}),\end{aligned}$$

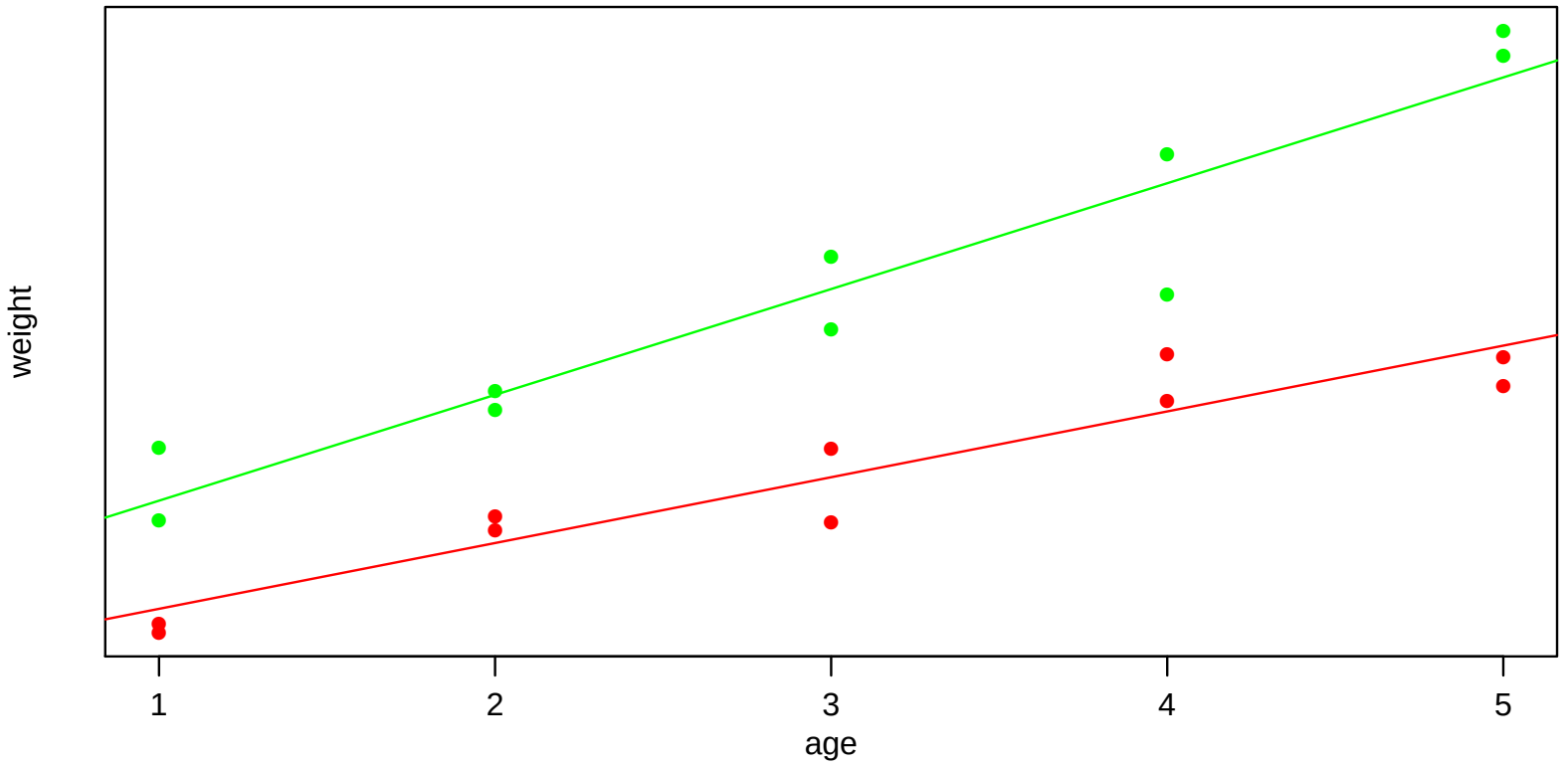
because all the 'cross terms' with Π and $I - \Pi$ are zero.

Hence the value of $\boldsymbol{\beta}$ that minimizes the SSR – for a given set of data – is $\hat{\boldsymbol{\beta}}_{\text{ols}}$.

OLS: using R

```
### OLS estimate
> beta.ols <- solve( t(X)%*%X )%*%t(X)%*%y
> beta.ols
                    [,1]
(Intercept) -0.06821632
xg           2.94485495
xa           2.84420729
xg:xa        1.72947648

### or less painfully, using lm()
> fit.ols<-lm( y~ xg*xa )

> coef( summary(fit.ols) )
               Estimate Std. Error     t value     Pr(>|t|)
(Intercept) -0.06821632  1.4222970 -0.04796208 9.623401e-01
xg           2.94485495  2.0114316  1.46405917 1.625482e-01
xa           2.84420729  0.4288387  6.63234803 5.760923e-06
xg:xa        1.72947648  0.6064695  2.85171239 1.154001e-02
```

OLS: using R




```
> coef( summary(fit.ols) )
               Estimate Std. Error     t value     Pr(>|t|)
(Intercept) -0.06821632  1.4222970 -0.04796208 9.623401e-01
xg           2.94485495  2.0114316  1.46405917 1.625482e-01
xa           2.84420729  0.4288387  6.63234803 5.760923e-06
xg:xa        1.72947648  0.6064695  2.85171239 1.154001e-02
```

Bayesian inference for regression models

$$y_i = \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p} + \epsilon_i$$

Motivation:

- Incorporating prior information
- Posterior probability statements: $\mathbb{P}[\beta_j > 0 | \boldsymbol{y}, \mathbf{X}]$
- OLS tends to overfit when p is large, Bayes' use of prior tends to make it more conservative – as we'll see
- Model selection and averaging (more later)

Prior and posterior distribution

$$
\begin{array}{lrcl}
\text{prior} & \boldsymbol{\beta} & \sim & \text{mvn}(\boldsymbol{\beta}_0, \Sigma_0) \\
\text{sampling model} & \boldsymbol{y} & \sim & \text{mvn}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}) \\
\text{posterior} & \boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X} & \sim & \text{mvn}(\boldsymbol{\beta}_n, \Sigma_n)
\end{array}
$$

where

$$
\begin{aligned}
\Sigma_n = \text{Var}\boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X}, \sigma^2 &= (\Sigma_0^{-1} + \mathbf{X}^T\mathbf{X}/\sigma^2)^{-1} \\
\boldsymbol{\beta}_n = \mathbb{E}\boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X}, \sigma^2 &= (\Sigma_0^{-1} + \mathbf{X}^T\mathbf{X}/\sigma^2)^{-1}(\Sigma_0^{-1}\boldsymbol{\beta}_0 + \mathbf{X}^T\boldsymbol{y}/\sigma^2).
\end{aligned}
$$

Notice:

- If $\Sigma_0^{-1} \ll \mathbf{X}^T\mathbf{X}/\sigma^2$, then $\boldsymbol{\beta}_n \approx \hat{\boldsymbol{\beta}}_{\text{ols}}$
- If $\Sigma_0^{-1} \gg \mathbf{X}^T\mathbf{X}/\sigma^2$, then $\boldsymbol{\beta}_n \approx \boldsymbol{\beta}_0$

The g-prior

How to pick β_0, Σ_0? A classical suggestion (Zellner, 1986) uses the **g-prior:**

$$\beta \sim \text{mvn}(\mathbf{0}, g\sigma^2(\mathbf{X}^T\mathbf{X})^{-1})$$

Idea: The variance of the OLS estimate $\hat{\beta}_{\text{ols}}$ is

$$\text{Var}\hat{\beta}_{\text{ols}} = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1} = \frac{\sigma^2}{n}(\mathbf{X}^T\mathbf{X}/n)^{-1}$$

This is (roughly) the uncertainty in β from n observations.

$$\text{Var}\beta_{\text{gprior}} = g\sigma^2(\mathbf{X}^T\mathbf{X})^{-1} = \frac{\sigma^2}{n/g}(\mathbf{X}^T\mathbf{X}/n)^{-1}$$

The g-prior can roughly be viewed as the uncertainty from n/g observations.

For example, $g = n$ means the prior has the same amount of info as 1 observation – so (roughly!) not much, in a large study.

Posterior distributions under the g-prior

After some algebra, it turns out that

$$\{\boldsymbol{\beta}|\boldsymbol{y},\mathbf{X},\sigma^2\} \sim \text{mvn}(\boldsymbol{\beta}_n,\Sigma_n),$$

$$\begin{aligned}\text{where } \Sigma_n = \text{Var}[\boldsymbol{\beta}|\boldsymbol{y},\mathbf{X},\sigma^2] &= \frac{g}{g+1}\sigma^2(\mathbf{X}^T\mathbf{X})^{-1} \\ \boldsymbol{\beta}_n = \mathbb{E}[\boldsymbol{\beta}|\boldsymbol{y},\mathbf{X},\sigma^2] &= \frac{g}{g+1}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y}\end{aligned}$$

Notes:

- The posterior mean estimate $\boldsymbol{\beta}_n$ is simply $\frac{g}{g+1}\hat{\boldsymbol{\beta}}_{\text{ols}}$.
- The posterior variance of $\boldsymbol{\beta}$ is simply $\frac{g}{g+1}\text{Var}\hat{\boldsymbol{\beta}}_{\text{ols}}$.
- g shrinks the coefficients towards $\mathbf{0}$ and can prevent overfitting to the data
- If $g = n$, then as n increases, inference approximates that using $\hat{\boldsymbol{\beta}}_{\text{ols}}$.

Monte Carlo simulation

What about the error variance σ^2? It's rarely known exactly, so more realistic to allow for uncertainty with a prior...

prior	$1/\sigma^2$	$\sim$	$\text{gamma}(\nu_0/2, \nu_0\sigma_0^2/2)$
sampling model	$\boldsymbol{y}$	$\sim$	$\text{mvn}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I})$
posterior	$1/\sigma^2\|\boldsymbol{y}, \mathbf{X}$	$\sim$	$\text{gamma}([\nu_0 + n]/2, [\nu_0\sigma_0^2 + \text{SSR}_g]/2)$

where SSR_g is somewhat complicated.

Simulating the joint posterior distribution:

joint distribution	$p(\sigma^2, \boldsymbol{\beta}\|\boldsymbol{y}, \mathbf{X})$	$=$	$p(\sigma^2\|\boldsymbol{y}, \mathbf{X}) \times p(\boldsymbol{\beta}\|\boldsymbol{y}, \mathbf{X}, \sigma^2)$
simulation	$\{\sigma^2, \boldsymbol{\beta}\} \sim p(\sigma^2, \boldsymbol{\beta}\|\boldsymbol{y}, \mathbf{X})$	$\Leftrightarrow$	$\sigma^2 \sim p(\sigma^2\|\boldsymbol{y}, \mathbf{X}), \boldsymbol{\beta} \sim p(\boldsymbol{\beta}\|\boldsymbol{y}, \mathbf{X}, \sigma^2)$

To simulate $\{\sigma^2, \boldsymbol{\beta}\} \sim p(\sigma^2, \boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X})$,

- First simulate σ^2 from $p(\sigma^2|\boldsymbol{y}, \mathbf{X})$
- Use this σ^2 to simulate $\boldsymbol{\beta}$ from $p(\boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X}, \sigma^2)$

Repeat 1000's of times to obtain MC samples: $\{\sigma^2, \boldsymbol{\beta}\}^{(1)}, \ldots, \{\sigma^2, \boldsymbol{\beta}\}^{(S)}$.

FTO example: Bayes

Priors:

$$\begin{aligned} 1/\sigma^2 &\sim \text{gamma}(1/2, 3.678/2) \\ \boldsymbol{\beta}|\sigma^2 &\sim \text{mvn}(\mathbf{0}, g \times \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}) \end{aligned}$$

Posteriors:

$$\begin{aligned} \{1/\sigma^2|\boldsymbol{y}, \mathbf{X}\} &\sim \text{gamma}((1+20)/2, (3.678+251.775)/2) \\ \{\boldsymbol{\beta}|\boldsymbol{y}, \mathbf{X}, \sigma^2\} &\sim \text{mvn}(.952 \times \hat{\boldsymbol{\beta}}_{\text{ols}}, .952 \times \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}) \end{aligned}$$

where

$$(\mathbf{X}^T\mathbf{X})^{-1} = \begin{pmatrix} 0.55 & -0.55 & -0.15 & 0.15 \\ -0.55 & 1.10 & 0.15 & -0.30 \\ -0.15 & 0.15 & 0.05 & -0.05 \\ 0.15 & -0.30 & -0.05 & 0.10 \end{pmatrix} \quad \hat{\boldsymbol{\beta}}_{\text{ols}} = \begin{pmatrix} -0.068 \\ 2.945 \\ 2.844 \\ 1.729 \end{pmatrix}$$

FTO example: Bayes in R

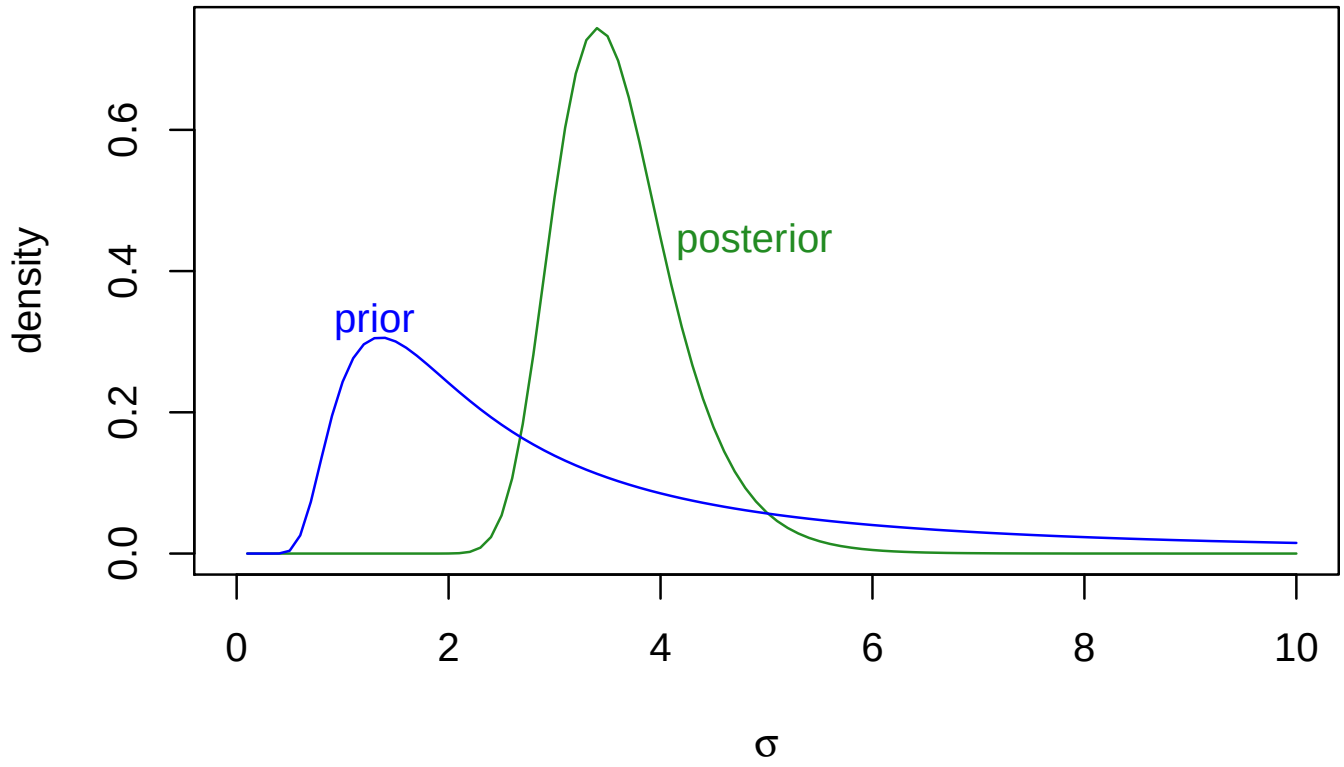
```
## data dimensions
n <- nrow(X)
p <- ncol(X)

## prior parameters
nu0 <- 1
s20 <- summary(lm(y~-1+X))$sigma^2
g <- n

## posterior calculations
Hg   <- (g/(g+1)) * X%*%solve(t(X)%*%X)%*%t(X)
SSRg <- t(y)%*%( diag(1,nrow=n)  - Hg ) %*%y

Vbeta <- g*solve(t(X)%*%X)/(g+1)
Ebeta <- Vbeta%*%t(X)%*%y

## simulate sigma^2  and beta
## may need to install the mvtnorm package, for rmvnorm()
library("mvtnorm")
set.seed(4)
s2.post <- 1/rgamma(5000, (nu0+n)/2, (nu0*s20+SSRg)/2 )
beta.post <- t( sapply(s2.post, function(s2val){
                  rmvnorm(1, Ebeta, s2val*Vbeta)}
     ) )
```

FTO example: Bayes in R

```
> # look at the first few samples
> s2.post[1:5]
[1] 11.940216 15.281855 15.821894  8.062999 10.385588
> beta.post[1:5,]
[1,] -0.4473916 4.1394091 2.918340 0.8798302
[2,]  2.1962233 0.9645979 1.444978 2.2662295
[3,]  1.6564342 0.4438559 2.907225 1.5375880
[4,]  0.3876347 1.0801567 2.357214 2.5752063
[5,] -0.4976167 3.2976243 2.004742 2.5915040
> # posterior quantiles for betas
> t(apply(beta.post,2,quantile,probs=c(.5,.025,.975)))
              50%       2.5%     97.5%
[1,] -0.06185524 -5.2449633  5.203937
[2,]  2.75532364 -4.4390185 10.172409
[3,]  2.71534342  1.1521791  4.293707
[4,]  1.63590630 -0.6553404  3.840388
> # compare with the OLS estimates
> cbind(coef(fit.ols), confint.default(fit.ols))
                                2.5 %   97.5 %
(Intercept) -0.06821632 -2.8558671 2.719434
xg           2.94485495 -0.9974786 6.887189
xa           2.84420729  2.0036989 3.684716
xg:xa        1.72947648  0.5408182 2.918135
```

FTO example: Bayes in R

Why are the estimates similar but the intervals so different? Here are the prior and posterior for σ;



The 'best guess' estimate is $\hat{\sigma} = \sigma_0 = 1.91$ – but the prior also supports much larger values – with which the data don't strongly disagree.

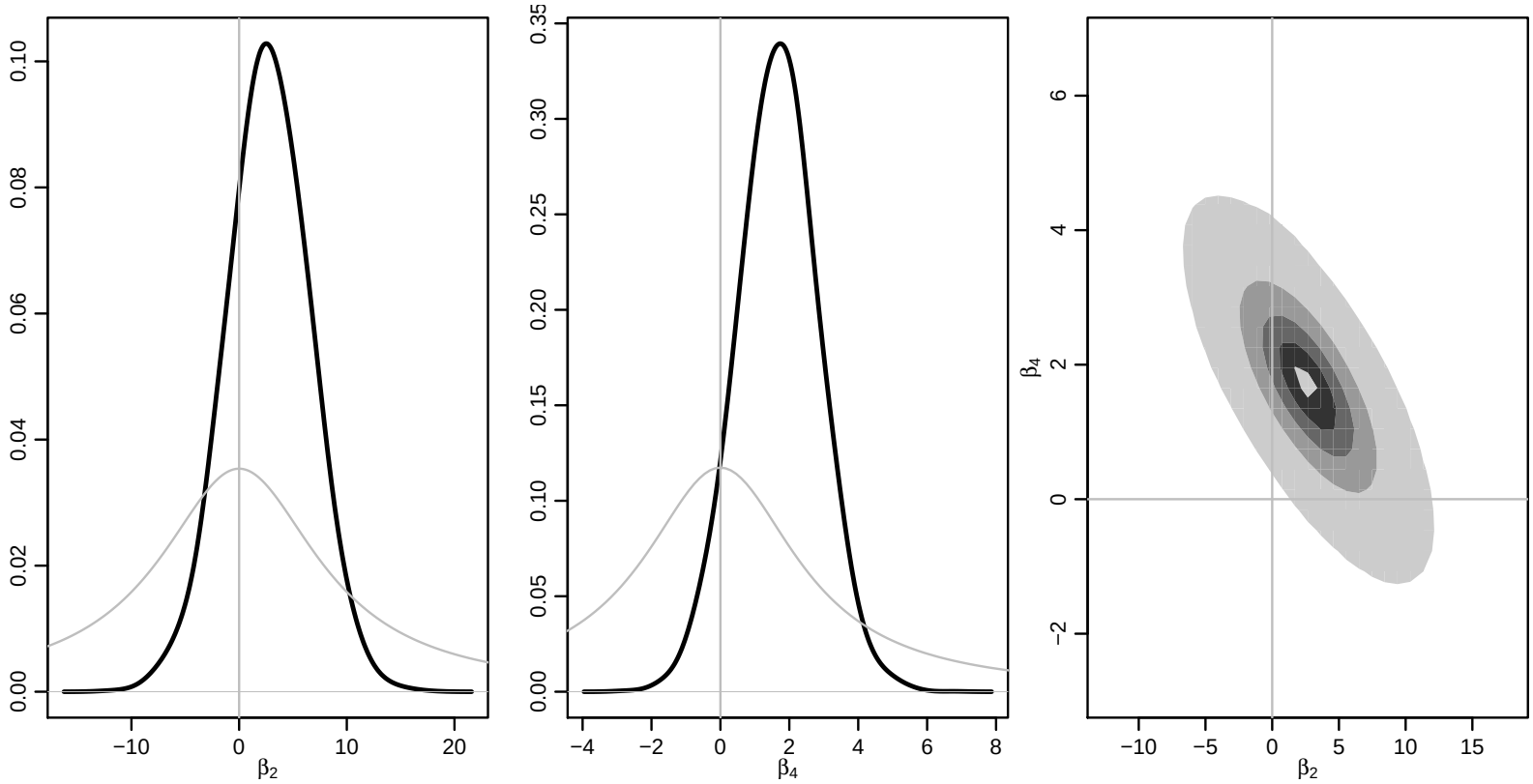
FTO example: Bayes in R

Numerical confirmation of the same thing; the posterior quantiles for σ^2, and σ.

```
> quantile(s2.post,probs=c(.025,.5,.975))
     2.5%       50%     97.5%
 7.244054 12.613746 24.430451
> quantile(sqrt(s2.post),probs=c(.025,.5,.975))
    2.5%      50%    97.5%
2.691478 3.551584 4.942717
```

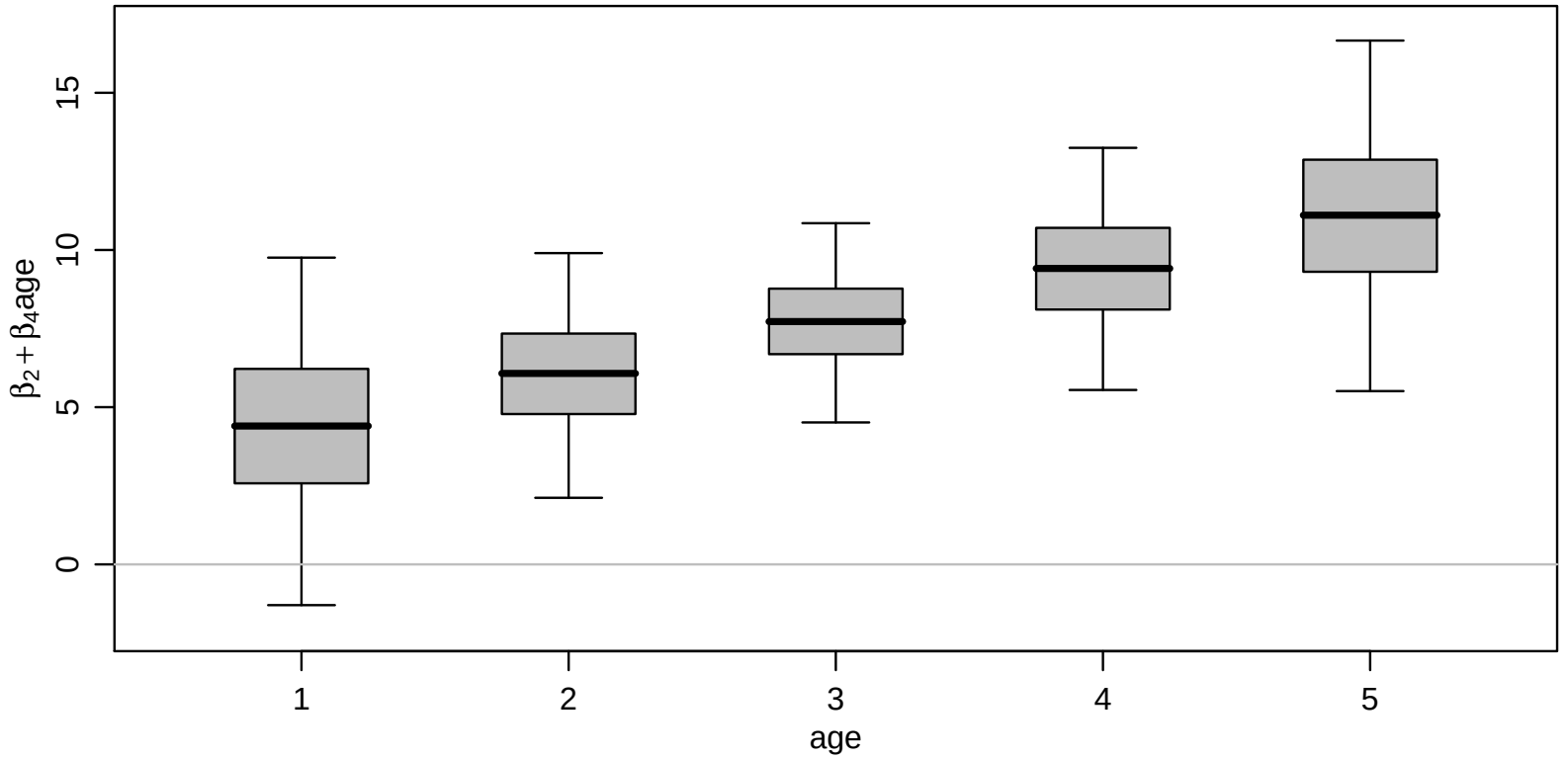
Can examine the posterior for any quantity of interest in the same straightforward manner, given a large sample from the posterior.

Posterior distributions



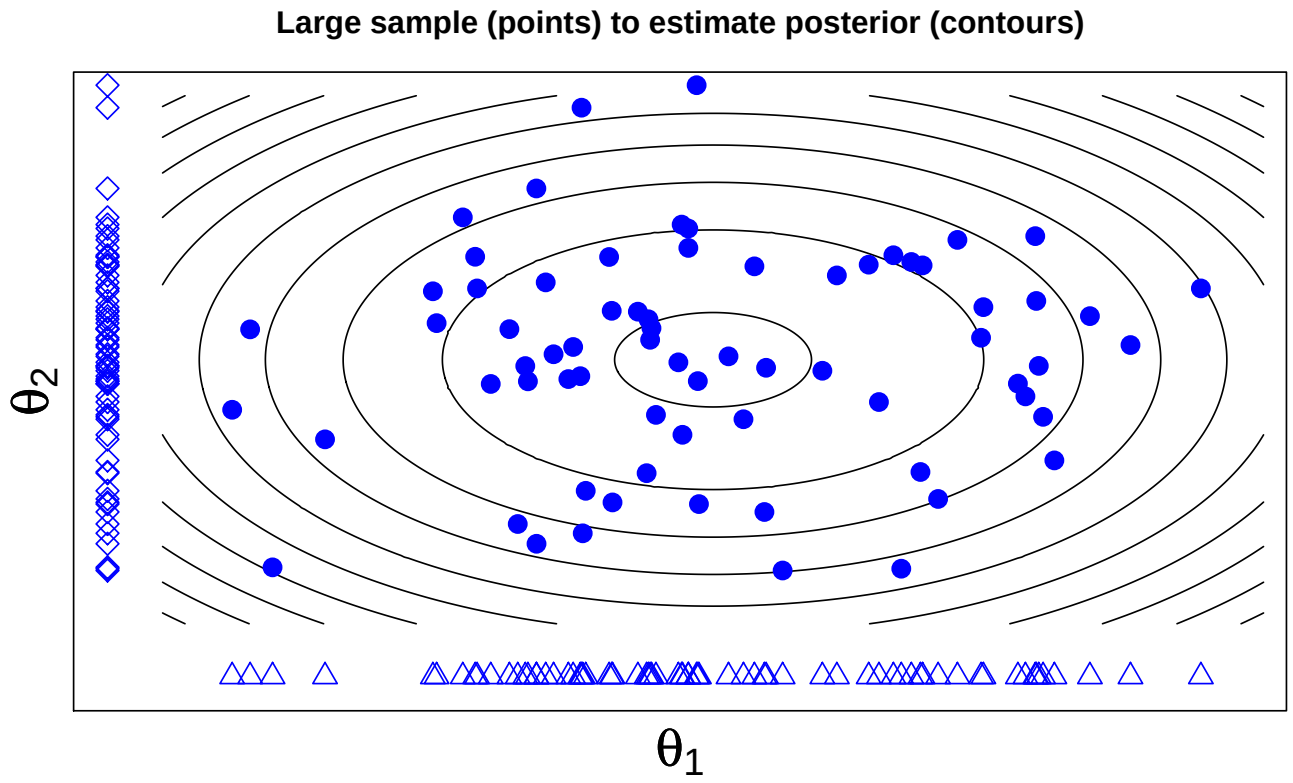
Summarizing the genetic effect

$$
\begin{aligned}
\text{Genetic effect} &= \mathbb{E}[\,y|age, +/-\,] - \mathbb{E}[\,y|age, -/-\,] \\
&= [(\beta_1 + \beta_2) + (\beta_3 + \beta_4) \times \text{age}] - [\beta_1 + \beta_3 \times \text{age}] \\
&= \beta_2 + \beta_4 \times \text{age}
\end{aligned}
$$



Numerical approximations

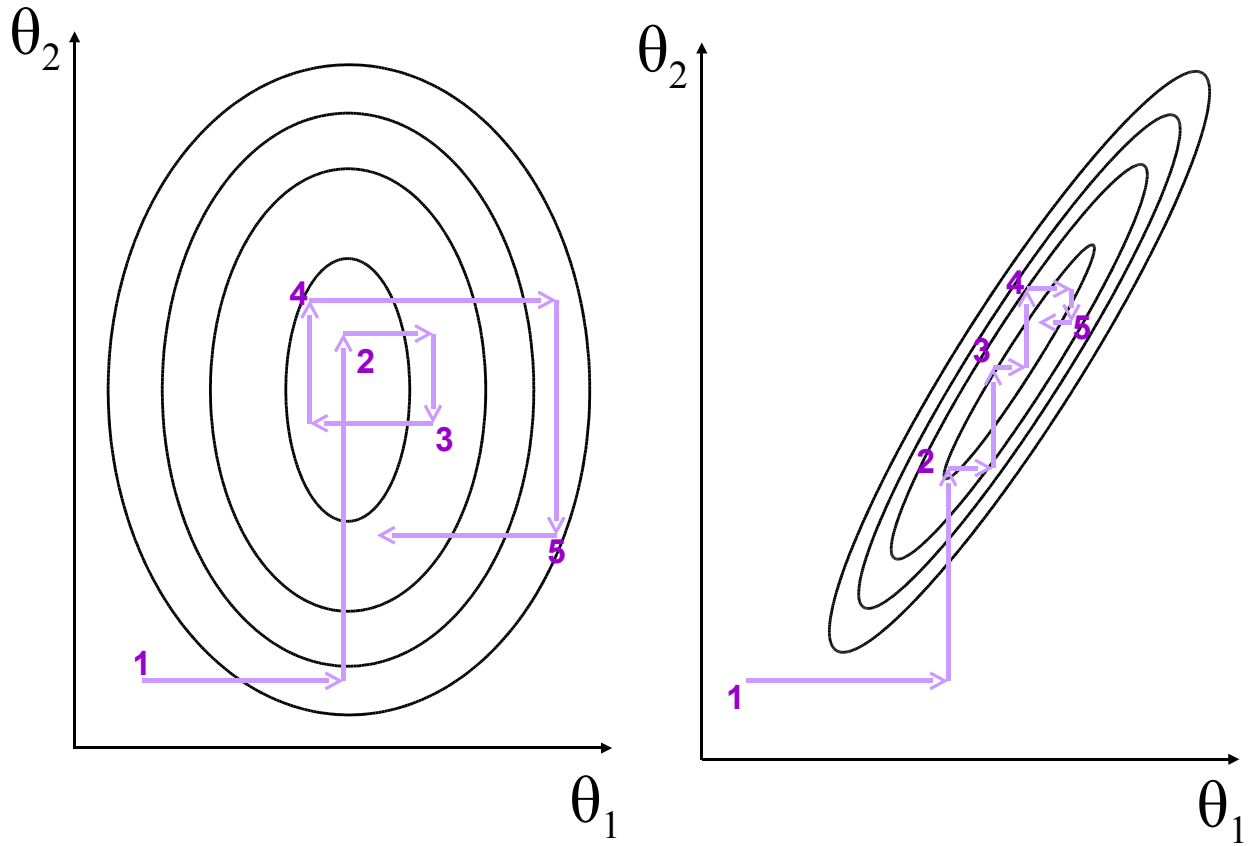
The *Monte Carlo method* is important for Bayesian work;



With a large sample from some distribution – i.e. the posterior – we can approximate any property of that distribution. It **does not matter** if how we get the sample.

Numerical approximations

Here's one way; (the *Gibbs Sampler* – in two examples)



Gibbs updates parameters 'one at a time', using $\pi(\theta_1|\theta_2)$, then $\pi(\theta_2|\theta_1)$. The sequence of samples $\boldsymbol{\theta}^{(s)}$ (a *Markov Chain*) **are** dependent, but the posterior **is** covered appropriately.

Numerical approximations

Algebra for the same thing; the posterior is

$$\pi(\theta_1, \theta_2|\mathbf{Y}) \propto f(\mathbf{Y}|\theta_1, \theta_2) \times \pi(\theta_1, \theta_2),$$

and is usually intractable. But it is equivalent to

$$\pi(\theta_1, \theta_2|\mathbf{Y}) = p(\theta_2|\mathbf{Y}) \times p(\theta_1|\theta_2, \mathbf{Y}),$$

and conditional $p(\theta_1|\theta_2, \mathbf{Y})$ may be more readily-available.

Gibbs uses *just* the conditionals, iterating between these steps:

$$\begin{aligned} \theta_1^{(s)} &\sim p(\theta_1|\theta_2^{(s-1)}, \mathbf{Y}) \\ \theta_2^{(s)} &\sim p(\theta_2|\theta_1^{(s)}\ \ , \mathbf{Y}) \end{aligned}$$

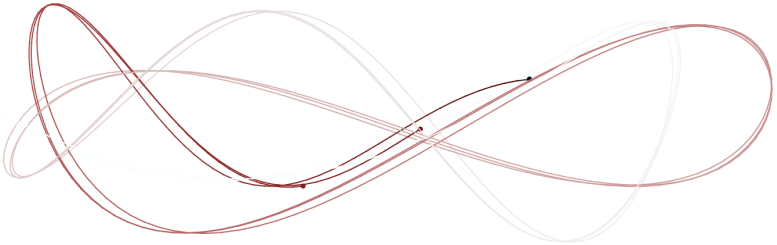
to produce the sequence

$$(\theta_1^{(0)}, \theta_2^{(0)}), (\theta_1^{(1)}, \theta_2^{(1)}), ..., (\theta_1^{(s)}, \theta_2^{(s)}), ...$$

- If the run is long enough ($s \to \infty$), this sequence *is* a sample from $\pi(\theta_1, \theta_2|\mathbf{Y})$, no matter where you started
- For more parameters, update each θ_k in turn, then start again

MCMC: with Stan

Stan does all these steps for us;



- Specify a model, including priors, and tell Stan what the data are
- Stan writes code to sample from the posterior, by 'walking around' – actually it runs the *No U-Turn Sampler*, a more
- Stan runs this code, and reports back all the samples
- The `rstan` package lets you run chains from R
- Some modeling limitations – no discrete parameters – but becoming very popular; works well with some models where other software would struggle
- Requires declarations (like C++) – unlike R – so models require a bit more typing...

MCMC: with Stan

A first attempt at the FTO example, running Stan from within R;

```
cat(file="FTOexample.stan", "
data {
  int n; //the number of observations
  int p; //the number of columns in the model matrix
  real y[n]; //the response
  matrix[n,p] X; //the model matrix
  real g; // Zellner scale factor
  vector[p] mu; // Zellner prior mean (all zeros)
  matrix[p,p] XtXinv; // information matrix
  real sigma; // std dev'n, assumed known
}
parameters {
  vector[p] beta; //the regression parameters
}
transformed parameters {
  vector[n] linpred;
  cov_matrix[p] Sigma;
  linpred = X*beta;
  for (j in 1:p){
    for (k in 1:p){
```

MCMC: with Stan

```
      Sigma[j,k] = g*sigma^2*XtXinv[j,k];
      }
  }
}
model {
  beta ~ multi_normal(mu, Sigma);
  y ~ normal(linpred, sigma);
}
")

# do the MCMC, store the results
library("rstan")
stan1 <- stan(file = "FTOexample.stan", data = list(
   n=n,p=p, y=y, X=X, g=n, mu=rep(0,p), XtXinv=solve(crossprod(X)),
   sigma=sqrt(s20)
   ), iter = 100000, chains = 1, pars="beta")
```

- Most of the work involves writing a model file (though our 'model' is only two lines!)
- We use 100,000 iterations – which takes only a few seconds

MCMC: with Stan

What it produces, and how it compares to exact calculation;

```
> print(stan1)
Inference for Stan model: FTOexample.
1 chains, each with iter=1e+05; warmup=50000; thin=1;
post-warmup draws per chain=50000, total post-warmup draws=50000.
          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
beta[1]  -0.07    0.01 1.39  -2.86  -0.99  -0.06   0.86   2.62 12762    1
beta[2]   2.82    0.02 1.98  -1.02   1.47   2.79   4.13   6.76 12738    1
beta[3]   2.71    0.00 0.42   1.90   2.43   2.71   2.99   3.55 12838    1
beta[4]   1.64    0.01 0.60   0.46   1.24   1.65   2.05   2.80 12721    1
lp__    -39.55    0.01 1.43 -43.13 -40.25 -39.22 -38.49 -37.77 16421    1

Samples were drawn using NUTS(diag_e) at Tue Aug 29 16:15:18 2017.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
> round( cbind(Ebeta, sqrt(diag(Vbeta*s20))), 2)
             [,1] [,2]
(Intercept) -0.06 1.39
xg           2.80 1.96
xa           2.71 0.42
xg:xa        1.65 0.59
```

MCMC: with Stan

Now the full version, with the inverse-gamma prior on σ^2;

```
cat(file="FTOexample.stan", "
data {
  int n; //the number of observations
  int p; //the number of columns in the model matrix
  real y[n]; //the response
  matrix[n,p] X; //the model matrix
  real g; // Zellner scale factor
  vector[p] mu; // Zellner prior mean (all zeros)
  matrix[p,p] XtXinv; // information matrix
}
parameters {
  vector[p] beta; //the regression parameters
  real invsigma2; //the standard deviation
}
transformed parameters {
  vector[n] linpred;
  cov_matrix[p] Sigma;
  real sigma;
  linpred = X*beta;
  sigma = 1/sqrt(invsigma2);
  for (j in 1:p){
```

MCMC: with Stan

```
    for (k in 1:p){
      Sigma[j,k] = g*sigma^2*XtXinv[j,k];
      }
  }
}
model {
  beta ~ multi_normal(mu, Sigma);
  y ~ normal(linpred, sigma);
  invsigma2 ~ gamma(0.5, 1.839);
}
")

# do the MCMC, store the results
library("rstan")
stan2 <- stan(file = "FTOexample.stan",
data = list(n=n,p=p, y=y, X=X, g=n, mu=rep(0,p), XtXinv=solve(crossprod(X)) ),
iter = 100000, chains = 1, pars=c("beta","sigma"))
```

MCMC: with Stan

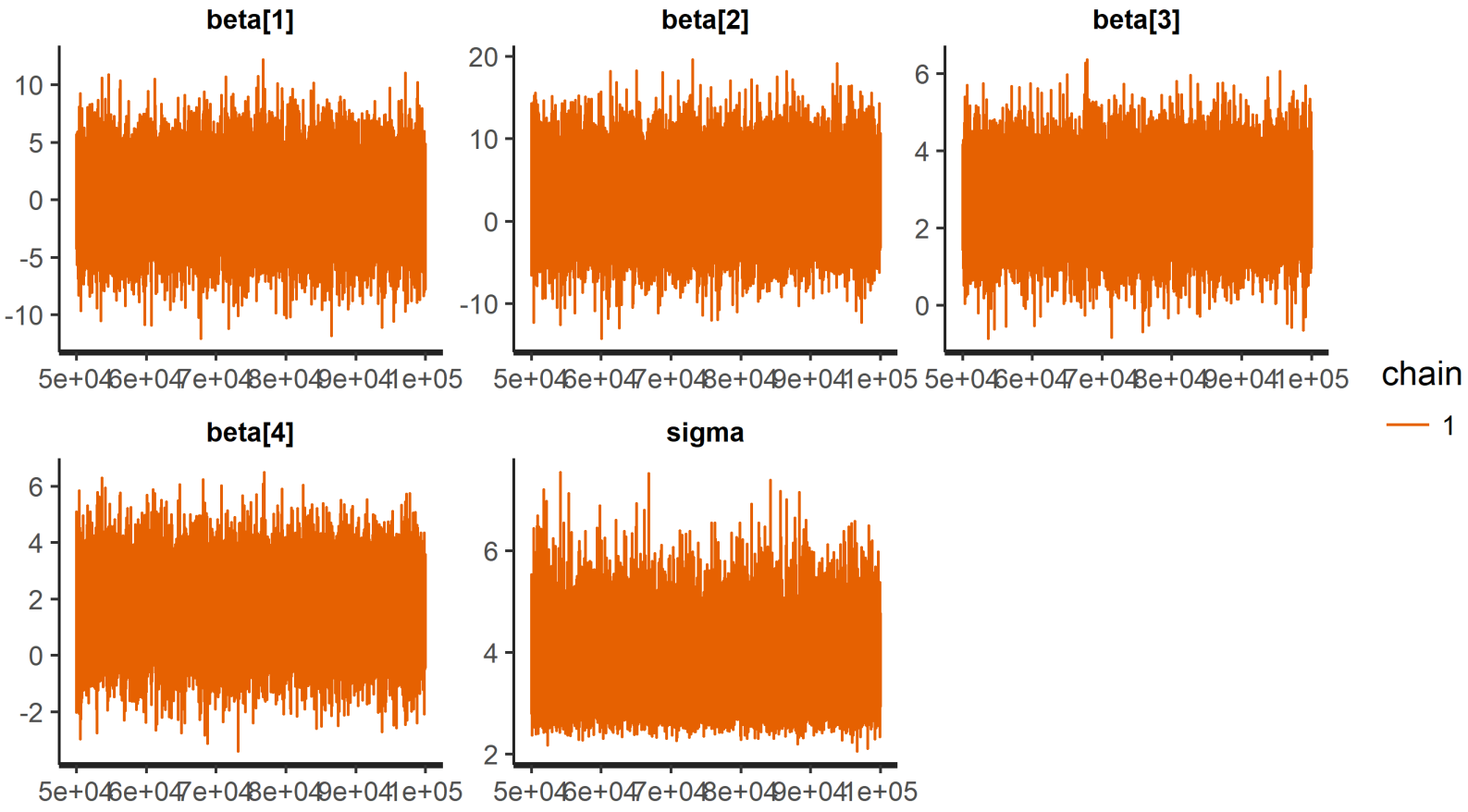
And comparing the output again;

```
> print(stan2)
Inference for Stan model: FTOexample.
1 chains, each with iter=1e+05; warmup=50000; thin=1;
post-warmup draws per chain=50000, total post-warmup draws=50000.

          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
beta[1]  -0.09    0.02 2.66  -5.32  -1.85  -0.08   1.65   5.13 15022    1
beta[2]   2.84    0.03 3.73  -4.46   0.38   2.84   5.29  10.25 14829    1
beta[3]   2.72    0.01 0.80   1.14   2.19   2.71   3.25   4.29 15065    1
beta[4]   1.64    0.01 1.12  -0.60   0.90   1.63   2.38   3.85 14780    1
sigma     3.62    0.00 0.59   2.69   3.20   3.55   3.95   4.97 18325    1
lp__    -42.49    0.01 1.65 -46.59 -43.34 -42.15 -41.27 -40.32 15016    1

> round(t(apply( cbind(beta.post, sqrt(s2.post)), 2,
+ function(x){c(m=mean(x), sd=sd(x), quantile(x, c(0.025, 0.5, 0.975)))})), 2)
         m   sd  2.5%   50% 97.5%
[1,] -0.05 2.57 -5.14 -0.04  5.07
[2,]  2.77 3.62 -4.38  2.76  9.90
[3,]  2.70 0.77  1.17  2.70  4.23
[4,]  1.66 1.09 -0.50  1.66  3.83
[5,]  3.50 0.54  2.63  3.43  4.74
```

MCMC: with Stan

To see where the the 'chain' went...

```
> traceplot(stan2)
```



MCMC: summary so far

- Stan - and other similar software - may look like overkill for this 'conjugate' problem, but Stan can provide posteriors for almost any model
- The 'modeling' language is modeled on R
- Users do have to decide how long a chain to run, and how long to 'burn in' for at the start of the chain. These are not easy to answer! We'll see some diagnostics in later sessions

Bonus: what if the model's wrong?

Different types of violation—in decreasing order of how much they typically matter in practice

- Just have the wrong data (!) i.e. not the data you claim to have
- Observations are not independent, e.g. repeated measures on same mouse over time
- Mean model is incorrect
- Error terms do not have constant variance
- Error terms are not Normally distributed

Wrong model: dependent outcomes

- Observations from the same mouse are more likely to be similar than those from different mice (even if they have same age and genotype)
- SBP from subjects (even with same age, genotype etc) in the same family are more likely to be similar than those in different familes – perhaps unmeasured common diet?
- Spatial and temporal relationships also tend to induce correlation

If the pattern of relationship is known, can allow for it – typically in "random effects modes" – see later session.

If not, treat results with caution! Precision is likely over-stated.

Wrong model: mean model

Even when the scientific background is highly informative about the variables of interest (e.g. we want to know about the association of Y with $\boldsymbol{x}_1$, adjusting for $\boldsymbol{x}_2$, $\boldsymbol{x}_3$...) there is rarely strong information about the form of the model

- Does mean weight increase with age? age^2? age^3?
- Could the effect of genotype also change non-linearly with age?

Including quadratic terms is a common approach – but quadratics are sensitive to the tails. Instead, including "spline" representations of covariates allows the model to capture many patterns.

Including interaction terms (as we did with $x_{i,2} \times x_{i,3}$) lets one covariate's effect vary with another.

(Deciding which covariates to use is addressed in the Model Choice session.)

Wrong model: non-constant variance

This is plausible in many situations; perhaps e.g. young mice are harder to measure, i.e. more variables. Or perhaps the FTO variant affects weight regulation — again, more variance.

- Having different variances at different covariate values is known as *heteroskedasticity*
- Unaddressed, it can result in over- or under-statement of precision

The most obvious approach is to model the variance, i.e.

$$\begin{aligned} Y_i &= \boldsymbol{\beta}^T \boldsymbol{x}_i + \epsilon_i, \\ \epsilon_i &\sim \text{Normal}(0, \sigma_i^2), \end{aligned}$$

where σ_i depends on covariates, e.g. σ_{homozy} and $\sigma_{heterozy}$ for the two genotypes.

Of course, these parameters need priors. Constraining variances to be positive also makes choosing a model difficult in practice.

Robust standard errors (in Bayes)

In linear regression, some robustness to model-misspecification and/or non-constant variance is available – but it relies on interest in linear 'trends'. Formally, we can define parameter

$$\boldsymbol{\theta} = \text{argmin } \mathbb{E}_x \left[\left(\mathbb{E}_y[y|x] - \mathbf{X}^t \boldsymbol{\theta} \right)^2 \right],$$

i.e. the straight line that best-captures random-sampling, in a least-squares sense.

- This 'trend' can capture important features in how the mean y varies at different x
- Fitting extremely flexible Bayesian models, we get a posterior for $\boldsymbol{\theta}$
- The posterior mean approaches $\widehat{\boldsymbol{\beta}}_{\text{ols}}$, in large samples
- The posterior variance approaches the 'robust' *sandwich estimate*, in large samples (details in Szpiro et al, 2011)

Robust standard errors (in Bayes)

The OLS estimator can be written as $\hat{\boldsymbol{\beta}}_{\text{ols}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y} = \sum_{i=1}^n c_i y_i$, for appropriate c_i.

$$\begin{aligned}
\textbf{True} \text{ variance} \quad \text{Var}[\,\hat{\beta}\,] &= \sum_{i=1}^n c_i^2 \text{Var}[\,Y_i\,] \\
\textbf{Robust} \text{ estimate} \quad \widehat{\text{Var}}_R[\,\hat{\beta}\,] &= \sum_{i=1}^n c_i^2 e_i^2 \\
\textbf{Model-based} \text{ estimate} \quad \widehat{\text{Var}}_M[\,\hat{\beta}\,] &= \text{Mean}(e_i^2) \sum_{i=1}^n c_i^2,
\end{aligned}$$

where $e_i = y_i - \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}_{\text{ols}}$, the residuals from fitting a linear model.

Non-Bayesian sandwich estimates are available through R's `sandwich` package – much quicker than Bayes with a very-flexible model. For correlated outcomes, see the `GEE` package for generalizations.

Wrong model: Non-Normality

This is not a big problem for learning about population parameters;

- The variance statements/estimates we just saw don't rely on Normality
- The *central limit theorem* means that $\hat{\beta}$ ends up Normal anyway, in large samples
- In small samples, expect to have limited power to detect non-Normality
- ... except, perhaps, for extreme outliers (data errors?)

For prediction – where we assume that outcomes do follow a Normal distibution – this assumption is more important.

Summary

- Linear regressions are of great applied interest
- Corresponding models are easy to fit, particularly with judicious prior choices
- Assumptions are made — but a well-chosen linear regression usually tells us **something** of interest, even if the assumptions are (mildly) incorrect