Linear (Gaussian) Models

Contents

1 Linear (Gaussian) Models

Configuring R

Functions from these packages will be used throughout this document:

```r
library(conflicted) # check for conflicting function definitions
# library(printr) # inserts help-file output into markdown output
library(rmarkdown) # Convert R Markdown documents into a variety of formats.
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggfortify) # help with graphics
library(dplyr) # manipulate data
library(tibble) # `tibble`s extend `data.frame`s
library(magrittr) # `%>%` and other additional piping tools
library(haven) # import Stata files
library(knitr) # format R output for markdown
library(tidyr) # Tools to help to create tidy data
library(plotly) # interactive graphics
library(dobson) # datasets from Dobson and Barnett 2018
library(parameters) # format model output tables for markdown
library(haven) # import Stata files
library(latex2exp) # use LaTeX in R code (for figures and tables)
library(fs) # filesystem path manipulations
library(survival) # survival analysis
library(survminer) # survival analysis graphics
library(KMsurv) # datasets from Klein and Moeschberger
library(parameters) # format model output tables for
library(webshot2) # convert interactive content to static for pdf
library(forcats) # functions for categorical variables ("factors")
library(stringr) # functions for dealing with strings
library(lubridate) # functions for dealing with dates and times
```

Here are some R settings I use in this document:

```r
rm(list = ls()) # delete any data that's already loaded into R

conflicts_prefer(dplyr::filter)
ggplot2::theme_set(
  ggplot2::theme_bw() +
        # ggplot2::labs(col = "") +
    ggplot2::theme(
      legend.position = "bottom",
      text = ggplot2::element_text(size = 12, family = "serif")))

knitr::opts_chunk$set(message = FALSE)
options('digits' = 6)

panderOptions("big.mark", ",")
pander::panderOptions("table.emphasize.rownames", FALSE)
pander::panderOptions("table.split.table", Inf)
conflicts_prefer(dplyr::filter) # use the `filter()` function from dplyr() by default
legend_text_size = 9
run_graphs = TRUE
```

```
include_reference_lines <- FALSE
```

Note

This content is adapted from:

- Dobson and Barnett (2018), Chapters 2-6
- Dunn and Smyth (2018), Chapters 2-3
- Vittinghoff et al. (2012), Chapter 4

There are numerous textbooks specifically for linear regression, including:

- Kutner et al. (2005): used for UCLA Biostatistics MS level linear models class
- Chatterjee and Hadi (2015): used for Stanford MS-level linear models class
- Seber and Lee (2012): used for UCLA Biostatistics PhD level linear models class and UC Davis STA 108.
- Kleinbaum et al. (2014): same first author as Kleinbaum and Klein (2010) and Kleinbaum and Klein (2012)
- *Linear Models with R* (Faraway 2025)
- *Applied Linear Regression* by Sanford Weisberg (Weisberg 2005)

For more recommendations, see the discussion on Reddit[a].

- see also https://web.stanford.edu/class/stats191 [1]

[a] https://www.reddit.com/r/statistics/comments/qwgctl/q_books_on_applied_linear_modelsregression_for/

1.1 Overview

1.1.1 Why this course includes linear regression

- This course is about *generalized linear models* (for non-Gaussian outcomes)
- UC Davis STA 108 ("Applied Statistical Methods: Regression Analysis") is a prerequisite for this course, so everyone here should have some understanding of linear regression already.
- We will review linear regression to:
- make sure everyone is caught up
- to provide an epidemiological perspective on model interpretation.

1.1.2 Chapter overview

- Section 1.2: how to interpret linear regression models
- Section 1.3: how to estimate linear regression models
- Section 1.4: how to quantify uncertainty about our estimates

[1] the current version of the first regression course I ever took

Table 1: `birthweight` data (Dobson and Barnett (2018) Example 2.2.2)

```
library(dobson)
data("birthweight", package = "dobson")
birthweight
#> # A tibble: 12 x 4
#>    `boys gestational age` `boys weight` `girls gestational age` `girls weight`
#>                     <dbl>         <dbl>                   <dbl>          <dbl>
#>  1                     40          2968                      40           3317
#>  2                     38          2795                      36           2729
#>  3                     40          3163                      40           2935
#>  4                     35          2925                      38           2754
#>  5                     36          2625                      42           3210
#>  6                     37          2847                      39           2817
#>  7                     41          3292                      40           3126
#>  8                     40          3473                      37           2539
#>  9                     37          2628                      36           2412
#> 10                     38          3176                      38           2991
#> 11                     40          3421                      39           2875
#> 12                     38          2975                      40           3231
```

- Section 1.8: how to tell if your model is insufficiently complex

1.2 Understanding Gaussian Linear Regression Models

1.2.1 Motivating example: birthweights and gestational age

Suppose we want to learn about the distributions of birthweights (*outcome* Y) for (human) babies born at different gestational ages (*covariate* A) and with different chromosomal sexes (*covariate* S) (Dobson and Barnett (2018) Example 2.2.2).

1.2.2 Dobson `birthweight` data

Data as table

Reshape data for graphing

Data as graph

```
plot1 <- bw |>
  ggplot(aes(
    x = age,
    y = weight,
    shape = sex,
    col = sex
  )) +
  theme_bw() +
  xlab("Gestational age (weeks)") +
  ylab("Birthweight (grams)") +
  theme(legend.position = "bottom") +
  # expand_limits(y = 0, x = 0) +
  geom_point(alpha = .7)
print(plot1 + facet_wrap(~sex))
```

Table 2: `birthweight` data reshaped

```
library(tidyverse)
bw <-
  birthweight |>
  pivot_longer(
    cols = everything(),
    names_to = c("sex", ".value"),
    names_sep = "s "
  ) |>
  rename(age = `gestational age`) |>
  mutate(
    id = row_number(),
    sex = sex |>
      case_match(
        "boy" ~ "male",
        "girl" ~ "female"
      ) |>
      factor(levels = c("female", "male")),
    male = sex == "male",
    female = sex == "female"
  )

bw
#> # A tibble: 24 x 6
#>    sex      age weight    id male  female
#>    <fct>  <dbl>  <dbl> <int> <lgl> <lgl>
#>  1 male      40   2968     1 TRUE  FALSE
#>  2 female    40   3317     2 FALSE TRUE
#>  3 male      38   2795     3 TRUE  FALSE
#>  4 female    36   2729     4 FALSE TRUE
#>  5 male      40   3163     5 TRUE  FALSE
#>  6 female    40   2935     6 FALSE TRUE
#>  7 male      35   2925     7 TRUE  FALSE
#>  8 female    38   2754     8 FALSE TRUE
#>  9 male      36   2625     9 TRUE  FALSE
#> 10 female    42   3210    10 FALSE TRUE
#> # i 14 more rows
```

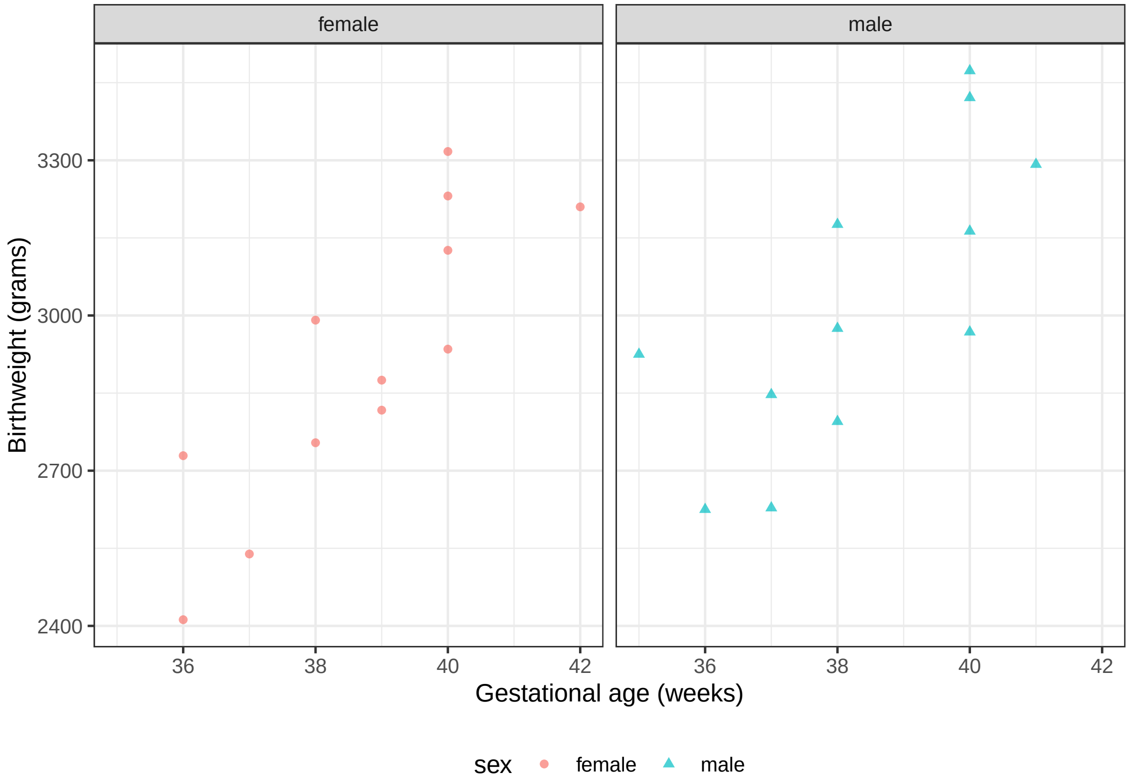


Figure 1: `birthweight` data (Dobson and Barnett (2018) Example 2.2.2)

Data notation

Let's define some notation to represent this data:

- Y: birthweight (measured in grams)
- S: chromosomal sex: "male" (XY) or "female" (XX)
- M: indicator variable for $S =$ "male"[2]
- $M = 0$ if $S =$ "female"
- $M = 1$ if $S =$ "male"
- F: indicator variable for $S =$ "female"[3]
- $F = 1$ if $S =$ "female"
- $F = 0$ if $S =$ "male"
- A: estimated gestational age at birth (measured in weeks).

Female is the **reference level** for the categorical variable S (chromosomal sex) and corresponding indicator variable M . The choice of a reference level is arbitrary and does not limit what we can do with the resulting model; it only makes it more computationally convenient to make inferences about comparisons involving that reference group.

M and F are called **dummy variables**; together, they are a numeric representation of the categorical variable S. Dummy variables with values 0 and 1 are also called **indicator variables**. There are other ways to construct dummy variables, such as using the values -1 and 1 (see Dobson and Barnett (2018) §2.4 for details).

[2] M is implicitly a deterministic function of S

[3] F is implicitly a deterministic function of S

1.2.3 Parallel lines regression

(c.f. Dunn and Smyth (2018) §2.10.3[4])

We don't have enough data to model the distribution of birth weight separately for each combination of gestational age and sex, so let's instead consider a (relatively) simple model for how that distribution varies with gestational age and sex:

$$\begin{aligned} Y|M,A \ &\sim_{\text{ciid}} \ N(\mu(M,A),\sigma^2) \\ \mu(m,a) &= \beta_0 + \beta_M m + \beta_A a \end{aligned} \tag{1}$$

Table 3 shows the parameter estimates from R. Figure 2 shows the estimated model, superimposed on the data.

```r
bw_lm1 <- lm(
  formula = weight ~ sex + age,
  data = bw
)

library(parameters)
bw_lm1 |>
  parameters::parameters() |>
  parameters::print_md(
    include_reference = include_reference_lines,
    select = "{estimate}"
  )
```

Table 3: Regression parameter estimates for Model 1 of `birthweight` data

Parameter	Coefficient
(Intercept)	-1773.32
sex (male)	163.04
age	120.89

[4] https://link.springer.com/chapter/10.1007/978-1-4419-0118-7_2#Sec31

```r
bw <-
  bw |>
  mutate(`E[Y|X=x]` = fitted(bw_lm1)) |>
  arrange(sex, age)

plot2 <-
  plot1 %+% bw +
  geom_line(aes(y = `E[Y|X=x]`))

print(plot2)
```

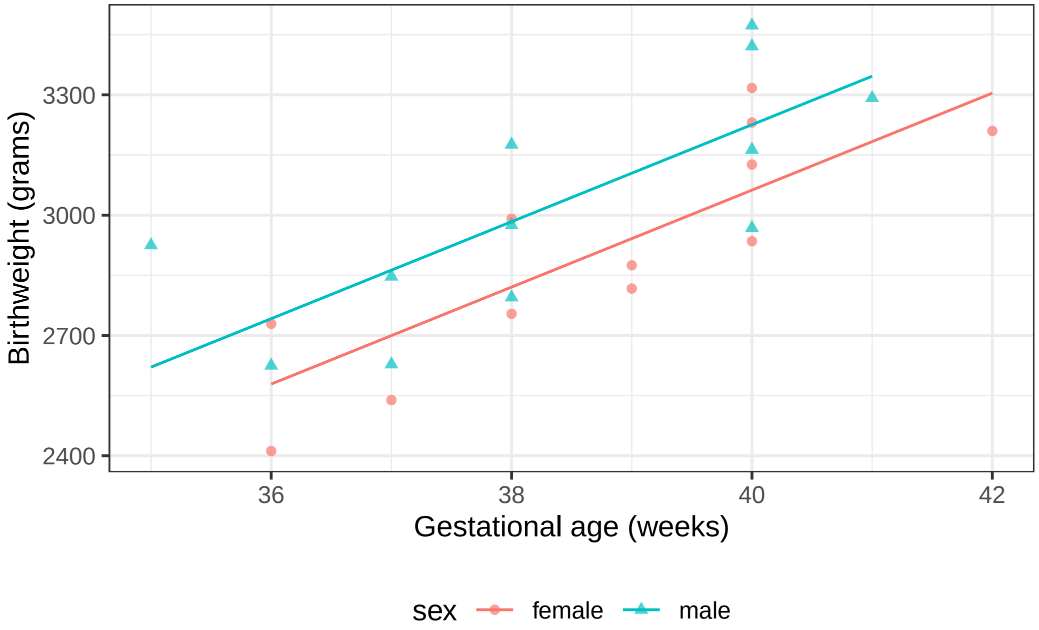


Figure 2: Graph of Model 1 for `birthweight` data

Model assumptions and predictions

To learn what this model is assuming, let's plug in a few values.

Exercise 1.1. What's the mean birthweight for a female born at 36 weeks?

Table 4: Estimated coefficients for model 1

Parameter	Coefficient
(Intercept)	-1773.32
sex (male)	163.04
age	120.89

Solution.

Table 5: Estimated coefficients for model 1

Parameter	Coefficient
(Intercept)	-1773.32
sex (male)	163.04
age	120.89

```r
pred_female <- coef(bw_lm1)["(Intercept)"] + coef(bw_lm1)["age"] * 36
### or using built-in prediction:
pred_female_alt <- predict(bw_lm1, newdata = tibble(sex = "female", age = 36))
```

$$\begin{aligned} E[Y|M=0,A=36] &= \beta_0 + (\beta_M \cdot 0) + (\beta_A \cdot 36) \\ &= -1773.321839 + (163.039303 \cdot 0) + (120.894327 \cdot 36) \\ &= 2578.873934 \end{aligned}$$

Exercise 1.2. What's the mean birthweight for a male born at 36 weeks?

Table 6: Estimated coefficients for model 1

Parameter	Coefficient
(Intercept)	-1773.32
sex (male)	163.04
age	120.89

Solution.

Table 7: Estimated coefficients for model 1

Parameter	Coefficient
(Intercept)	-1773.32
sex (male)	163.04
age	120.89

```r
pred_male <-
  coef(bw_lm1)["(Intercept)"] +
  coef(bw_lm1)["sexmale"] +
  coef(bw_lm1)["age"] * 36
```

$$\begin{aligned} E[Y|M=1,A=36] &= \beta_0 + \beta_M \cdot 1 + \beta_A \cdot 36 \\ &= 2741.913237 \end{aligned}$$

Exercise 1.3. What's the difference in mean birthweights between males born at 36 weeks and females born at 36 weeks?

```r
coef(bw_lm1)
#> (Intercept)     sexmale         age
#>   -1773.322     163.039     120.894
```

Solution.

$$\begin{aligned}&E[Y|M=1,A=36]-E[Y|M=0,A=36]\\&=2741.913237-2578.873934\\&=163.039303\end{aligned}$$

Shortcut:

$$\begin{aligned}&E[Y|M=1,A=36]-E[Y|M=0,A=36]\\&=(\beta_0+\beta_M\cdot 1+\beta_A\cdot 36)-(\beta_0+\beta_M\cdot 0+\beta_A\cdot 36)\\&=\beta_M\\&=163.039303\end{aligned}$$

Age cancels out in this difference. In other words, according to this model, the difference between females and males with the same gestational age is the same for every age.

This characteristic is an assumption of the model specified by Equation 1. It's hardwired into the parametric model structure, even before we estimated values for those parameters.

Coefficient Interpretation

Recall Model 1:

$$E[Y|M=m,A=a]=\mu(m,a)=\beta_0+\beta_M m+\beta_A a$$

Slope (of the mean with respect to age) for males:

$$\begin{aligned}\frac{d}{da}\mu(1,a)&=\frac{d}{da}(\beta_0+\beta_M 1+\beta_A a)\\&=\left(\frac{d}{da}\beta_0+\frac{d}{da}\beta_M 1+\frac{d}{da}\beta_A a\right)\\&=(0+0+\beta_A)\\&=\beta_A\end{aligned}$$

Slope for females:

$$\begin{aligned}\frac{d}{da}\mu(0,a)&=\frac{d}{da}(\beta_0+\beta_M 0+\beta_A a)\\&=\left(\frac{d}{da}\beta_0+\frac{d}{da}\beta_M 0+\frac{d}{da}\beta_A a\right)\\&=(0+0+\beta_A)\\&=\beta_A\end{aligned}$$

Exercise 1.4. What is the interpretation of β_A in Model 1?

Solution.

$$\begin{aligned}\frac{d}{da}\mu(m,a)&=\frac{d}{da}(\beta_0+\beta_M m+\beta_A a)\\&=\left(\frac{d}{da}\beta_0+\frac{d}{da}\beta_M m+\frac{d}{da}\beta_A a\right)\\&=(0+0+\beta_A)\\&=\beta_A\end{aligned}$$

Conclusion:

$$\beta_A = \frac{d}{da}\mu(m, a)$$

β_A is the slope of mean birthweight with respect to gestational age, adjusting for sex.

Or we can plug in the definition of slope:

$$\beta_A = E[Y|M = m, A = a + 1] - E[Y|M = m, A = a]$$

Exchangeability and consistency have not been assessed; so we are not discussing potential outcomes (causality), only observed outcomes.

Exercise 1.5. What is the interpretation of β_M in Model 1?

Solution.

More precisely written:

$$\text{E}[Y|M = m, A = a] = \mu(m, a) = \begin{cases} \beta_0 + \beta_M m + \beta_A a, & \text{for } m \in \{0, 1\} \\ \text{undefined}, & \text{for } m \notin \{0, 1\} \end{cases}$$

The model is undefined for $m \notin \{0, 1\}$, so the derivative with respect to m doesn't exist.

$$\begin{aligned} E[Y|M = 1, A = a] &= \beta_0 + \beta_M 1 + \beta_A a \\ &= \beta_0 + \beta_M + \beta_A a \\ E[Y|M = 0, A = a] &= \beta_0 + \beta_M 0 + \beta_A a \\ &= \beta_0 + \beta_A a \end{aligned}$$

So:

$$\begin{aligned} E[Y|M = 1, A = a] - E[Y|M = 0, A = a] &= (\beta_0 + \beta_M + \beta_A a) - (\beta_0 + \beta_A a) \\ &= \beta_M \end{aligned}$$

Therefore:

$$\begin{aligned} \beta_M &= E[Y|M = 1, A = a] - E[Y|M = 0, A = a] \\ &= \mu(1, a) - \mu(0, a) \end{aligned}$$

In words: β_M is the difference in mean birthweight between males and females adjusting for age.

Exercise 1.6. $\beta_0 = ?$

Solution.

$$\begin{aligned} \text{E}[Y|M = 0, A = 0] &= \mu(0, 0) \\ &= \beta_0 + \beta_M 0 + \beta_A 0 \\ &= \beta_0 \\ \beta_0 &= \text{E}[Y|M = 0, A = 0] = \mu(0, 0) \end{aligned}$$

β_0 is the mean birthweight for a female with gestational age 0 weeks.

1.2.4 Interactions

What if we don't like that parallel lines assumption?

Then we need to allow an "interaction" between age A and sex S:

$$E[Y|S=s,A=a] = \beta_0 + \beta_A a + \beta_M m + \beta_{AM}(a \cdot m) \tag{2}$$

Now, the slope of mean birthweight $E[Y|A,S]$ with respect to gestational age A depends on the value of sex S.

```r
bw_lm2 <- lm(weight ~ sex + age + sex:age, data = bw)
bw_lm2 |>
  parameters() |>
  parameters::print_md(
    include_reference = include_reference_lines,
    select = "{estimate}"
  )
```

Table 8: Birthweight model with interaction term

Parameter	Coefficient
(Intercept)	-2141.67
sex (male)	872.99
age	130.40
sex (male) × age	-18.42

```r
bw <-
  bw |>
  mutate(
    predlm2 = predict(bw_lm2)
  ) |>
  arrange(sex, age)

plot1_interact <-
  plot1 %+% bw +
  geom_line(aes(y = predlm2))

print(plot1_interact)
```

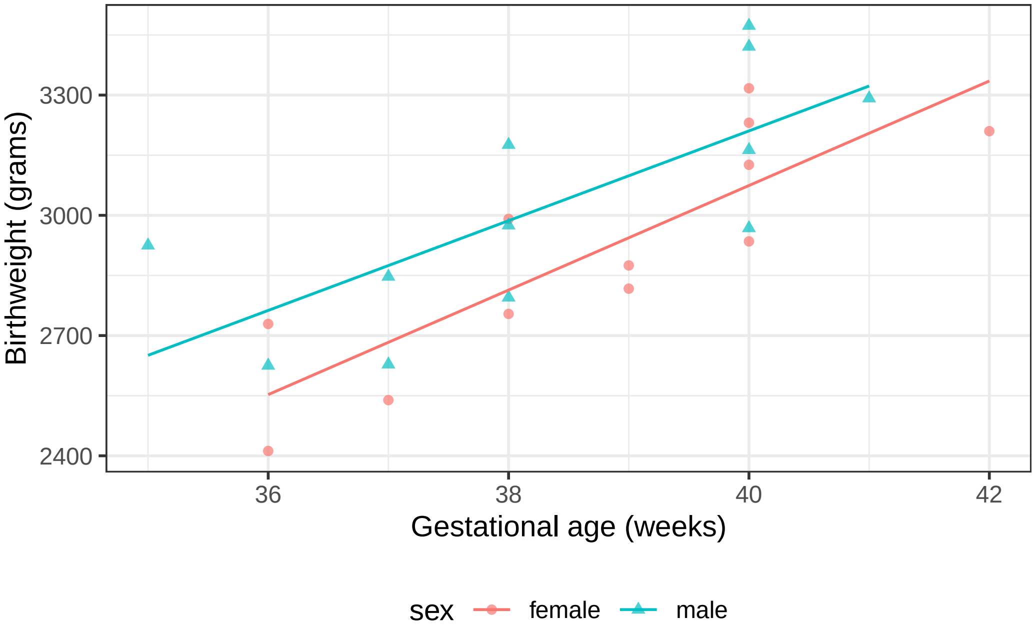


Figure 3: Birthweight model with interaction term

Now we can see that the lines aren't parallel.

Here's another way we could rewrite this model (by collecting terms involving S):

$$E[Y|M, A] = \beta_0 + \beta_M M + (\beta_A + \beta_{AM} M)A$$

If you want to understand a coefficient in a model with interactions, collect terms for the corresponding variable, and you will see which other covariates interact with the variable whose coefficient you are interested in. In this case, the association between A (age) varies between males and females (that is, by sex S). [5] So the slope of Y with respect to A depends on the value of M. According to this model, there is no such thing as "*the* slope of birthweight with respect to age". There are two slopes, one for each sex. We can only talk about "the slope of birthweight with respect to age among males" and "the slope of birthweight with respect to age among females". Then: each non-interaction slope coefficient is the difference in means per unit difference in its corresponding variable, when all interacting variables are set to 0.

To learn what this model is assuming, let's plug in a few values.

Exercise 1.7. According to this model, what's the mean birthweight for a female born at 36 weeks?

Parameter	Coefficient
(Intercept)	-2141.67
sex (male)	872.99
age	130.40
sex (male) × age	-18.42

---

[5] some call this kind of variation "interaction" or "effect modification", but "act", "effect", "modify", and "by" all suggest causality, which we are not prepared to assess here; let's try to avoid using causal terms, unless we are constructing a causal model.

Solution.

Parameter	Coefficient
(Intercept)	-2141.67
sex (male)	872.99
age	130.40
sex (male) × age	-18.42

```r
pred_female <- coef(bw_lm2)["(Intercept)"] + coef(bw_lm2)["age"] * 36
```

$$E[Y|M=0, X_2=36] = \beta_0 + \beta_M \cdot 0 + \beta_A \cdot 36 + \beta_{AM} \cdot (0 * 36) = 2552.733333$$

Exercise 1.8. What's the mean birthweight for a male born at 36 weeks?

Parameter	Coefficient
(Intercept)	-2141.67
sex (male)	872.99
age	130.40
sex (male) × age	-18.42

Solution.

Parameter	Coefficient
(Intercept)	-2141.67
sex (male)	872.99
age	130.40
sex (male) × age	-18.42

```r
pred_male <-
  coef(bw_lm2)["(Intercept)"] +
  coef(bw_lm2)["sexmale"] +
  coef(bw_lm2)["age"] * 36 +
  coef(bw_lm2)["sexmale:age"] * 36
```

$$\begin{aligned} E[Y|M=1, X_2=36] &= \beta_0 + \beta_M \cdot 1 + \beta_A \cdot 36 + \beta_{AM} \cdot 1 \cdot 36 \\ &= 2762.706897 \end{aligned}$$

Exercise 1.9. What's the difference in mean birthweights between males born at 36 weeks and females born at 36 weeks?

Solution.

$$\begin{aligned} & E[Y|M=1, A=36] - E[Y|M=0, A=36] \\ &= (\beta_0 + \beta_M \cdot 1 + \beta_A \cdot 36 + \beta_{AM} \cdot 1 \cdot 36) \\ &\quad - (\beta_0 + \beta_M \cdot 0 + \beta_A \cdot 36 + \beta_{AM} \cdot 0 \cdot 36) \\ &= \beta_S + \beta_{AM} \cdot 36 \\ &= 209.973563 \end{aligned}$$

Note that age now does show up in the difference: in other words, according to this model, the difference in mean birthweights between females and males with the same gestational age can vary by gestational age.

That's how the lines in the graph ended up non-parallel.

Coefficient Interpretation

Exercise 1.10. What is the interpretation of β_M in Model 2?

Solution.

Mean birthweight among males with gestational age 0 weeks:

$$\begin{aligned}\mu(1,0) &= \text{E}[Y|M=1,A=0]\\ &= \beta_0 + \beta_M \cdot 1 + \beta_A \cdot 0 + \beta_{AM} \cdot 1 \cdot 0\\ &= \beta_0 + \beta_M\end{aligned}$$

Mean birthweight among females with gestational age 0 weeks:

$$\begin{aligned}\mu(0,0) &= \text{E}[Y|M=0,A=0]\\ &= \beta_0 + \beta_M \cdot 0 + \beta_A \cdot 0 + \beta_{AM} \cdot 0 \cdot 0\\ &= \beta_0\end{aligned}$$

$$\begin{aligned}\beta_M &= \mu(1,0) - \mu(0,0)\\ &= \text{E}[Y|M=1,A=0] - \text{E}[Y|M=0,A=0]\end{aligned}$$

β_M is the difference in mean birthweight between males with gestational age 0 weeks and females with gestational age 0 weeks.

Exercise 1.11. What is the interpretation of β_{AM} in Model 2?

Solution.

Slope among males:

$$\begin{aligned}\frac{\partial}{\partial a}\mu(1,a) &= \frac{\partial}{\partial a}\left(\beta_0 + \beta_M \cdot 1 + \beta_A \cdot a + \beta_{AM} \cdot 1 \cdot a\right)\\ &= \frac{\partial}{\partial a}\left(\beta_0 + \beta_M + \beta_A \cdot a + \beta_{AM} \cdot a\right)\\ &= \beta_A + \beta_{AM}\end{aligned}$$

or

$$\begin{aligned}E[Y|1,a+1] - E[Y|1,a] =& \beta_0 + \beta_M 1 + \beta_A(a+1) + \beta_{AM}1(a+1)\\ &- (\beta_0 + \beta_M 1 + \beta_A(a) + \beta_{AM}1(a))\\ =& \beta_A + \beta_{AM}\end{aligned}$$

Slope among females:

$$\begin{aligned}\frac{\partial}{\partial a}\mu(0,a) &= \frac{\partial}{\partial a}\left(\beta_0 + \beta_M \cdot 0 + \beta_A \cdot a + \beta_{AM} \cdot 0 \cdot a\right)\\ &= \frac{\partial}{\partial a}\left(\beta_0 + \beta_A \cdot a\right)\\ &= \beta_A\end{aligned}$$

or

$$\begin{aligned}E[Y|0,a+1]-E[Y|0,a] =&\beta_0+\beta_M 0+\beta_A(a+1)+\beta_{AM}0(a+1)\\&-(\beta_0+\beta_M 0+\beta_A(a)+\beta_{AM}0(a))\\=&\beta_0+\beta_A(a+1)-(\beta_0+\beta_A(a))\\=&\beta_A\end{aligned}$$

Difference in slopes:

$$\begin{aligned}\frac{\partial}{\partial a}\mu(1,a)-\frac{\partial}{\partial a}\mu(0,a) &= \beta_A+\beta_{AM}-\beta_A\\&=\beta_{AM}\end{aligned}$$

or

$$\begin{aligned}(E[Y|1,a+1]-E[Y|1,a])-(E[Y|0,a+1]-E[Y|0,a]) &= \beta_A+\beta_{AM}-\beta_A\\&=\beta_{AM}\end{aligned}$$

Therefore

$$\begin{aligned}\beta_{AM} =&\frac{\partial}{\partial a}\mu(1,a)-\frac{\partial}{\partial a}\mu(0,a)\\=&(E[Y|M=1,A=a+1]-E[Y|M=1,A=a])\\&-(E[Y|M=0,A=a+1]-E[Y|M=0,A=a])\end{aligned}$$

β_{AM} is the difference in slope of mean birthweight with respect to gestational age between males and females.

Compare coefficient interpretations

Table 13: Coefficient interpretations, by model structure

$\mu(m,a)$	$\beta_0+\beta_M m+\beta_A a$	$\beta_0+\beta_M m+\beta_A a+\beta_{AM} ma$
β_0	$\mu(0,0)$	$\mu(0,0)$
β_A	$\frac{\partial}{\partial a}\mu(m,a)$	$\frac{\partial}{\partial a}\mu(0,a)$
β_M	$\mu(1,a)-\mu(0,a)$	$\mu(1,0)-\mu(0,0)$
β_{AM}		$\frac{\partial}{\partial a}\mu(1,a)-\frac{\partial}{\partial a}\mu(0,a)$

In the model with an interaction term multiplying $A \times M$, the interpretation of β_A involves the reference level of M, and interpretation of β_M involves the reference level of A (Table 13).

1.2.5 Stratified regression

We could re-write the interaction model as a stratified model, with a slope and intercept for each sex:

$$\text{E}[Y|A=a,S=s]=\beta_M m+\beta_{AM}(a\cdot m)+\beta_F f+\beta_{AF}(a\cdot f) \tag{3}$$

Compare this stratified model (Equation 3) with our interaction model, Equation 2:

$$\text{E}[Y|A=a,S=s]=\beta_0+\beta_A a+\beta_M m+\beta_{AM}(a\cdot m)$$

In the stratified model, the intercept term β_0 has been relabeled as β_F.

```r
bw_lm2 <- lm(weight ~ sex + age + sex:age, data = bw)
bw_lm2 |>
  parameters() |>
  print_md(
    include_reference = include_reference_lines,
    select = "{estimate}"
  )
```

Table 14: Birthweight model with interaction term

Parameter	Coefficient
(Intercept)	-2141.67
sex (male)	872.99
age	130.40
sex (male) × age	-18.42

```
bw_lm_strat <-
  bw |>
  lm(
    formula = weight ~ sex + sex:age - 1,
    data = _
  )

bw_lm_strat |>
  parameters() |>
  print_md(
    select = "{estimate}"
  )
```

Table 15: Birthweight model - stratified betas

Parameter	Coefficient
sex (female)	-2141.67
sex (male)	-1268.67
sex (female) × age	130.40
sex (male) × age	111.98

1.2.6 Curved-line regression

If we transform some of our covariates (Xs) and plot the resulting model on the original covariate scale, we end up with curved regression lines:

```
bw_lm3 <- lm(weight ~ sex:log(age) - 1, data = bw)

ggbw <-
  bw |>
  ggplot(
    aes(x = age, y = weight)
  ) +
  geom_point() +
  xlab("Gestational Age (weeks)") +
  ylab("Birth Weight (g)")

ggbw2 <- ggbw +
  stat_smooth(
    method = "lm",
    formula = y ~ log(x),
    geom = "smooth"
  ) +
  xlab("Gestational Age (weeks)") +
  ylab("Birth Weight (g)")
```

```
ggbw2 |> print()
```

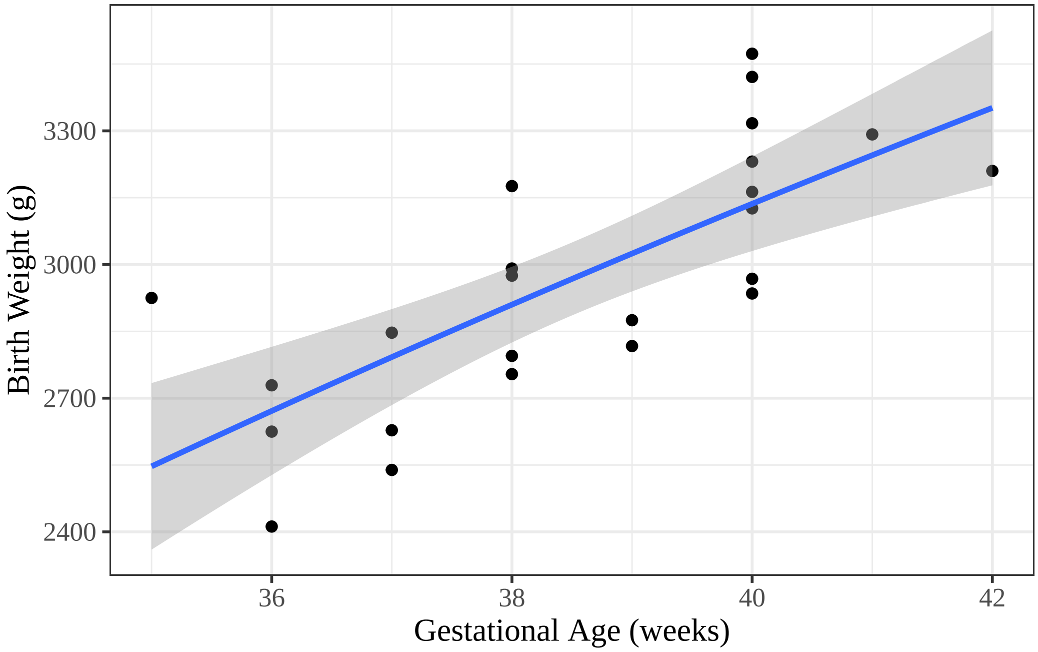


Figure 4: `birthweight` model with `age` entering on log scale

Below is an example with a slightly more obvious curve.

```
library(palmerpenguins)

ggpenguins <-
  palmerpenguins::penguins |>
  dplyr::filter(species == "Adelie") |>
  ggplot(
    aes(x = bill_length_mm, y = body_mass_g)
  ) +
  geom_point() +
  xlab("Bill length (mm)") +
  ylab("Body mass (g)")

ggpenguins2 <- ggpenguins +
  stat_smooth(
    method = "lm",
    formula = y ~ log(x),
    geom = "smooth"
  ) +
  xlab("Bill length (mm)") +
  ylab("Body mass (g)")


ggpenguins2 |> print()
```

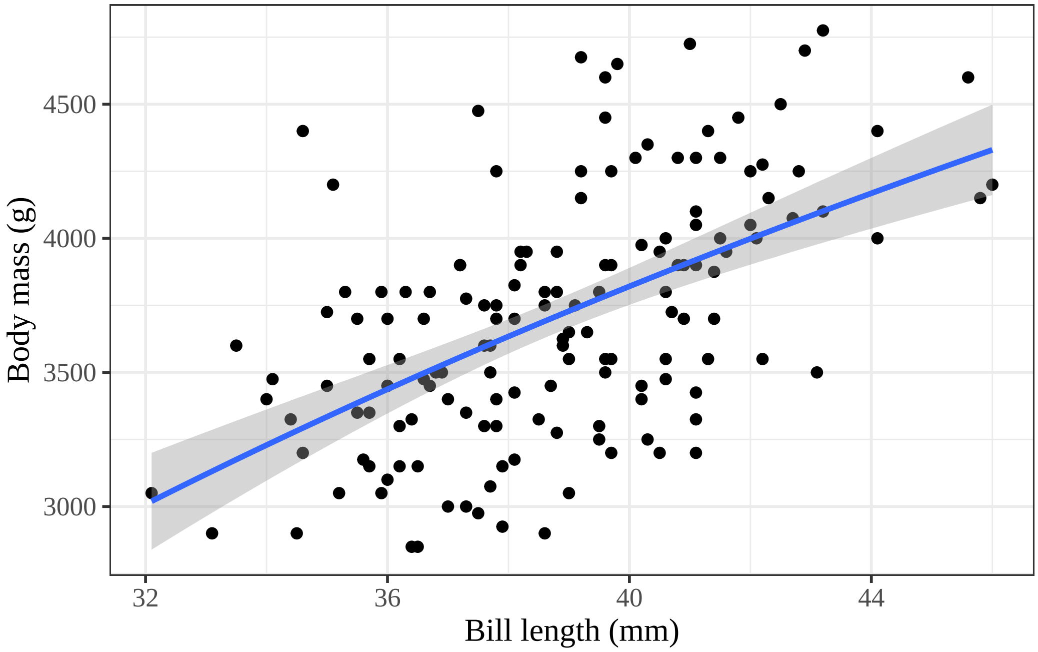


Figure 5: `palmerpenguins` model with `bill_length` entering on log scale

1.3 Estimating Linear Models via Maximum Likelihood

In EPI 203 and our review of MLEs[6], we learned how to fit outcome-only models of the form $p(X = x|\theta)$ to iid data $\tilde{x} = (x_1, ..., x_n)$ using maximum likelihood estimation.

Now, we apply the same procedure to linear regression models:

1.3.1 Likelihood

$$
\begin{aligned}
\mathcal{L}_i &\stackrel{\text{def}}{=} p(Y_i = y_i|\tilde{X}_i = \tilde{x}_i) \\
&= (2\pi\sigma^2)^{-1/2} \text{exp}\left\{-\frac{1}{2\sigma^2}\varepsilon_i^2\right\} \\
\varepsilon_i &\stackrel{\text{def}}{=} y_i - \mu_i \\
\mu_i &\stackrel{\text{def}}{=} \mu(x_i) \\
&= x_i \cdot \beta
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L} &\stackrel{\text{def}}{=} \mathcal{L}(\tilde{y}|\mathbf{x}, \tilde{\beta}, \sigma^2) \\
&\stackrel{\text{def}}{=} \text{p}(\tilde{Y} = \tilde{y}|\mathbf{X} = \mathbf{x}) \\
&= \prod_{i=1}^{n} \mathcal{L}_i
\end{aligned}
\tag{4}
$$

---

[6] intro-MLEs.qmd#sec-intro-MLEs

1.3.2 Log-likelihood

$$\begin{aligned}
\ell_i &\overset{\text{def}}{=} \log\{\mathcal{L}_i\} \\
&= \log\left\{(2\pi\sigma^2)^{-1/2}\exp\left\{-\frac{1}{2\sigma^2}\varepsilon_i^2\right\}\right\} \\
&= -\frac{1}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}\varepsilon_i^2
\end{aligned}$$

$$\begin{aligned}
\ell &\overset{\text{def}}{=} \ell(\tilde{y}|\mathbf{x},\beta,\sigma^2) \\
&\overset{\text{def}}{=} \log\{\mathcal{L}(\tilde{y}|\mathbf{x},\beta,\sigma^2)\} \\
&= \log\left\{\prod_{i=1}^{n}\mathcal{L}_i\right\} \\
&= \sum_{i=1}^{n}\log\{\mathcal{L}_i\} \\
&= \sum_{i=1}^{n}\ell_i \\
&= \sum_{i=1}^{n}\left(-\frac{1}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}\varepsilon_i^2\right) \qquad (5) \\
&= -\frac{n}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\varepsilon_i^2 \\
&= -\frac{n}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}(\tilde{\varepsilon}\cdot\tilde{\varepsilon}) \\
&= -\frac{n}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}((\tilde{y}-\tilde{\mu})\cdot(\tilde{y}-\tilde{\mu})) \\
&= -\frac{n}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}\left((\tilde{y}-\mathbf{X}\tilde{\beta})\cdot(\tilde{y}-\mathbf{X}\tilde{\beta})\right) \\
&= -\frac{n}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_i - \left(\tilde{x}_i\cdot\tilde{\beta}\right)\right)^2
\end{aligned}$$

1.3.3 Score function

$$\begin{aligned}
\mu_i' &\overset{\text{def}}{=} \frac{\partial}{\partial\tilde{\beta}}\mu_i \\
&= \frac{\partial}{\partial\tilde{\beta}}\left(\tilde{x}_i\cdot\tilde{\beta}\right) \\
&= \left(\frac{\partial}{\partial\tilde{\beta}}\tilde{\beta}\right)\tilde{x}_i \\
&= \mathbb{I}\tilde{x}_i \\
&= \tilde{x}_i
\end{aligned}$$

$$\begin{aligned}
\varepsilon_i' &\overset{\text{def}}{=} \frac{\partial}{\partial\tilde{\beta}}\varepsilon_i \\
&= \frac{\partial}{\partial\tilde{\beta}}(y_i-\mu_i) \\
&= \frac{\partial}{\partial\tilde{\beta}}y_i - \frac{\partial}{\partial\tilde{\beta}}\mu_i \\
&= 0 - \tilde{x}_i \\
&= -\tilde{x}_i
\end{aligned}$$

$$\begin{aligned}
\ell_i' &\overset{\text{def}}{=} \frac{\partial}{\partial\tilde{\beta}}\ell_i \\
&= \frac{\partial}{\partial\tilde{\beta}}\Big(-\frac{1}{2}\log\{2\pi\sigma^2\} - \frac{1}{2\sigma^2}\varepsilon_i^2\Big) \\
&= \frac{\partial}{\partial\tilde{\beta}}\Big(-\frac{1}{2}\log\{2\pi\sigma^2\}\Big) - \frac{\partial}{\partial\tilde{\beta}}\frac{1}{2\sigma^2}\varepsilon_i^2 \\
&= 0 - \frac{1}{2\sigma^2}\frac{\partial}{\partial\tilde{\beta}}\varepsilon_i^2 \\
&= -\frac{1}{2\sigma^2}2(\varepsilon_i')\varepsilon_i \\
&= -\frac{1}{\sigma^2}(-\tilde{x}_i\varepsilon_i) \\
&= \frac{1}{\sigma^2}\tilde{x}_i\varepsilon_i
\end{aligned}$$

$$\begin{aligned}
\ell_{\tilde{\beta}}' &\overset{\text{def}}{=} \frac{\partial}{\partial\tilde{\beta}}\ell_{\tilde{\beta}} \\
&= \frac{\partial}{\partial\tilde{\beta}}\sum_{i=1}^{n}\ell_i \\
&= \sum_{i=1}^{n}\frac{\partial}{\partial\tilde{\beta}}\ell_i \\
&= \sum_{i=1}^{n}\ell_i' \\
&= \sum_{i=1}^{n}\frac{1}{\sigma^2}\tilde{x}_i\varepsilon_i \\
&= \frac{1}{\sigma^2}\sum_{i=1}^{n}\tilde{x}_i\varepsilon_i \\
&= \frac{1}{\sigma^2}\mathbf{X}^\top\tilde{\varepsilon}
\end{aligned}$$

1.3.4 Hessian

$$\begin{aligned}
\ell_i'' &\overset{\text{def}}{=} \frac{\partial}{\partial\tilde{\beta}^\top}\frac{\partial}{\partial\tilde{\beta}}\ell_i \\
&= \frac{\partial}{\partial\tilde{\beta}^\top}\ell_i' \\
&= \frac{\partial}{\partial\tilde{\beta}^\top}\Big(\frac{1}{\sigma^2}\tilde{x}_i\varepsilon_i\Big) \\
&= \frac{1}{\sigma^2}\tilde{x}_i\varepsilon_i'^\top \\
&= \frac{1}{\sigma^2}\tilde{x}_i(-\tilde{x}_i^\top) \\
&= -\frac{1}{\sigma^2}\tilde{x}_i\tilde{x}_i^\top
\end{aligned}$$

$$
\begin{aligned}
\ell'' &\stackrel{\text{def}}{=} \frac{\partial}{\partial \tilde{\beta}^{\top}} \frac{\partial}{\partial \tilde{\beta}} \ell \\
&= \frac{\partial}{\partial \tilde{\beta}^{\top}} \ell' \\
&= \frac{\partial}{\partial \tilde{\beta}^{\top}} \sum_{i=1}^{n} \ell_i' \\
&= \sum_{i=1}^{n} \frac{\partial}{\partial \tilde{\beta}^{\top}} \ell_i' \\
&= \sum_{i=1}^{n} \ell_i'' \\
&= \sum_{i=1}^{n} -\frac{1}{\sigma^2} \tilde{x}_i \tilde{x}_i^{\top} \\
&= -\frac{1}{\sigma^2} \sum_{i=1}^{n} \tilde{x}_i \tilde{x}_i^{\top} \\
&= -\frac{1}{\sigma^2} \mathbf{X}^{\top} \mathbf{X}
\end{aligned}
$$

That is,

$$
\ell'' = -\frac{1}{\sigma^2} \sum_{i=1}^{n} \tilde{x}_i \tilde{x}_i^{\top} \tag{6}
$$

1.3.5 Alternative approach using matrix derivatives

$$
\begin{aligned}
\ell_{\tilde{\beta}}'(\tilde{y}|\mathbf{x}, \tilde{\beta}, \sigma^2) &\stackrel{\text{def}}{=} \frac{\partial}{\partial \tilde{\beta}} \ell_{\tilde{\beta}}(\tilde{y}|\mathbf{x}, \tilde{\beta}, \sigma^2) \\
&= -\frac{1}{2\sigma^2} \frac{\partial}{\partial \tilde{\beta}} \left(\sum_{i=1}^{n} \left(y_i - \left(\tilde{x}_i \cdot \tilde{\beta} \right) \right)^2 \right)
\end{aligned} \tag{7}
$$

Let's switch to matrix-vector notation:

$$
\sum_{i=1}^{n} (y_i - \tilde{x}_i^{\top} \tilde{\beta})^2 = (\tilde{y} - \mathbf{X}\tilde{\beta}) \cdot (\tilde{y} - \mathbf{X}\tilde{\beta})
$$

So

$$
\begin{aligned}
(\tilde{y} - \mathbf{X}\tilde{\beta})'(\tilde{y} - \mathbf{X}\tilde{\beta}) &= (\tilde{y}' - \tilde{\beta}' X')(\tilde{y} - \mathbf{X}\tilde{\beta}) \\
&= \tilde{y}'\tilde{y} - \tilde{\beta}' X' \tilde{y} - \tilde{y}' \mathbf{X} \tilde{\beta} + \tilde{\beta}' \mathbf{X}' \mathbf{X} \beta \\
&= \tilde{y}'\tilde{y} - 2\tilde{y}' \mathbf{X} \beta + \beta' \mathbf{X}' \mathbf{X} \beta
\end{aligned}
$$

We will use some results from vector calculus[7]:

[7] math-prereqs.qmd#sec-vector-calculus

$$\begin{aligned}\frac{\partial}{\partial\tilde{\beta}}\left(\sum_{i=1}^{n}(y_i - x_i'\beta)^2\right) &= \frac{\partial}{\partial\tilde{\beta}}(\tilde{y} - X\beta)'(\tilde{y} - X\beta)\\ &= \frac{\partial}{\partial\tilde{\beta}}(y'y - 2y'X\beta + \beta'\mathbf{X}'\mathbf{X}\beta)\\ &= (-2X'y + 2\mathbf{X}'\mathbf{X}\beta)\\ &= -2X'(y - X\beta)\\ &= -2X'(y - \mathrm{E}[y])\\ &= -2X'\varepsilon(y)\end{aligned}\tag{8}$$

So if $\ell'(\beta, \sigma^2) = 0$, then

$$\begin{aligned}0 &= (-2X'y + 2\mathbf{X}'\mathbf{X}\beta)\\ 2X'y &= 2\mathbf{X}'\mathbf{X}\beta\\ X'y &= \mathbf{X}'\mathbf{X}\beta\\ (\mathbf{X}'\mathbf{X})^{-1}X'y &= \beta\end{aligned}$$

Hessian

The Hessian (second derivative matrix) is:

$$\ell''_{\beta,\beta'}(\beta, \sigma^2; \tilde{y}, \mathbf{X}) = -\frac{1}{2\sigma^2}\mathbf{X}'\mathbf{X}$$

$\ell''_{\beta,\beta'}(\beta, \sigma^2; \mathbf{X}, \tilde{y})$ is negative definite at $\beta = (\mathbf{X}'\mathbf{X})^{-1}X'y$, so $\hat{\beta}_{ML} = (\mathbf{X}'\mathbf{X})^{-1}X'y$ is the MLE for β.

Similarly (not shown):

$$\hat{\sigma}^2_{ML} = \frac{1}{n}(Y - X\hat{\beta})'(Y - X\hat{\beta})$$

And

$$\begin{aligned}\mathcal{I}_\beta &= E[-\ell''_{\beta,\beta'}(Y|X, \beta, \sigma^2)]\\ &= \frac{1}{\sigma^2}\mathbf{X}'\mathbf{X}\end{aligned}$$

So:

$$Var(\hat{\beta}) \approx (\mathcal{I}_\beta)^{-1} = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$$

and

$$\hat{\beta}\dot{\sim}N(\beta, \mathcal{I}_\beta^{-1})$$

These are all results you have hopefully seen before.

Table 16: Covariance matrix of $\hat{\hat{\beta}}$ for `birthweight` model 2 (with interaction term)

```
bw_lm2 |> vcov()
#>             (Intercept)  sexmale        age sexmale:age
#> (Intercept)     1353968 -1353968 -34870.966   34870.966
#> sexmale        -1353968  2596387  34870.966  -67210.974
#> age              -34871    34871    899.896    -899.896
#> sexmale:age       34871   -67211   -899.896    1743.548
```

In the Gaussian linear regression case, we also have exact results:

$$\frac{\hat{\beta}_j}{\widehat{\text{se}}\left(\hat{\beta}_j\right)} \sim t_{n-p}$$

Example 1.1 (MLEs for birthweight data)**.** In model 2 above, $\hat{\mathscr{I}}(\beta)$ is:

If we take the square roots of the diagonals, we get the standard errors listed in the model output:

```
bw_lm2 |>
  vcov() |>
  diag() |>
  sqrt()
#> (Intercept)     sexmale         age sexmale:age
#>   1163.6015   1611.3309     29.9983     41.7558
```

```
bw_lm2 |>
  parameters() |>
  print_md()
```

Table 17: Estimated model for `birthweight` data with interaction term

Parameter	Coefficient	SE	95% CI	t(20)	p
(Intercept)	-2141.67	1163.60	(-4568.90, 285.56)	-1.84	0.081
sex (male)	872.99	1611.33	(-2488.18, 4234.17)	0.54	0.594
age	130.40	30.00	(67.82, 192.98)	4.35	< .001
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

So we can do confidence intervals, hypothesis tests, and p-values exactly as in the one-variable case we looked at previously.

1.3.6 Residual Standard Deviation

$\hat{\sigma}$ represents an *estimate* of the *Residual Standard Deviation* parameter, σ. We can extract $\hat{\sigma}$ from the fitted model, using the `sigma()` function:

```
sigma(bw_lm2)
#> [1] 180.613
```

σ is NOT "Residual standard error"

In the `summary.lm()` output, this estimate is labeled as `"Residual standard error"`:

```
summary(bw_lm2)
#>
#> Call:
```

```
#> lm(formula = weight ~ sex + age + sex:age, data = bw)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -246.7 -138.1  -39.1  176.6  274.3
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  -2141.7     1163.6   -1.84  0.08057 .
#> sexmale        873.0     1611.3    0.54  0.59395
#> age            130.4       30.0    4.35  0.00031 ***
#> sexmale:age    -18.4       41.8   -0.44  0.66389
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 181 on 20 degrees of freedom
#> Multiple R-squared:  0.643,  Adjusted R-squared:  0.59
#> F-statistic:   12 on 3 and 20 DF,  p-value: 0.000101
```

However, this is a misnomer: see note in `?stats::sigma`

1.4 Inference about Gaussian Linear Regression Models

1.4.1 Motivating example: `birthweight` data

Research question: is there really an interaction between sex and age?

$H_0 : \beta_{AM} = 0$

$H_A : \beta_{AM} \neq 0$

$P(|\hat{\beta}_{AM}| > |-18.417241| \mid H_0) = ?$

1.4.2 Wald tests and CIs

R can give you Wald tests for single coefficients and corresponding CIs:

```
bw_lm2 |>
  parameters() |>
  print_md(
    include_reference = TRUE
  )
```

Parameter	Coefficient	SE	95% CI	t(20)	p
(Intercept)	-2141.67	1163.60	(-4568.90, 285.56)	-1.84	0.081
sex (female)	0.00				
sex (male)	872.99	1611.33	(-2488.18, 4234.17)	0.54	0.594
age	130.40	30.00	(67.82, 192.98)	4.35	< .001
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

To understand what's happening, let's replicate these results by hand for the interaction term.

1.4.3 P-values

```
bw_lm2 |>
  parameters(keep = "sexmale:age") |>
  print_md(
    include_reference = TRUE
  )
```

Parameter	Coefficient	SE	95% CI	t(20)	p
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

```r
beta_hat <- coef(summary(bw_lm2))["sexmale:age", "Estimate"]
se_hat <- coef(summary(bw_lm2))["sexmale:age", "Std. Error"]
dfresid <- bw_lm2$df.residual
t_stat <- abs(beta_hat) / se_hat
pval_t <-
  pt(-t_stat, df = dfresid, lower.tail = TRUE) +
  pt(t_stat, df = dfresid, lower.tail = FALSE)
```

$$
\begin{aligned}
&P\left(|\hat{\beta}_{AM}| > |-18.417241| \middle| H_0\right) \\
&= \Pr\left(\left|\frac{\hat{\beta}_{AM}}{\widehat{SE}(\hat{\beta}_{AM})}\right| > \left|\frac{-18.417241}{41.755817}\right| \middle| H_0\right) \\
&= \Pr\left(|T_{20}| > 0.44107 | H_0\right) \\
&= 0.663893
\end{aligned}
$$

This matches the result in the table above.

1.4.4 Confidence intervals

```r
bw_lm2 |>
  parameters(keep = "sexmale:age") |>
  print_md(
    include_reference = TRUE
  )
```

Parameter	Coefficient	SE	95% CI	t(20)	p
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

```r
q_t <- qt(
  p = 0.975,
  df = dfresid,
  lower.tail = TRUE
)

q_t <- qt(
  p = 0.025,
  df = dfresid,
  lower.tail = TRUE
)


confint_radius_t <-
  se_hat * q_t

confint_t <- beta_hat + c(-1, 1) * confint_radius_t

print(confint_t)
#> [1]   68.6839 -105.5184
```

This also matches.

1.4.5 Gaussian approximations

Here are the asymptotic (Gaussian approximation) equivalents:

1.4.6 P-values

```
bw_lm2 |>
  parameters(keep = "sexmale:age") |>
  print_md(
    include_reference = TRUE
  )
```

Parameter	Coefficient	SE	95% CI	t(20)	p
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

```
pval_z <- pnorm(abs(t_stat), lower = FALSE) * 2

print(pval_z)
#> [1] 0.659162
```

1.4.7 Confidence intervals

```
bw_lm2 |>
  parameters(keep = "sexmale:age") |>
  print_md(
    include_reference = TRUE
  )
```

Parameter	Coefficient	SE	95% CI	t(20)	p
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

```
confint_radius_z <- se_hat * qnorm(0.975, lower = TRUE)
confint_z <-
  beta_hat + c(-1, 1) * confint_radius_z
print(confint_z)
#> [1] -100.2571   63.4227
```

1.4.8 Likelihood ratio statistics

```
logLik(bw_lm2)
#> 'log Lik.' -156.579 (df=5)
logLik(bw_lm1)
#> 'log Lik.' -156.695 (df=4)

log_LR <- (logLik(bw_lm2) - logLik(bw_lm1)) |> as.numeric()
delta_df <- (bw_lm1$df.residual - df.residual(bw_lm2))


x_max <- 1
```

```
d_log_LR <- function(x, df = delta_df) dchisq(x, df = df)

chisq_plot <-
```

```
  ggplot() +
  geom_function(fun = d_log_LR) +
  stat_function(
    fun = d_log_LR,
    xlim = c(log_LR, x_max),
    geom = "area",
    fill = "gray"
  ) +
  geom_segment(
    aes(
      x = log_LR,
      xend = log_LR,
      y = 0,
      yend = d_log_LR(log_LR)
    ),
    col = "red"
  ) +
  xlim(0.0001, x_max) +
  ylim(0, 4) +
  ylab("p(X=x)") +
  xlab("log(likelihood ratio) statistic [x]") +
  theme_classic()
chisq_plot |> print()
```

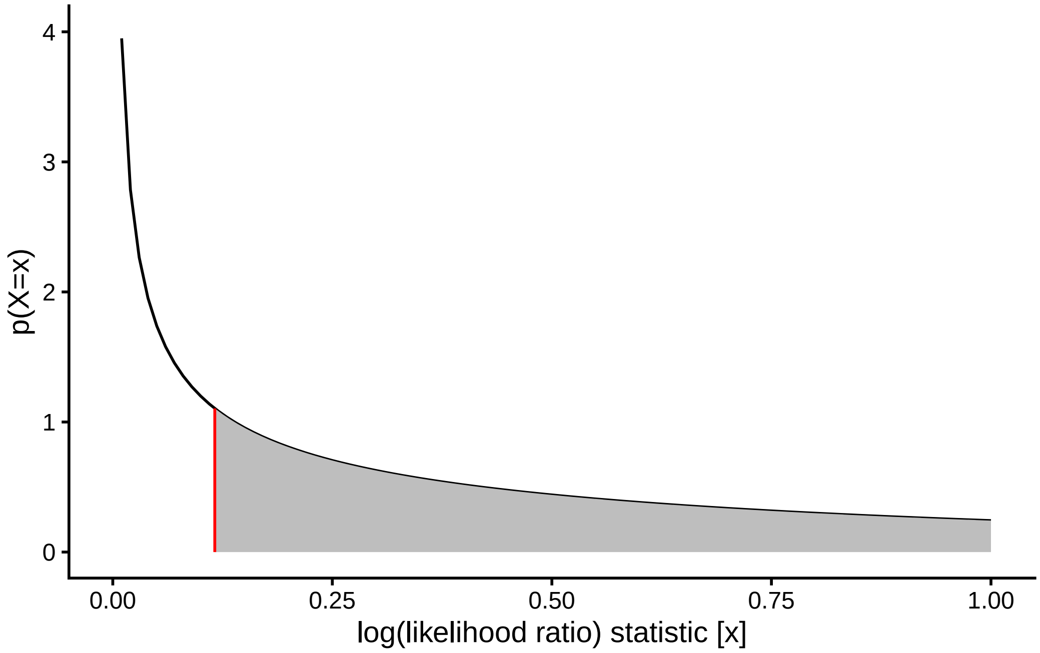


Figure 6: Chi-square distribution

Now we can get the p-value:

```
pchisq(
  q = 2 * log_LR,
  df = delta_df,
  lower = FALSE
) |>
  print()
```

```
#> [1] 0.629806
```

In practice you don't have to do this by hand; there are functions to do it for you:

```
# built in
library(lmtest)
lrtest(bw_lm2, bw_lm1)
#> # A tibble: 2 x 5
#>   `#Df` LogLik    Df  Chisq `Pr(>Chisq)`
#>   <dbl>  <dbl> <dbl>  <dbl>        <dbl>
#> 1     5  -157.    NA NA           NA
#> 2     4  -157.    -1  0.232        0.630
```

1.5 Goodness of fit

1.5.1 AIC and BIC

When we use likelihood ratio tests, we are comparing how well different models fit the data.

Likelihood ratio tests require "nested" models: one must be a special case of the other.

If we have non-nested models, we can instead use the Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC):

- AIC $= -2 * \ell(\hat{\theta}) + 2 * p$
- BIC $= -2 * \ell(\hat{\theta}) + p * \log(n)$

where ℓ is the log-likelihood of the data evaluated using the parameter estimates $\hat{\theta}$, p is the number of estimated parameters in the model (including $\hat{\sigma}^2$), and n is the number of observations.

You can calculate these criteria using the `logLik()` function, or use the built-in R functions:

AIC in R

```
-2 * logLik(bw_lm2) |> as.numeric() +
  2 * (length(coef(bw_lm2)) + 1) # sigma counts as a parameter here
#> [1] 323.159

AIC(bw_lm2)
#> [1] 323.159
```

BIC in R

```
-2 * logLik(bw_lm2) |> as.numeric() +
  (length(coef(bw_lm2)) + 1) * log(nobs(bw_lm2))
#> [1] 329.049

BIC(bw_lm2)
#> [1] 329.049
```

Large values of AIC and BIC are worse than small values. There are no hypothesis tests or p-values associated with these criteria.

1.5.2 (Residual) Deviance

Let q be the number of distinct covariate combinations in a data set.

```
bw_X_unique <-
  bw |>
  count(sex, age)
```

Table 23: Unique covariate combinations in the **birthweight** data, with replicate counts

```
bw_X_unique
#> # A tibble: 12 x 3
#>    sex      age     n
#>    <fct>  <dbl> <int>
#>  1 female    36     2
#>  2 female    37     1
#>  3 female    38     2
#>  4 female    39     2
#>  5 female    40     4
#>  6 female    42     1
#>  7 male      35     1
#>  8 male      36     1
#>  9 male      37     2
#> 10 male      38     3
#> 11 male      40     4
#> 12 male      41     1
```

```
n_unique_bw <- nrow(bw_X_unique)
```

For example, in the `birthweight` data, there are $q = 12$ unique patterns (Table 23).

Definition 1.1 (Replicates)**.** If a given covariate pattern has more than one observation in a dataset, those observations are called **replicates**.

Example 1.2 (Replicates in the `birthweight` data)**.** In the `birthweight` dataset, there are 2 replicates of the combination "female, age 36" (Table 23).

Exercise 1.12 (Replicates in the `birthweight` data)**.** Which covariate pattern(s) in the `birthweight` data has the most replicates?

Solution 1.1 (Replicates in the `birthweight` data). Two covariate patterns are tied for most replicates: males at age 40 weeks and females at age 40 weeks. 40 weeks is the usual length for human pregnancy (Polin, Fox, and Abman (2011)), so this result makes sense.

```
bw_X_unique |> dplyr::filter(n == max(n))
#> # A tibble: 2 x 3
#>   sex      age     n
#>   <fct>  <dbl> <int>
#> 1 female    40     4
#> 2 male      40     4
```

Saturated models

The most complicated model we could fit would have one parameter (a mean) for each covariate pattern, plus a variance parameter:

```
lm_max <-
  bw |>
  mutate(age = factor(age)) |>
  lm(
```

```r
    formula = weight ~ sex:age - 1,
    data = _
  )

lm_max |>
  parameters() |>
  print_md()
```

Table 24: Saturated model for the `birthweight` data

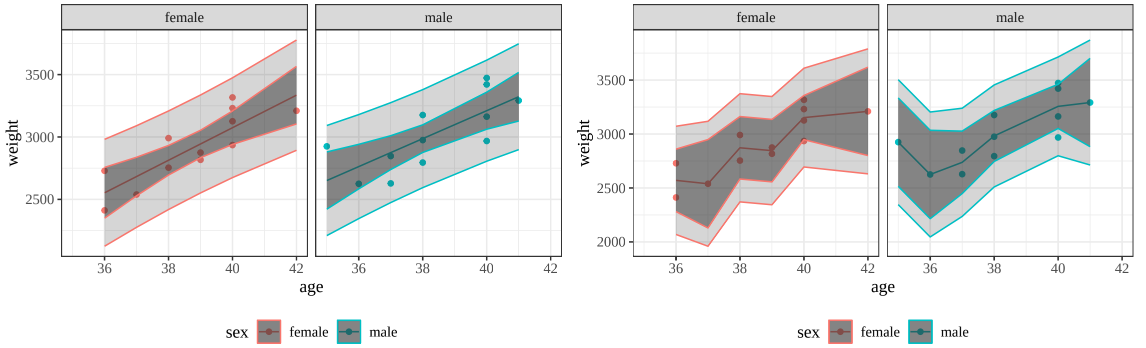
Parameter	Coefficient	SE	95% CI	t(12)	p
sex (male) × age35	2925.00	187.92	(2515.55, 3334.45)	15.56	< .001
sex (female) × age36	2570.50	132.88	(2280.98, 2860.02)	19.34	< .001
sex (male) × age36	2625.00	187.92	(2215.55, 3034.45)	13.97	< .001
sex (female) × age37	2539.00	187.92	(2129.55, 2948.45)	13.51	< .001
sex (male) × age37	2737.50	132.88	(2447.98, 3027.02)	20.60	< .001
sex (female) × age38	2872.50	132.88	(2582.98, 3162.02)	21.62	< .001
sex (male) × age38	2982.00	108.50	(2745.60, 3218.40)	27.48	< .001
sex (female) × age39	2846.00	132.88	(2556.48, 3135.52)	21.42	< .001
sex (female) × age40	3152.25	93.96	(2947.52, 3356.98)	33.55	< .001
sex (male) × age40	3256.25	93.96	(3051.52, 3460.98)	34.66	< .001
sex (male) × age41	3292.00	187.92	(2882.55, 3701.45)	17.52	< .001
sex (female) × age42	3210.00	187.92	(2800.55, 3619.45)	17.08	< .001

We call this model the **full**, **maximal**, or **saturated** model for this dataset.

```r
library(rlang) # defines the `.data` pronoun
plot_PIs_and_CIs <- function(model, data) {
  cis <- model |>
    predict(interval = "confidence") |>
    suppressWarnings() |>
    tibble::as_tibble()
  names(cis) <- paste("ci", names(cis), sep = "_")

  preds <- model |>
    predict(interval = "predict") |>
    suppressWarnings() |>
    tibble::as_tibble()
  names(preds) <- paste("pred", names(preds), sep = "_")
  dplyr::bind_cols(bw, cis, preds) |>
    ggplot2::ggplot() +
    ggplot2::aes(
      x = .data$age,
      y = .data$weight,
      col = .data$sex
    ) +
    ggplot2::geom_point() +
    ggplot2::theme(legend.position = "bottom") +
    ggplot2::geom_line(ggplot2::aes(y = .data$ci_fit)) +
    ggplot2::geom_ribbon(
      ggplot2::aes(
        ymin = .data$pred_lwr,
        ymax = .data$pred_upr
      ),
      alpha = 0.2
    ) +
    ggplot2::geom_ribbon(
      ggplot2::aes(
        ymin = .data$ci_lwr,
        ymax = .data$ci_upr
      ),
      alpha = 0.5
    ) +
    ggplot2::facet_wrap(~sex)
}
```

```
plot_PIs_and_CIs(bw_lm2, bw)
```

```
plot_PIs_and_CIs(lm_max, bw)
```



(a) Model 2 (linear with age:sex interaction)

(b) Saturated model

Figure 7: Model 2 and saturated model for **birthweight** data, with confidence and prediction intervals

---

We can calculate the log-likelihood of this model as usual:

```
logLik(lm_max)
#> 'log Lik.' -151.402 (df=13)
```

We can compare this model to our other models using chi-square tests, as usual:

```
lrtest(lm_max, bw_lm2)
#> # A tibble: 2 x 5
#>   `#Df` LogLik    Df Chisq `Pr(>Chisq)`
#>   <dbl>  <dbl> <dbl> <dbl>        <dbl>
#> 1    13  -151.    NA  NA           NA
#> 2     5  -157.    -8  10.4          0.241
```

The likelihood ratio statistic for this test is

$$\lambda = 2 * (\ell_{\text{full}} - \ell) = 10.355374$$

where:

- ℓ_{full} is the log-likelihood of the full model: -151.401601
- ℓ is the log-likelihood of our comparison model (two slopes, two intercepts): -156.579288

This statistic is called the **deviance** or **residual deviance** for our two-slopes and two-intercepts model; it tells us how much the likelihood of that model deviates from the likelihood of the maximal model.

The corresponding p-value tells us whether there we have enough evidence to detect that our two-slopes, two-intercepts model is a worse fit for the data than the maximal model; in other words, it tells us if there's evidence that we missed any important patterns. (Remember, a nonsignificant p-value could mean that we didn't miss anything and a more complicated model is unnecessary, or it could mean we just don't have enough data to tell the difference between these models.)

1.5.3 Null Deviance

Similarly, the *least* complicated model we could fit would have only one mean parameter, an intercept:

$$\mathrm{E}[Y|X = x] = \beta_0$$

We can fit this model in R like so:

```
lm0 <- lm(weight ~ 1, data = bw)

lm0 |>
  parameters() |>
  print_md()
```

Parameter	Coefficient	SE	95% CI	t(23)	p
(Intercept)	2967.67	57.58	(2848.56, 3086.77)	51.54	< .001

```
lm0 |> plot_PIs_and_CIs()
```

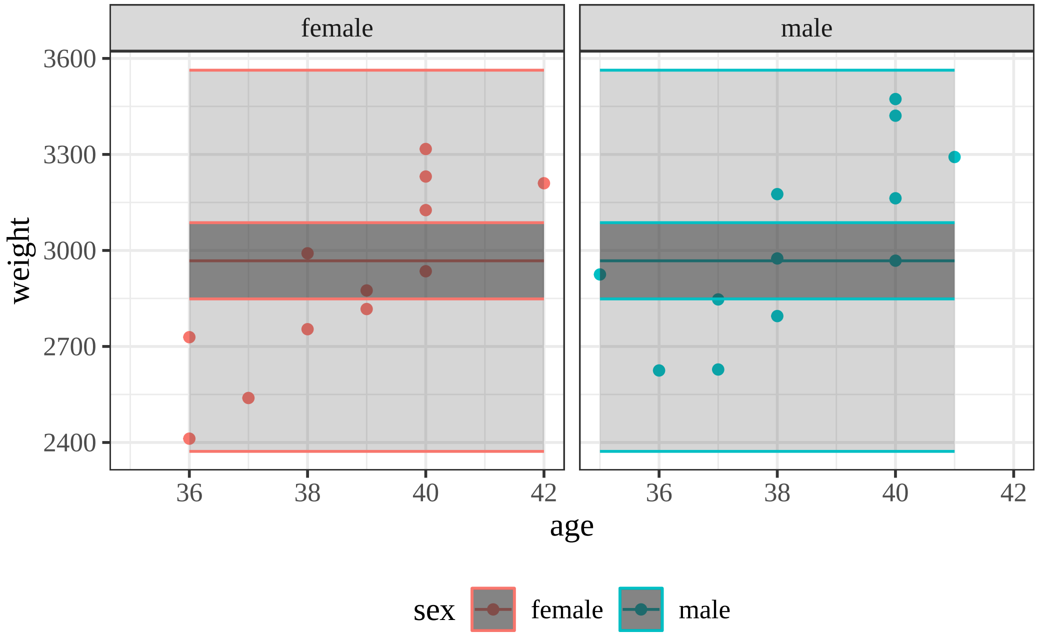


Figure 8: Null model for birthweight data, with 95% confidence and prediction intervals.

This model also has a likelihood:

```
logLik(lm0)
#> 'log Lik.' -168.955 (df=2)
```

And we can compare it to more complicated models using a likelihood ratio test:

```
lrtest(bw_lm2, lm0)
#> # A tibble: 2 x 5
#>   `#Df` LogLik    Df Chisq `Pr(>Chisq)`
#>   <dbl>  <dbl> <dbl> <dbl>        <dbl>
#> 1     5  -157.    NA  NA     NA
#> 2     2  -169.    -3  24.8    0.0000174
```

The likelihood ratio statistic for the test comparing the null model to the maximal model is

$$\lambda = 2 * (\ell_{\text{full}} - \ell_0) = 35.106732$$

where:

- ℓ_0 is the log-likelihood of the null model: -168.954967
- ℓ_{full} is the log-likelihood of the maximal model: -151.401601

In R, this test is:

```
lrtest(lm_max, lm0)
#> # A tibble: 2 x 5
#>   `#Df` LogLik    Df Chisq `Pr(>Chisq)`
#>   <dbl>  <dbl> <dbl> <dbl>        <dbl>
#> 1    13  -151.    NA  NA       NA
#> 2     2  -169.   -11  35.1     0.000238
```

This log-likelihood ratio statistic is called the **null deviance**. It tells us whether we have enough data to detect a difference between the null and full models.

1.6 Rescaling

1.6.1 Rescale age

```
bw <-
  bw |>
  mutate(
    `age - mean` = age - mean(age),
    `age - 36wks` = age - 36
  )

lm1_c <- lm(weight ~ sex + `age - 36wks`, data = bw)

lm2_c <- lm(weight ~ sex + `age - 36wks` + sex:`age - 36wks`, data = bw)

parameters(lm2_c, ci_method = "wald") |> print_md()
```

Parameter	Coefficient	SE	95% CI	t(20)	p
(Intercept)	2552.73	97.59	(2349.16, 2756.30)	26.16	< .001
sex (male)	209.97	129.75	(-60.68, 480.63)	1.62	0.121
age - 36wks	130.40	30.00	(67.82, 192.98)	4.35	< .001
sex (male) × age - 36wks	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

Compare with what we got without rescaling:

```
parameters(bw_lm2, ci_method = "wald") |> print_md()
```

Parameter	Coefficient	SE	95% CI	t(20)	p
(Intercept)	-2141.67	1163.60	(-4568.90, 285.56)	-1.84	0.081
sex (male)	872.99	1611.33	(-2488.18, 4234.17)	0.54	0.594
age	130.40	30.00	(67.82, 192.98)	4.35	< .001
sex (male) × age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

1.7 Prediction

1.7.1 Prediction for linear models

Definition 1.2 (Predicted value)**.** In a regression model $\text{p}(y|\tilde{x})$, the **predicted value** of y given $\tilde{x}$ is the estimated mean of Y given $\tilde{X} = \tilde{x}$:

$$\hat{y} \overset{\text{def}}{=} \hat{\text{E}}\left[Y|\tilde{X} = \tilde{x}\right]$$

For linear models, the predicted value can be straightforwardly calculated by multiplying each predictor value x_j by its corresponding coefficient β_j and adding up the results:

$$\begin{aligned}
\hat{y} &= \hat{\text{E}}\left[Y|\tilde{X} = \tilde{x}\right] \\
&= \tilde{x}'\hat{\beta} \\
&= \hat{\beta}_0 \cdot 1 + \hat{\beta}_1 x_1 + \ldots + \hat{\beta}_p x_p
\end{aligned}$$

1.7.2 Example: prediction for the `birthweight` data

```
x <- c(1, 1, 40)
sum(x * coef(bw_lm1))
#> [1] 3225.49
```

R has built-in functions for prediction:

```
x <- tibble(age = 40, sex = "male")
bw_lm1 |> predict(newdata = x)
#>       1
#> 3225.49
```

If you don't provide `newdata`, R will use the covariate values from the original dataset:

```
predict(bw_lm1)
#>       1       2       3       4       5       6       7       8       9      10
#> 3225.49 3062.45 2983.70 2578.87 3225.49 3062.45 2621.02 2820.66 2741.91 3304.24
#>      11      12      13      14      15      16      17      18      19      20
#> 2862.81 2941.56 3346.38 3062.45 3225.49 2699.77 2862.81 2578.87 2983.70 2820.66
#>      21      22      23      24
#> 3225.49 2941.56 2983.70 3062.45
```

These special predictions are called the *fitted values* of the dataset:

Definition 1.3. For a given dataset $(\tilde{Y}, \mathbf{X})$ and corresponding fitted model $\text{p}_{\hat{\beta}}(\tilde{y}|\mathbf{x})$, the **fitted value** of y_i is the predicted value of y when $\tilde{X} = \tilde{x}_i$ using the estimate parameters $\hat{\beta}$.

R has an extra function to get these values:

```
fitted(bw_lm1)
#>       1       2       3       4       5       6       7       8       9      10
#> 3225.49 3062.45 2983.70 2578.87 3225.49 3062.45 2621.02 2820.66 2741.91 3304.24
#>      11      12      13      14      15      16      17      18      19      20
#> 2862.81 2941.56 3346.38 3062.45 3225.49 2699.77 2862.81 2578.87 2983.70 2820.66
#>      21      22      23      24
#> 3225.49 2941.56 2983.70 3062.45
```

1.7.3 Confidence intervals

Use `predict(se.fit = TRUE)` to compute SEs for predicted values:

```
bw_lm1 |>
  predict(
    newdata = x,
    se.fit = TRUE
  )
#> $fit
#>       1
#> 3225.49
#>
#> $se.fit
#> [1] 61.4599
#>
#> $df
#> [1] 21
#>
#> $residual.scale
#> [1] 177.116
```

The output of `predict.lm(se.fit = TRUE)` is a `list()`; you can extract the elements with `$` or `magrittr::use_series()`:

```
library(magrittr)
bw_lm1 |>
  predict(
    newdata = x,
    se.fit = TRUE
  ) |>
  use_series(se.fit)
#> [1] 61.4599
```

We can construct **confidence intervals** for $\text{E}[Y|X = x]$ using the usual formula:

$$\mu(\tilde{x}) \in (\hat{\mu}(\tilde{x}) \pm \zeta_\alpha)$$

$$\zeta_\alpha = t_{n-p}\left(1 - \frac{\alpha}{2}\right) * \widehat{\text{se}}(\hat{\mu}(\tilde{x}))$$

$$\hat{\mu}(\tilde{x}) = \tilde{x} \cdot \hat{\beta}$$

$$\begin{aligned}
\text{se}(\hat{\mu}(\tilde{x})) &= \sqrt{\text{Var}(\hat{\mu}(\tilde{x}))} \\
\text{Var}(\hat{\mu}(\tilde{x})) &= \text{Var}\left(x'\hat{\beta}\right) \\
&= x'\text{Var}\left(\hat{\beta}\right)x \\
&= x'\sigma^2(\mathbf{X}'\mathbf{X})^{-1}x \\
&= \sigma^2 x'(\mathbf{X}'\mathbf{X})^{-1}x \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j \text{Cov}\left(\hat{\beta}_i, \hat{\beta}_j\right) \\
\widehat{\text{Var}}(\hat{\mu}(\tilde{x})) &= \hat{\sigma}^2 x'(\mathbf{X}'\mathbf{X})^{-1}x
\end{aligned}$$

```
bw_lm2 |> predict(
  newdata = x,
  interval = "confidence"
)
#>       fit     lwr     upr
#> 1 3210.64 3062.23 3359.05
```

```
library(sjPlot)
bw_lm2 |>
  plot_model(type = "pred", terms = c("age", "sex"), show.data = TRUE) +
  theme_sjplot() +
  theme(legend.position = "bottom")
```

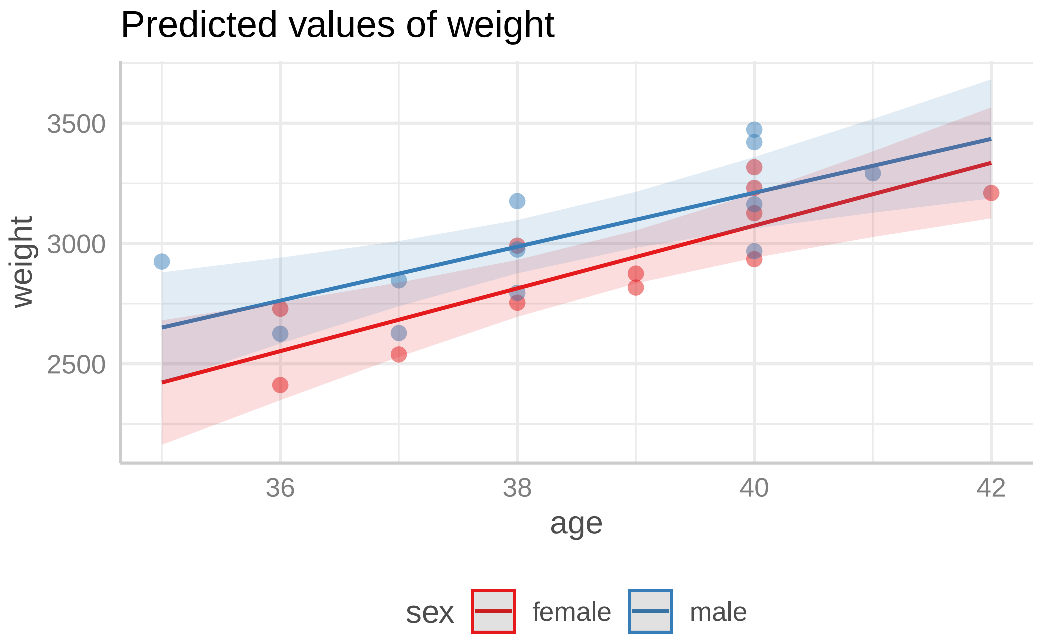


Figure 9: Predicted values and confidence bands for the `birthweight` model with interaction term

1.7.4 Prediction intervals

We can also construct **prediction intervals** for the value of a new observation Y^*, given a covariate pattern $\tilde{x}^*$:

$$
\begin{aligned}
\hat{Y^*} &= \hat{\mu}(\tilde{x}^*) + \hat{\epsilon}^* \\
\text{Var}\left(\hat{Y}\right) &= \text{Var}(\hat{\mu}) + \text{Var}(\hat{\epsilon}) \\
\text{Var}\left(\hat{Y}\right) &= \text{Var}\left(x'\hat{\beta}\right) + \text{Var}(\hat{\epsilon}) \\
&= \tilde{x}^{*\top}\text{Var}\left(\hat{\beta}\right)\tilde{x}^* + \sigma^2 \\
&= \tilde{x}^{*\top}(\sigma^2(\mathbf{X}'\mathbf{X})^{-1})\tilde{x}^* + \sigma^2 \\
&= \sigma^2\tilde{x}^{*\top}(\mathbf{X}'\mathbf{X})^{-1}\tilde{x}^* + \sigma^2
\end{aligned}
\tag{9}
$$

See Hogg, Tanis, and Zimmerman (2015) §7.6 (p. 340) for a longer version.

```
bw_lm2 |>
  predict(newdata = x, interval = "predict")
#>       fit     lwr     upr
#> 1 3210.64 2805.71 3615.57
```

If you don't specify `newdata`, you get a warning:

```
bw_lm2 |>
  predict(interval = "predict") |>
  head()
#> Warning in predict.lm(bw_lm2, interval = "predict"): predictions on current data refer to _future
#>       fit     lwr     upr
#> 1 2552.73 2124.50 2980.97
#> 2 2552.73 2124.50 2980.97
#> 3 2683.13 2275.99 3090.27
#> 4 2813.53 2418.60 3208.47
#> 5 2813.53 2418.60 3208.47
#> 6 2943.93 2551.48 3336.38
```

The warning from the last command is: "predictions on current data refer to *future* responses" (since you already know what happened to the current data, and thus don't need to predict it).

See `?predict.lm` for more.

```
plot_PIs_and_CIs(bw_lm2, bw)
```

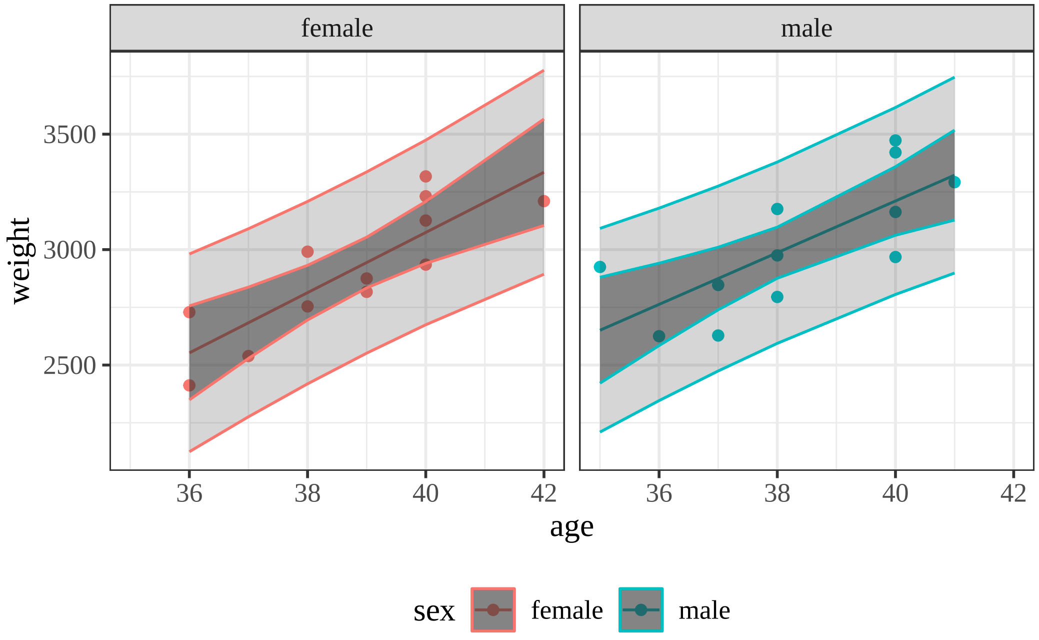


Figure 10: Confidence and prediction intervals for `birthweight` model 2

1.8 Diagnostics

Tip

This section is adapted from Dobson and Barnett (2018, secs. 6.2–6.3) and Dunn and Smyth (2018) Chapter 3[a].

[a] https://link.springer.com/chapter/10.1007/978-1-4419-0118-7_3

1.8.1 Assumptions in linear regression models

$$Y_i|\tilde{X}_i \sim_{\perp\!\!\!\perp} \mathrm{N}(\mu_i, \sigma^2)$$

$$\mu_i = \tilde{x} \cdot \tilde{\beta}$$

1. Normality

The model assumes that the distribution conditional on a given X value is Gaussian.

2. Correct Functional Form of Conditional Mean Structure (Linear Component)

The model assumes that the conditional means have the structure:

$$\mathrm{E}[Y|\tilde{X} = \tilde{x}] = \tilde{x}'\tilde{\beta}$$

3. Homoskedasticity

The model assumes that variance σ^2 is constant (with respect to $\tilde{x}$).

4. Independence

The model assumes that the observations are statistically independent.

1.8.2 Direct visualization

The most direct way to examine the fit of a model is to compare it to the raw observed data.

```r
bw <-
  bw |>
  mutate(
    predlm2 = predict(bw_lm2)
  ) |>
  arrange(sex, age)

plot1_interact <-
  plot1 %+% bw +
  geom_line(aes(y = predlm2))

print(plot1_interact)
```

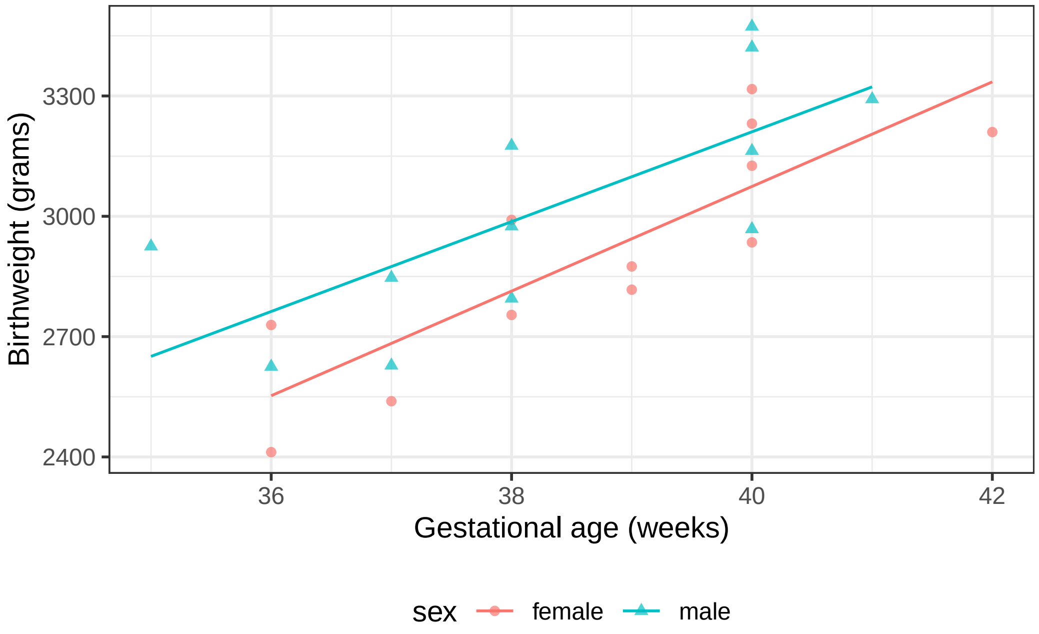


Figure 11: Birthweight model with interaction term

It's not easy to assess these assumptions from this model. If there are multiple continuous covariates, it becomes even harder to visualize the raw data.

Table 28: hers data

```
hers <- fs::path_package("rme", "extdata/hersdata.dta") |>
  haven::read_dta()
hers
#> # A tibble: 2,763 x 37
#>    HT              age raceth  nonwhite smoking drinkany exercise physact globrat
#>    <dbl+lbl>     <dbl> <dbl+l> <dbl+lb> <dbl+l> <dbl+lb> <dbl+lb> <dbl+l> <dbl+l>
#>  1 0 [placebo]     70 2 [Afr~ 1 [yes]  0 [no]  0 [no]   0 [no]   5 [muc~ 3 [goo~
#>  2 0 [placebo]     62 2 [Afr~ 1 [yes]  0 [no]  0 [no]   0 [no]   1 [muc~ 3 [goo~
#>  3 1 [hormone ~    69 1 [Whi~ 0 [no]   0 [no]  0 [no]   0 [no]   3 [abo~ 3 [goo~
#>  4 0 [placebo]     64 1 [Whi~ 0 [no]   1 [yes] 1 [yes]  0 [no]   1 [muc~ 3 [goo~
#>  5 0 [placebo]     65 1 [Whi~ 0 [no]   0 [no]  0 [no]   0 [no]   2 [som~ 3 [goo~
#>  6 1 [hormone ~    68 2 [Afr~ 1 [yes]  0 [no]  1 [yes]  0 [no]   3 [abo~ 3 [goo~
#>  7 0 [placebo]     70 1 [Whi~ 0 [no]   0 [no]  0 [no]   0 [no]   3 [abo~ 2 [fai~
#>  8 1 [hormone ~    69 1 [Whi~ 0 [no]   0 [no]  0 [no]   1 [yes]  5 [muc~ 4 [ver~
#>  9 1 [hormone ~    61 1 [Whi~ 0 [no]   0 [no]  1 [yes]  1 [yes]  3 [abo~ 4 [ver~
#> 10 1 [hormone ~    62 1 [Whi~ 0 [no]   1 [yes] 1 [yes]  0 [no]   2 [som~ 3 [goo~
#> # i 2,753 more rows
#> # i 28 more variables: poorfair <dbl+lbl>, medcond <dbl>, htnmeds <dbl+lbl>,
#> #   statins <dbl+lbl>, diabetes <dbl+lbl>, dmpills <dbl+lbl>,
#> #   insulin <dbl+lbl>, weight <dbl>, BMI <dbl>, waist <dbl>, WHR <dbl>,
#> #   glucose <dbl>, weight1 <dbl>, BMI1 <dbl>, waist1 <dbl>, WHR1 <dbl>,
#> #   glucose1 <dbl>, tchol <dbl>, LDL <dbl>, HDL <dbl>, TG <dbl>, tchol1 <dbl>,
#> #   LDL1 <dbl>, HDL1 <dbl>, TG1 <dbl>, SBP <dbl>, DBP <dbl>, age10 <dbl>
```

Fitted model for hers data

Consider the hers data from Vittinghoff et al. (2012).

The "heart and estrogen/progestin study" (HERS) was a clinical trial of hormone therapy for prevention of recurrent heart attacks and death among 2,763 post-menopausal women with existing coronary heart disease (CHD) (Hulley et al. 1998).

Suppose we consider models with and without intercept terms (i.e., possibly forcing the intercept to go through 0):

```
hers_lm_with_int <- lm(
  na.action = na.exclude,
  LDL ~ smoking * age, data = hers
)


library(equatiomatic)
equatiomatic::extract_eq(hers_lm_with_int)
```

$$\text{LDL} = \alpha + \beta_1(\text{smoking}) + \beta_2(\text{age}) + \beta_3(\text{smoking} \times \text{age}) + \epsilon \tag{10}$$

```
hers_lm_no_int <- lm(
  na.action = na.exclude,
  LDL ~ age + smoking:age - 1, data = hers
)

library(equatiomatic)
equatiomatic::extract_eq(hers_lm_no_int)
```

Table 29: `hers` data models with and without intercepts

(a) With intercept

```r
library(gtsummary)
hers_lm_with_int |>
  tbl_regression(intercept = TRUE)
```

Characteristic	**Beta**	**95% CI**	**p-value**
(Intercept)	154	138, 170	<0.001
current smoker	54	15, 94	0.007
age in years	-0.14	-0.38, 0.09	0.2
current smoker * age in years	-0.79	-1.4, -0.17	0.012

Abbreviation: CI = Confidence Interval

(b) No intercept

```r
hers_lm_no_int |>
  tbl_regression(intercept = TRUE)
```

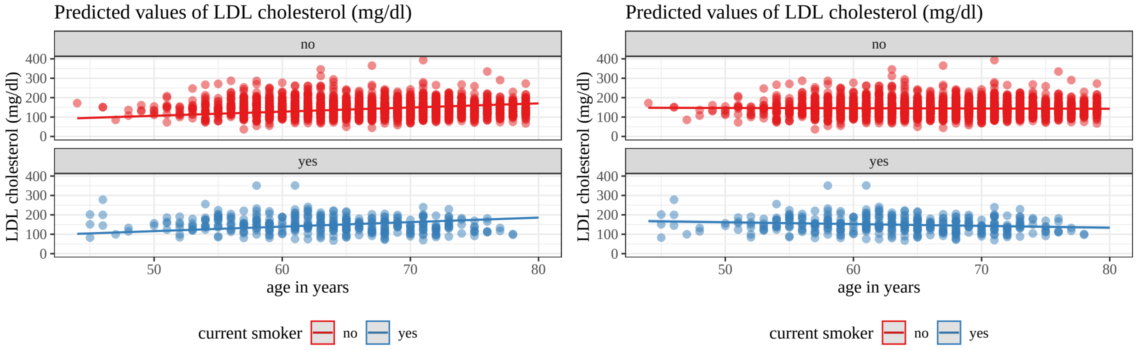
Characteristic	**Beta**	**95% CI**	**p-value**
age in years	2.1	2.1, 2.2	<0.001
age in years * current smoker	0.19	0.12, 0.26	<0.001

Abbreviation: CI = Confidence Interval

$$\text{LDL} = \beta_1(\text{age}) + \beta_2(\text{age} \times \text{age}_{\text{smoking}}) + \epsilon \tag{11}$$

```r
library(sjPlot)

hers_plot1 <- hers_lm_no_int |>
  sjPlot::plot_model(
    type = "pred",
    terms = c("age", "smoking"),
    show.data = TRUE
  ) +
  facet_wrap(~group_col, ncol = 1) +
  expand_limits(y = 0) +
  theme(legend.position = "bottom")

hers_plot1
```

```r
library(sjPlot)

hers_plot2 <- hers_lm_with_int |>
  sjPlot::plot_model(
    type = "pred",
    terms = c("age", "smoking"),
    show.data = TRUE
  ) +
  facet_wrap(~group_col, ncol = 1) +
  expand_limits(y = 0) +
  theme(legend.position = "bottom")

hers_plot2
```



(a) No intercept

(b) With intercept

Figure 12: `hers` data models with and without intercepts

1.8.3 Residuals

Maybe we can transform the data and model in some way to make it easier to inspect.

Definition 1.4 (Residual noise/deviation from the population mean)**.** The **residual noise** in a probabilistic model $p(Y)$, also known as the **residual deviation of an observation from its**

population mean or **residual** for short, is the difference between an observed value y and its population mean:

$$\varepsilon(y) \stackrel{\text{def}}{=} y - \text{E}[Y] \tag{12}$$

We use the same notation for residual noise that we used for errors[8].

$\text{E}[Y]$ can be viewed as an estimate of Y, before y is observed. Conversely, each observation y can be viewed as an estimate of $\text{E}[Y]$ (albeit an imprecise one, individually, since $n = 1$).

We can rearrange Equation 12 to view y as the sum of its mean plus the residual noise:

$$y = \text{E}[Y] + \varepsilon(y)$$

Theorem 1.1 (Residuals in Gaussian models)**.** *If Y has a Gaussian distribution, then $\varepsilon(Y)$ also has a Gaussian distribution, and vice versa.*

Proof. Left to the reader. □

Definition 1.5 (Residuals of a fitted model value)**.** The **residual of a fitted value** $\hat{y}$ (shorthand: "residual") is its error[9] relative to the observed data:

$$\begin{aligned} e(\hat{y}) &\stackrel{\text{def}}{=} \varepsilon(\hat{y}) \\ &= y - \hat{y} \end{aligned}$$

Example 1.3 (residuals in `birthweight` data)**.**

[8] estimation.qmd#def-error

[9] estimation.qmd#def-error

```r
plot1_interact +
  facet_wrap(~sex) +
  geom_segment(
    aes(
      x = age,
      y = predlm2,
      xend = age,
      yend = weight,
      col = sex,
      group = id
    ),
    linetype = "dotted"
  )
```

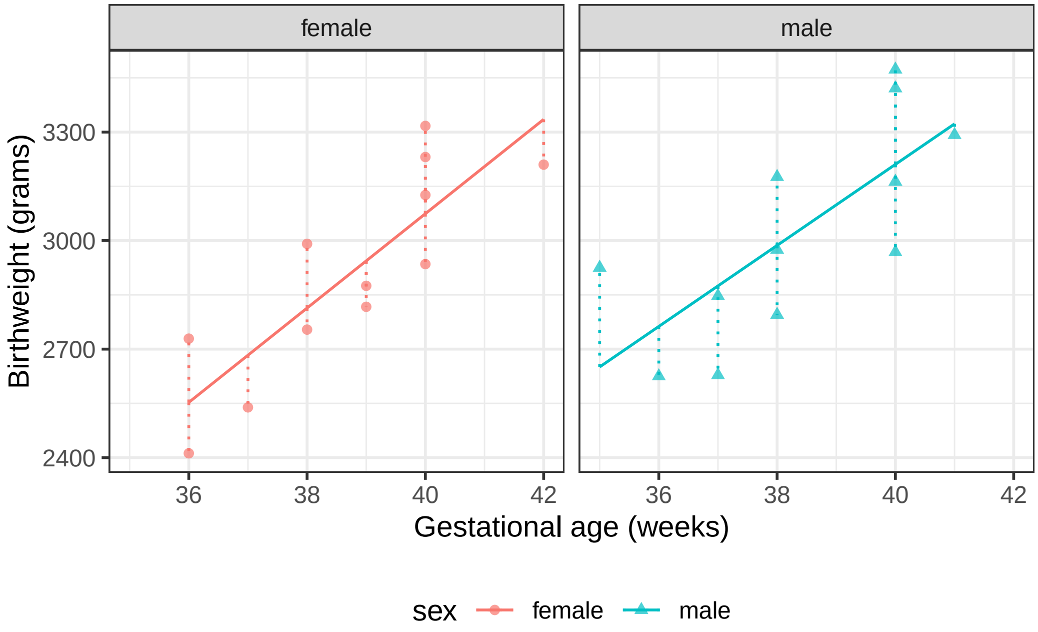


Figure 13: Fitted values and residuals for interaction model for `birthweight` data

---

Residuals of fitted values vs residual noise

$e(\hat{y})$ can be seen as the maximum likelihood estimate of the residual noise:

$$\begin{aligned} e(\hat{y}) &= y - \hat{y} \\ &= \hat{\varepsilon}_{ML} \end{aligned}$$

General characteristics of residuals

Theorem 1.2. *If $\hat{E}[Y]$ is an unbiased*[10] *estimator of the mean $E[Y]$, then:*

$$E[e(y)] = 0 \tag{13}$$

$$Var(e(y)) \approx \sigma^2 \tag{14}$$

---

[10] estimation.qmd#sec-unbiased-estimators

Proof.

Equation 13:

$$
\begin{aligned}
\mathrm{E}[e(y)] &= \mathrm{E}[y - \hat{y}] \\
&= \mathrm{E}[y] - \mathrm{E}[\hat{y}] \\
&= \mathrm{E}[y] - \mathrm{E}[y] \\
&= 0
\end{aligned}
$$

Equation 14:

$$
\begin{aligned}
\mathrm{Var}(e(y)) &= \mathrm{Var}(y - \hat{y}) \\
&= \mathrm{Var}(y) + \mathrm{Var}(\hat{y}) - 2\mathrm{Cov}(y, \hat{y}) \\
&\dot{\approx} \mathrm{Var}(y) + 0 - 2 \cdot 0 \\
&= \mathrm{Var}(y) \\
&= \sigma^2
\end{aligned}
$$

□

Characteristics of residuals in Gaussian models

With enough data and a correct model, the residuals will be approximately Guassian distributed, with variance σ^2, which we can estimate using $\hat{\sigma}^2$; that is:

$$e_i \ \sim_{\text{iid}} \ N(0, \hat{\sigma}^2)$$

Computing residuals in R

R provides a function for residuals:

```
resid(bw_lm2)
#>         1         2         3         4         5         6         7         8 
#>  176.2667 -140.7333 -144.1333  -59.5333  177.4667 -126.9333  -68.9333  242.6667 
#>         9        10        11        12        13        14        15        16 
#> -139.3333   51.6667  156.6667 -125.1333  274.2759 -137.7069  -27.6897 -246.6897 
#>        17        18        19        20        21        22        23        24 
#> -191.6724  189.3276  -11.6724 -242.6379  -47.6379  262.3621  210.3621  -30.6207
```

Exercise 1.13. Check R's output by computing the residuals directly.

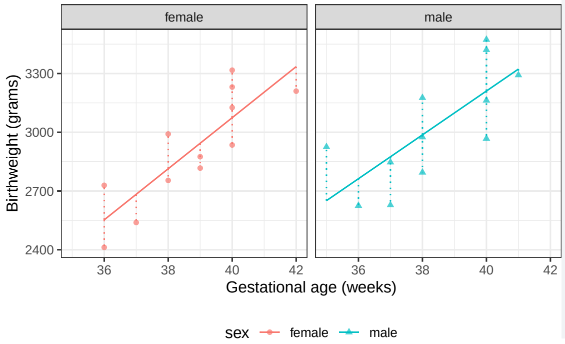
Solution.

```
bw$weight - fitted(bw_lm2)
#>         1         2         3         4         5         6         7         8 
#>  176.2667 -140.7333 -144.1333  -59.5333  177.4667 -126.9333  -68.9333  242.6667 
#>         9        10        11        12        13        14        15        16 
#> -139.3333   51.6667  156.6667 -125.1333  274.2759 -137.7069  -27.6897 -246.6897 
#>        17        18        19        20        21        22        23        24 
#> -191.6724  189.3276  -11.6724 -242.6379  -47.6379  262.3621  210.3621  -30.6207
```

This matches R's output!

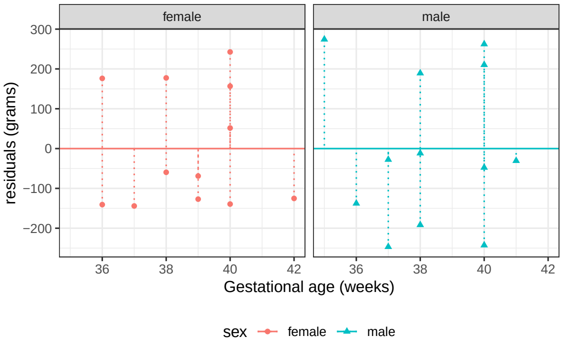
Graphing the residuals

```
plot1_interact +
  facet_wrap(~sex) +
  geom_segment(
    aes(
      x = age,
      y = predlm2,
      xend = age,
      yend = weight,
      col = sex,
      group = id
    ),
    linetype = "dotted"
  )
```



(a) fitted values

```
bw <- bw |>
  mutate(
    resids_intxn =
      weight - fitted(bw_lm2)
  )

plot_bw_resid <-
  bw |>
  ggplot(aes(
    x = age,
    y = resids_intxn,
    linetype = sex,
    shape = sex,
    col = sex
  )) +
  theme_bw() +
  xlab("Gestational age (weeks)") +
  ylab("residuals (grams)") +
  theme(legend.position = "bottom") +
  geom_hline(aes(
    yintercept = 0,
    col = sex
  )) +
  geom_segment(
    aes(yend = 0),
    linetype = "dotted"
  ) +
  geom_point()
# expand_limits(y = 0, x = 0) +
geom_point(alpha = .7)
#> geom_point: na.rm = FALSE
#> stat_identity: na.rm = FALSE
#> position_identity
print(plot_bw_resid + facet_wrap(~sex))
```



(b) Residuals

Figure 14: Fitted values and residuals for interaction model for `birthweight` data

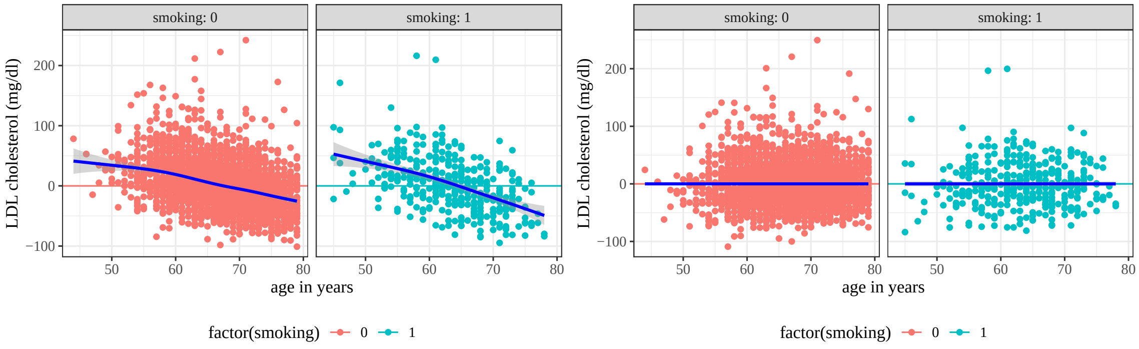
---

Residuals versus predictors

```
hers <- hers |>
  mutate(
    resids_no_intcpt =
      LDL - fitted(hers_lm_no_int),
    resids_with_intcpt =
      LDL - fitted(hers_lm_with_int)
  )
```

```
hers |>
  arrange(age) |>
  ggplot() +
  aes(x = age, y = resids_no_intcpt, col
  geom_point() +
  geom_hline(aes(yintercept = 0, col = fa
  facet_wrap(~smoking, labeller = "label_
  theme(legend.position = "bottom") +
  geom_smooth(col = "blue")
```

```
hers |>
  arrange(age) |>
  ggplot() +
  aes(x = age, y = resids_with_intcpt, col = factor(smoking
  geom_point() +
  geom_hline(aes(yintercept = 0, col = factor(smoking))) +
  facet_wrap(~smoking, labeller = "label_both") +
  theme(legend.position = "bottom") +
  geom_smooth(col = "blue")
```



(a) no intercept

(b) with intercept

Figure 15: Residuals of `hers` data vs predictors

Residuals versus fitted values

If the model contains multiple continuous covariates, how do we check for errors in the mean structure assumption?

```
library(ggfortify)
hers_lm_no_int |>
  update(na.action = na.omit) |>
  autoplot(
    which = 1,
    ncol = 1,
    smooth.colour = NA
  ) +
  geom_hline(yintercept = 0, col = "red")
```

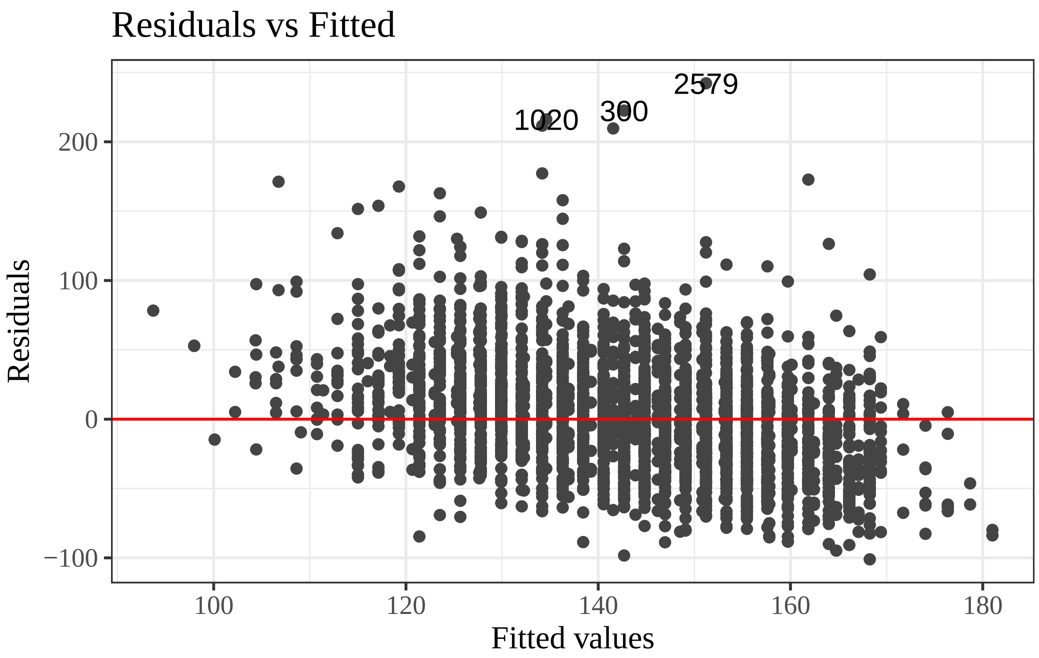


Figure 16: Residuals of interaction model for `hers` data

We can add a LOESS smooth to visualize where the residual mean is nonzero:

```r
library(ggfortify)
hers_lm_no_int |>
  update(na.action = na.omit) |>
  autoplot(
    which = 1,
    ncol = 1
  ) +
  geom_hline(yintercept = 0, col = "red")
```

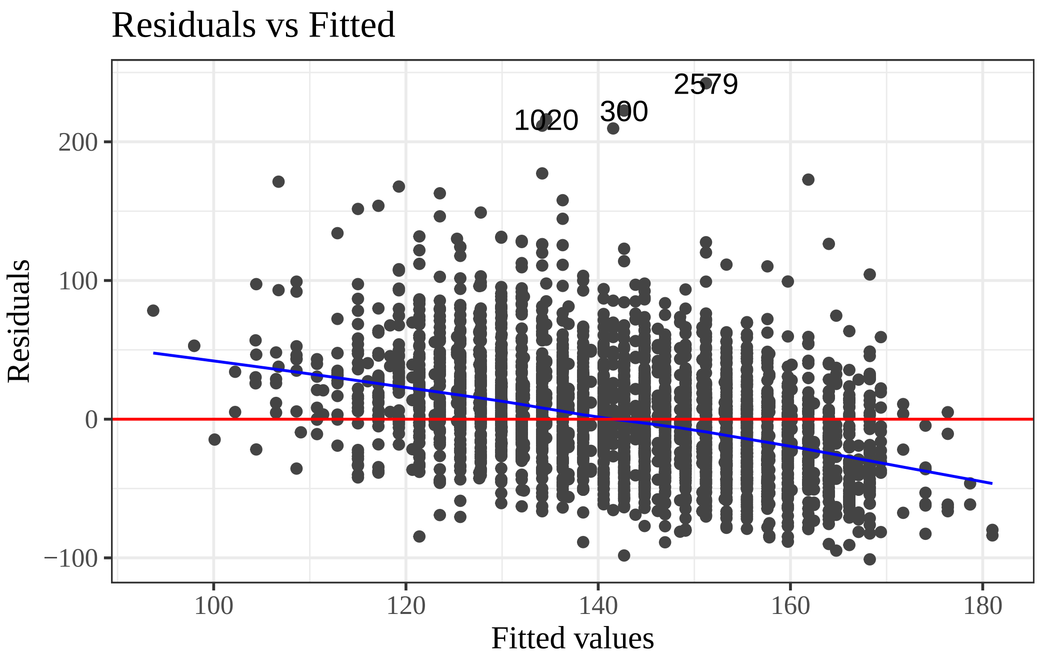
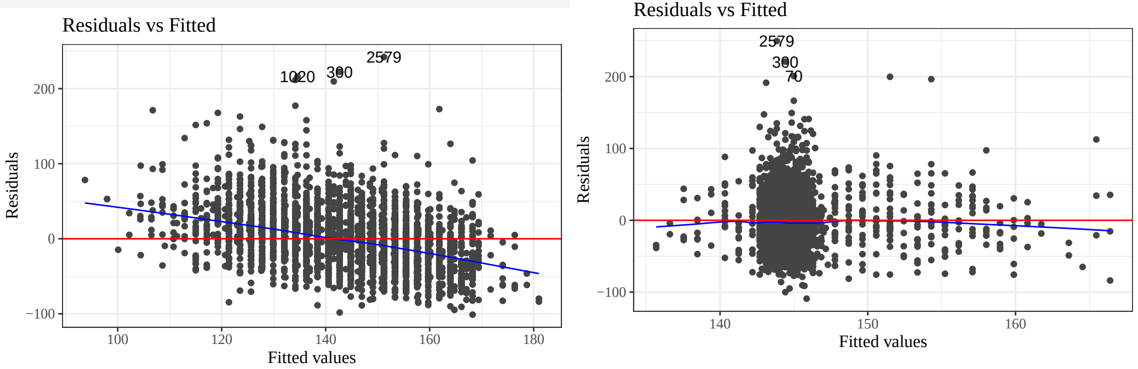


Figure 17: Residuals of interaction model for `hers` data, no intercept term

---

```r
library(ggfortify)
hers_lm_no_int |>
  update(na.action = na.omit) |>
  autoplot(
    which = 1,
    ncol = 1
  ) +
  geom_hline(yintercept = 0, col = "red")
```

```r
hers_lm_with_int |>
  update(na.action = na.omit) |>
  autoplot(
    which = 1,
    ncol = 1
  ) +
  geom_hline(yintercept = 0, col = "red")
```



(a) no intercept term

(b) with intercept term

Figure 18: Residuals of interaction model for `hers` data, with and without intercept term

---

Definition 1.6 (Standardized residuals)**.**

$$r_i = \frac{e_i}{\widehat{SD}(e_i)}$$

Hence, with enough data and a correct model, the standardized residuals will be approximately standard Gaussian; that is,

$$r_i \sim_{\text{iid}} N(0,1)$$

1.8.4 Marginal distributions of residuals

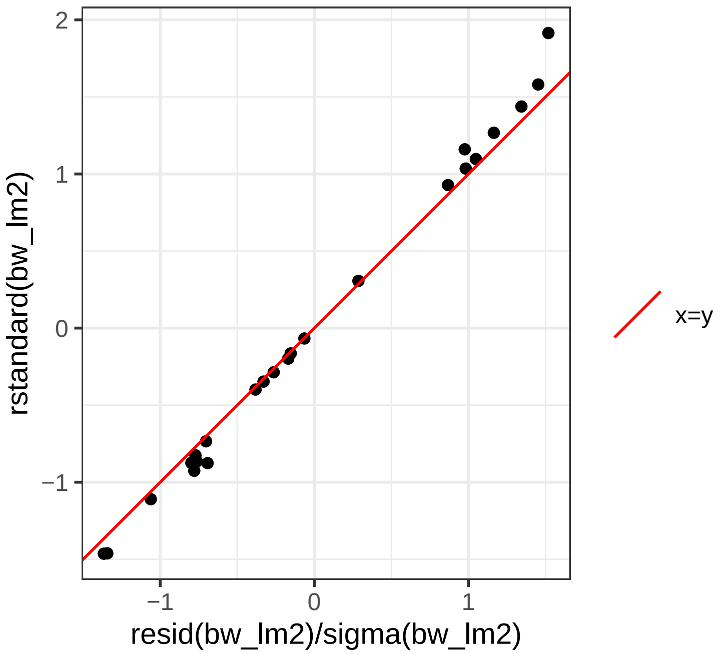
To look for problems with our model, we can check whether the residuals e_i and standardized residuals r_i look like they have the distributions that they are supposed to have, according to the model.

Standardized residuals in R

```
rstandard(bw_lm2)
#>          1          2          3          4          5          6          7
#>  1.1598166 -0.9260109 -0.8747917 -0.3472255  1.0350665 -0.7347315 -0.3990086
#>          8          9         10         11         12         13         14
#>  1.4375164 -0.8253872  0.3060646  0.9280669 -0.8761592  1.9142780 -0.8655921
#>         15         16         17         18         19         20         21
#> -0.1642993 -1.4637574 -1.1101599  1.0965787 -0.0676062 -1.4615865 -0.2869582
#>         22         23         24
#>  1.5803994  1.2671652 -0.1980543
resid(bw_lm2) / sigma(bw_lm2)
#>          1          2          3          4          5          6          7
#>  0.9759331 -0.7791962 -0.7980209 -0.3296173  0.9825771 -0.7027900 -0.3816622
#>          8          9         10         11         12         13         14
#>  1.3435690 -0.7714449  0.2860621  0.8674141 -0.6928239  1.5185792 -0.7624398
#>         15         16         17         18         19         20         21
#> -0.1533089 -1.3658431 -1.0612299  1.0482473 -0.0646265 -1.3434099 -0.2637562
#>         22         23         24
#>  1.4526163  1.1647086 -0.1695371
```

These are not quite the same, because R is doing something more complicated and precise to get the standard errors. Let's not worry about those details for now; the difference is pretty small in this case:

```
rstandard_compare_plot <-
  tibble(
    x = resid(bw_lm2) / sigma(bw_lm2),
    y = rstandard(bw_lm2)
  ) |>
  ggplot(aes(x = x, y = y)) +
  geom_point() +
  theme_bw() +
  coord_equal() +
  xlab("resid(bw_lm2)/sigma(bw_lm2)") +
  ylab("rstandard(bw_lm2)") +
  geom_abline(
    aes(
      intercept = 0,
      slope = 1,
      col = "x=y"
    )
  ) +
  labs(colour = "") +
  scale_colour_manual(values = "red")

print(rstandard_compare_plot)
```



Let’s add these residuals to the `tibble` of our dataset:

```
bw <-
  bw |>
  mutate(
    fitted_lm2 = fitted(bw_lm2),
    resid_lm2 = resid(bw_lm2),
    resid_lm2_alt = weight - fitted_lm2,
    std_resid_lm2 = rstandard(bw_lm2),
    std_resid_lm2_alt = resid_lm2 / sigma(bw_lm2)
  )

bw |>
  select(
    sex,
    age,
    weight,
    fitted_lm2,
    resid_lm2,
    std_resid_lm2
  )
#> # A tibble: 24 x 6
#>    sex      age weight fitted_lm2 resid_lm2 std_resid_lm2
#>    <fct>  <dbl>  <dbl>      <dbl>     <dbl>         <dbl>
#>  1 female    36   2729      2553.     176.          1.16
#>  2 female    36   2412      2553.    -141.         -0.926
#>  3 female    37   2539      2683.    -144.         -0.875
#>  4 female    38   2754      2814.     -59.5        -0.347
#>  5 female    38   2991      2814.     177.          1.04
#>  6 female    39   2817      2944.    -127.         -0.735
#>  7 female    39   2875      2944.     -68.9        -0.399
#>  8 female    40   3317      3074.     243.          1.44
#>  9 female    40   2935      3074.    -139.         -0.825
#> 10 female    40   3126      3074.      51.7         0.306
#> # i 14 more rows
```

Now let's build histograms:

```
resid_marginal_hist <-
  bw |>
  ggplot(aes(x = resid_lm2)) +
  geom_histogram()

print(resid_marginal_hist)
```

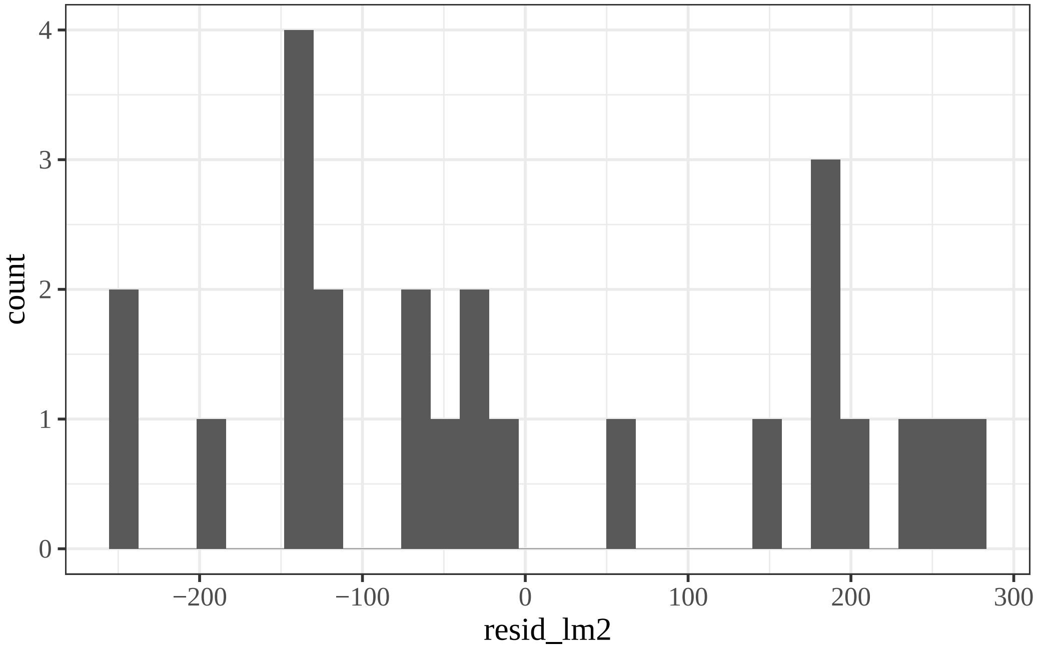


Figure 19: Marginal distribution of (nonstandardized) residuals

Hard to tell with this small amount of data, but I'm a bit concerned that the histogram doesn't show a bell-curve shape.

```
std_resid_marginal_hist <-
  bw |>
  ggplot(aes(x = std_resid_lm2)) +
  geom_histogram()

print(std_resid_marginal_hist)
```

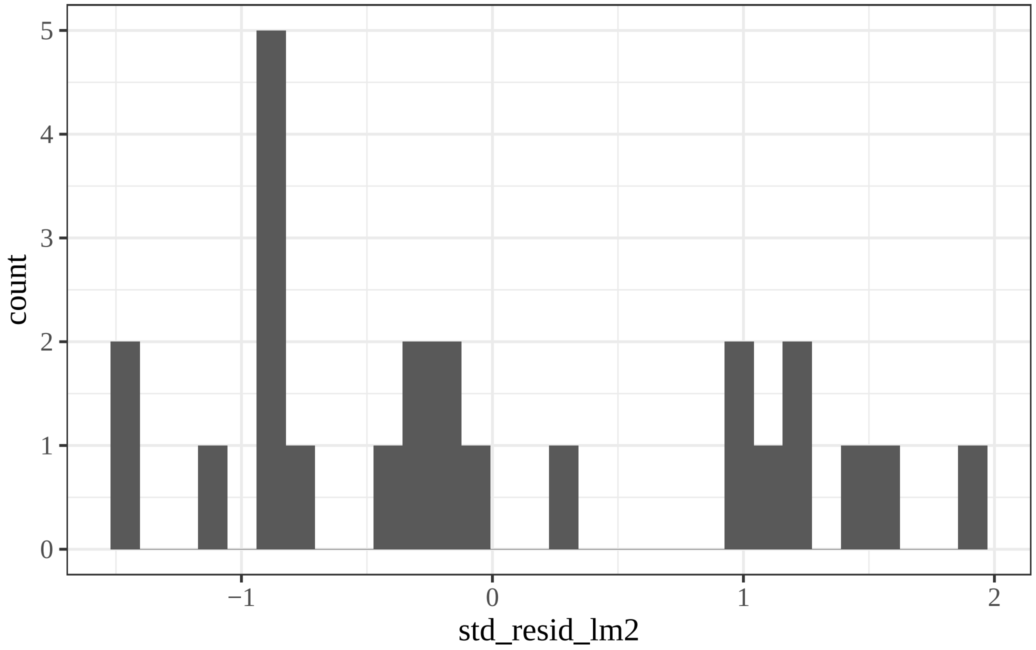


Figure 20: Marginal distribution of standardized residuals

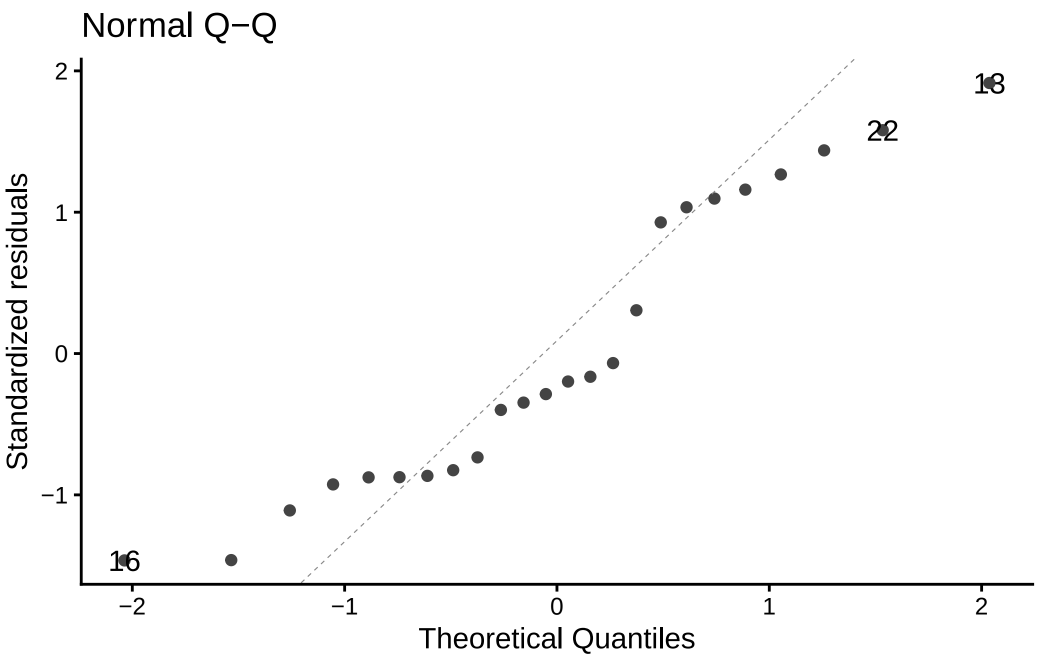
This looks similar, although the scale of the x-axis got narrower, because we divided by $\hat{\sigma}$ (roughly speaking).

Still hard to tell if the distribution is Gaussian.

1.8.5 QQ plot of standardized residuals

Another way to assess normality is the QQ plot of the standardized residuals versus normal quantiles:

```r
library(ggfortify)
# needed to make ggplot2::autoplot() work for `lm` objects

qqplot_lm2_auto <-
  bw_lm2 |>
  autoplot(
    which = 2, # options are 1:6; can do multiple at once
    ncol = 1
  ) +
  theme_classic()

print(qqplot_lm2_auto)
```



If the Gaussian model were correct, these points should follow the dotted line.

Fig 2.4 panel (c) in Dobson and Barnett (2018) is a little different; they didn't specify how they produced it, but other statistical analysis systems do things differently from R.

See also Dunn and Smyth (2018) §3.5.4[11].

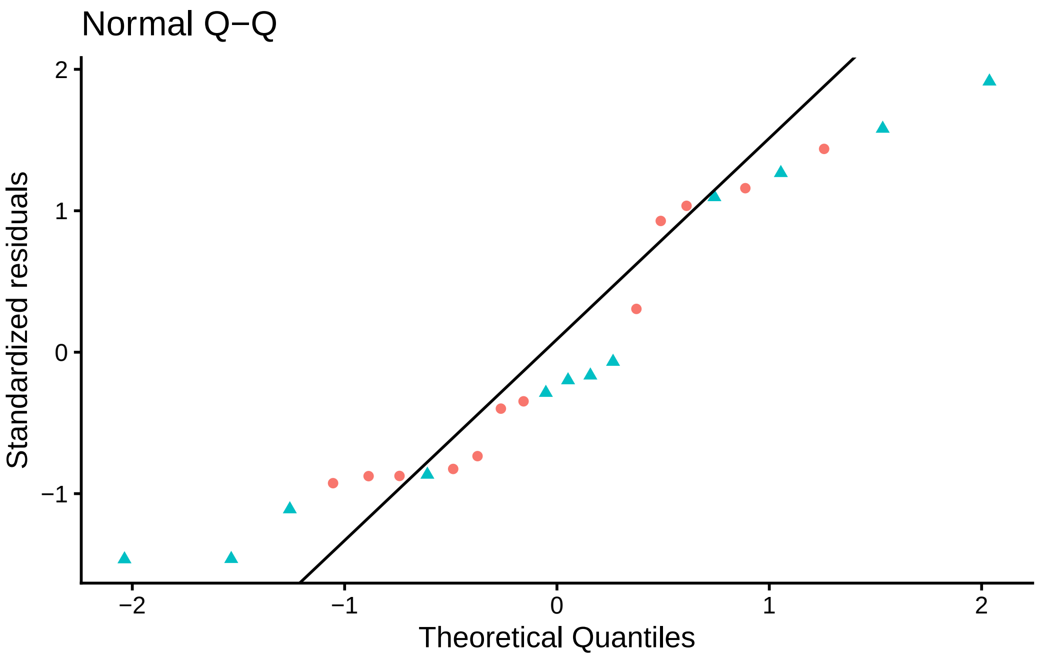
QQ plot - how it's built

Let's construct it by hand:

```r
bw <- bw |>
  mutate(
    p = (rank(std_resid_lm2) - 1 / 2) / n(), # "Blom's method"
    expected_quantiles_lm2 = qnorm(p)
  )

qqplot_lm2 <-
  bw |>
  ggplot(
    aes(
      x = expected_quantiles_lm2,
      y = std_resid_lm2,
      col = sex,
      shape = sex
    )
  ) +
  geom_point() +
  theme_classic() +
  theme(legend.position = "none") + # removing the plot legend
  ggtitle("Normal Q-Q") +
  xlab("Theoretical Quantiles") +
  ylab("Standardized residuals")
```

[11] https://link.springer.com/chapter/10.1007/978-1-4419-0118-7_3#Sec14:~:text=3.5.4%20Q%E2%80%93Q%20Plots%20and%20Normality

```
# find the expected line:

ps <- c(.25, .75) # reference probabilities
a <- quantile(rstandard(bw_lm2), ps) # empirical quantiles
b <- qnorm(ps) # theoretical quantiles

qq_slope <- diff(a) / diff(b)
qq_intcpt <- a[1] - b[1] * qq_slope

qqplot_lm2 <-
  qqplot_lm2 +
  geom_abline(slope = qq_slope, intercept = qq_intcpt)

print(qqplot_lm2)
```



1.8.6 Conditional distributions of residuals

If our Gaussian linear regression model is correct, the residuals e_i and standardized residuals r_i should have:

- an approximately Gaussian distribution, with:
- a mean of 0
- a constant variance

This should be true **for every** value of x.

If we didn't correctly guess the functional form of the linear component of the mean,

$$\text{E}[Y|X = x] = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p$$

Then the residuals might have nonzero mean.

Regardless of whether we guessed the mean function correctly, ther the variance of the residuals might differ between values of x.

Residuals versus fitted values

To look for these issues, we can plot the residuals e_i against the fitted values $\hat{y}_i$ (Figure 21).

```r
autoplot(bw_lm2, which = 1, ncol = 1) |> print()
```

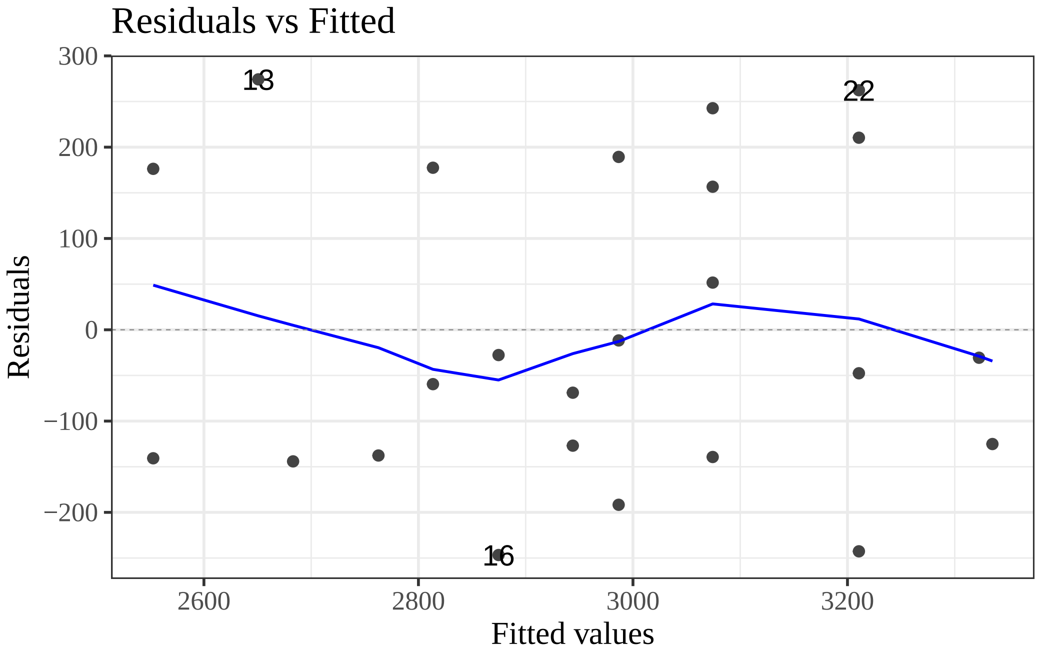


Figure 21: `birthweight` model (Equation 2): residuals versus fitted values

If the model is correct, the blue line should stay flat and close to 0, and the cloud of dots should have the same vertical spread regardless of the fitted value.

If not, we probably need to change the functional form of linear component of the mean,

$$\text{E}[Y|X=x] = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p$$

Example: PLOS Medicine title length data

(Adapted from Dobson and Barnett (2018), §6.7.1)

```r
data(PLOS, package = "dobson")
library(ggplot2)
fig1 =
  PLOS |>
  ggplot(
    aes(x = authors,
        y = nchar)
  ) +
  geom_point() +
  theme(legend.position = "bottom") +
  labs(col = "") +
  guides(col=guide_legend(ncol=3))
fig1
```

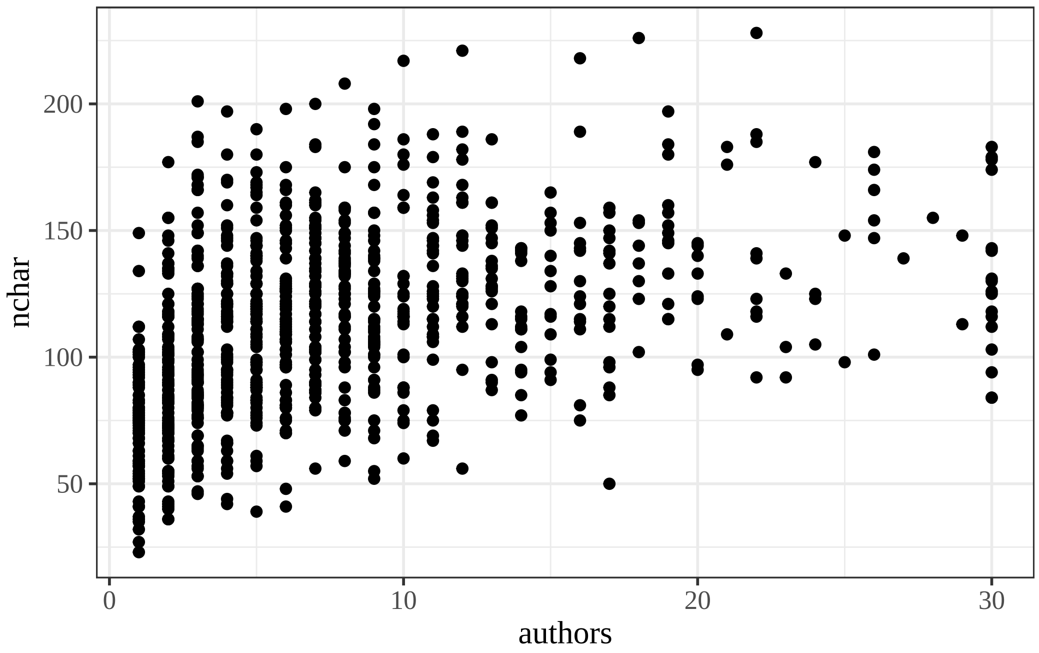


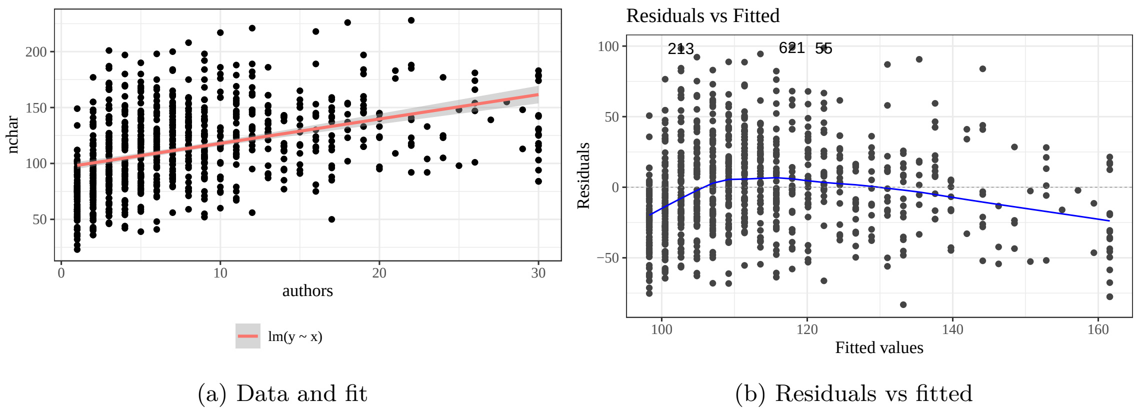
Figure 22: Number of authors versus title length in *PLOS Medicine* articles

---

Linear fit

```r
lm_PLOS_linear = lm(
  formula = nchar ~ authors,
  data = PLOS)
```

```r
fig2 = fig1 +
  geom_smooth(
    method = "lm",
              fullrange = TRUE,
              aes(col = "lm(y ~ x)"))
fig2

library(ggfortify)
autoplot(lm_PLOS_linear, which = 1, ncol = 1)
```



(a) Data and fit

(b) Residuals vs fitted

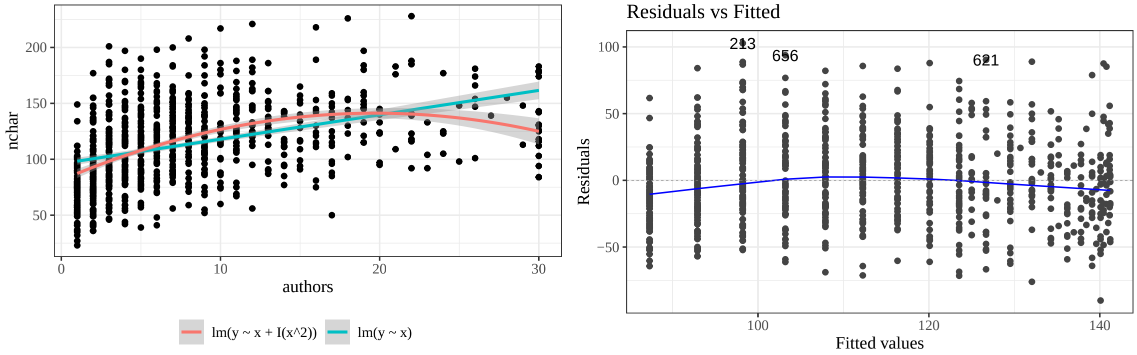
Figure 23: Number of authors versus title length in *PLOS Medicine*, with linear model fit

---

Quadratic fit

```r
lm_PLOS_quad = lm(
  formula = nchar ~ authors + I(authors^2),
  data = PLOS)
```

```r
fig3 =
  fig2 +
geom_smooth(
    method = "lm",
    fullrange = TRUE,
    formula = y ~ x + I(x ^ 2),
    aes(col = "lm(y ~ x + I(x^2))")
  )
fig3

autoplot(lm_PLOS_quad, which = 1, ncol = 1)
```



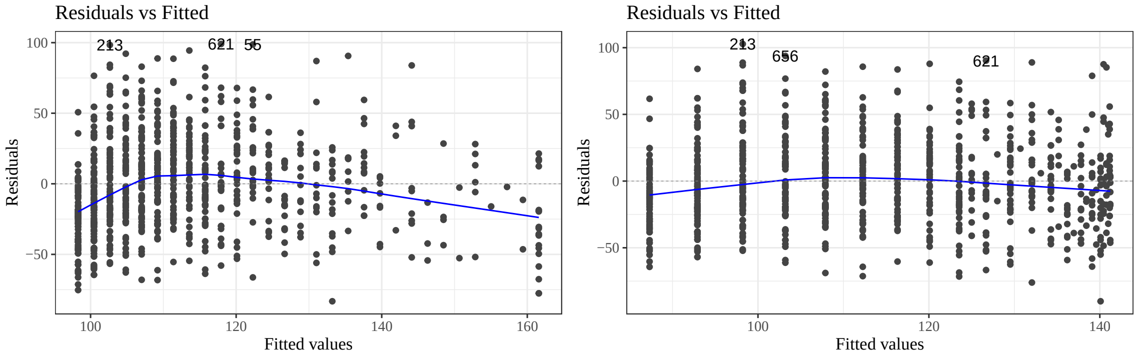
(a) Data and fit

(b) Residuals vs fitted

Figure 24: Number of authors versus title length in *PLOS Medicine*, with quadratic model fit

Linear versus quadratic fits

```r
library(ggfortify)
autoplot(lm_PLOS_linear, which = 1, ncol = 1)

autoplot(lm_PLOS_quad, which = 1, ncol = 1)
```



(a) Linear

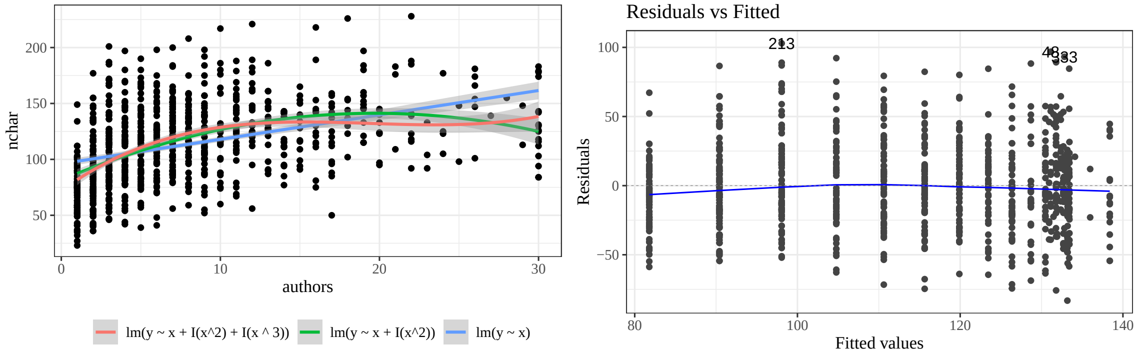
(b) Quadratic

Figure 25: Residuals versus fitted plot for linear and quadratic fits to `PLOS` data

Cubic fit

```
lm_PLOS_cub = lm(
  formula = nchar ~ authors + I(authors^2) + I(authors^3),
  data = PLOS)
```

```
fig4 =
  fig3 +
geom_smooth(
    method = "lm",
    fullrange = TRUE,
    formula = y ~ x + I(x ^ 2) + I(x ^ 3),
    aes(col = "lm(y ~ x + I(x^2) + I(x ^ 3))")
  )
fig4

autoplot(lm_PLOS_cub, which = 1, ncol = 1)
```



(a) Data and fit

(b) Residuals vs fitted

Figure 26: Number of authors versus title length in *PLOS Medicine*, with cubic model fit

---

Logarithmic fit

```
lm_PLOS_log = lm(nchar ~ log(authors), data = PLOS)
```

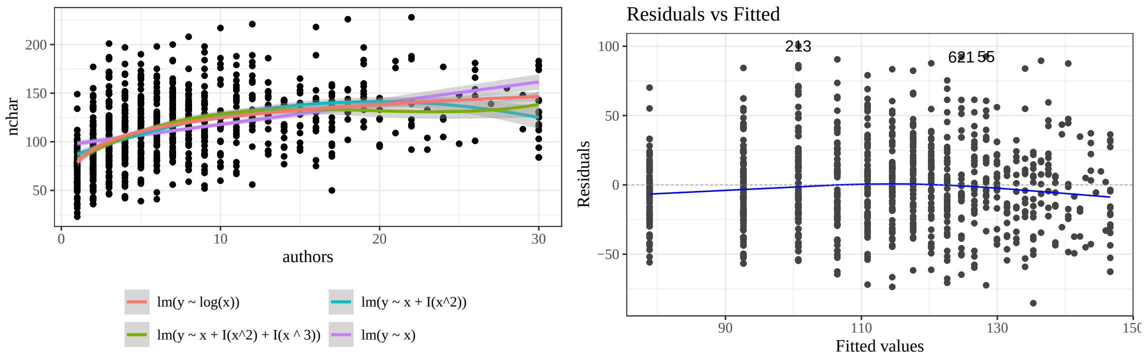
```
fig5 = fig4 +
  geom_smooth(
    method = "lm",
    fullrange = TRUE,
    formula = y ~ log(x),
    aes(col = "lm(y ~ log(x))")
  )
fig5

autoplot(lm_PLOS_log, which = 1, ncol = 1)
```

Table 30: linear vs quadratic

```
anova(lm_PLOS_linear, lm_PLOS_quad)
#> # A tibble: 2 x 6
#>   Res.Df     RSS    Df `Sum of Sq`     F  `Pr(>F)`
#>    <dbl>   <dbl> <dbl>       <dbl> <dbl>     <dbl>
#> 1    876 947502.    NA         NA    NA   NA
#> 2    875 880950.     1      66552.  66.1  1.46e-15
```

Table 31: quadratic vs cubic

```
anova(lm_PLOS_quad, lm_PLOS_cub)
#> # A tibble: 2 x 6
#>   Res.Df     RSS    Df `Sum of Sq`     F  `Pr(>F)`
#>    <dbl>   <dbl> <dbl>       <dbl> <dbl>     <dbl>
#> 1    875 880950.    NA         NA    NA   NA
#> 2    874 865933.     1      15018.  15.2  0.000106
```



(a) Data and fit

(b) Residuals vs fitted

Figure 27: logarithmic fit

Model selection

AIC/BIC

```
AIC(lm_PLOS_quad)
#> [1] 8567.61
AIC(lm_PLOS_cub)
#> [1] 8554.51
```

```
AIC(lm_PLOS_cub)
#> [1] 8554.51
AIC(lm_PLOS_log)
#> [1] 8543.63
```

```
BIC(lm_PLOS_cub)
#> [1] 8578.4
BIC(lm_PLOS_log)
#> [1] 8557.97
```

Extrapolation is dangerous

```
fig_all = fig5 +
  xlim(0, 60)
fig_all
```

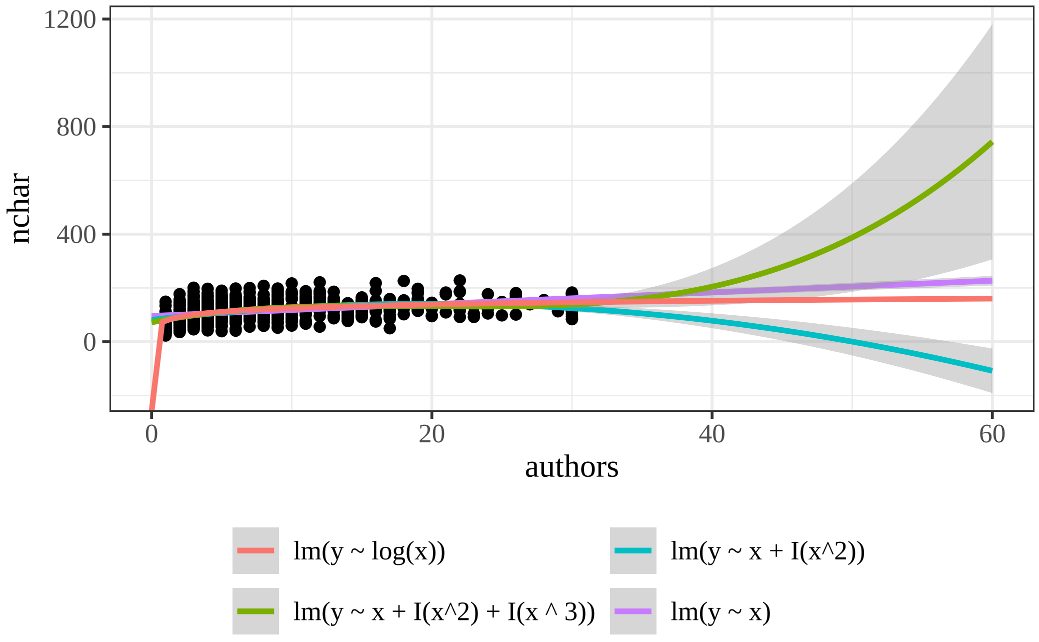


Figure 28: Number of authors versus title length in *PLOS Medicine*

Scale-location plot

We can also plot the square roots of the absolute values of the standardized residuals against the fitted values (Figure 29).

```
autoplot(bw_lm2, which = 3, ncol = 1) |> print()
```

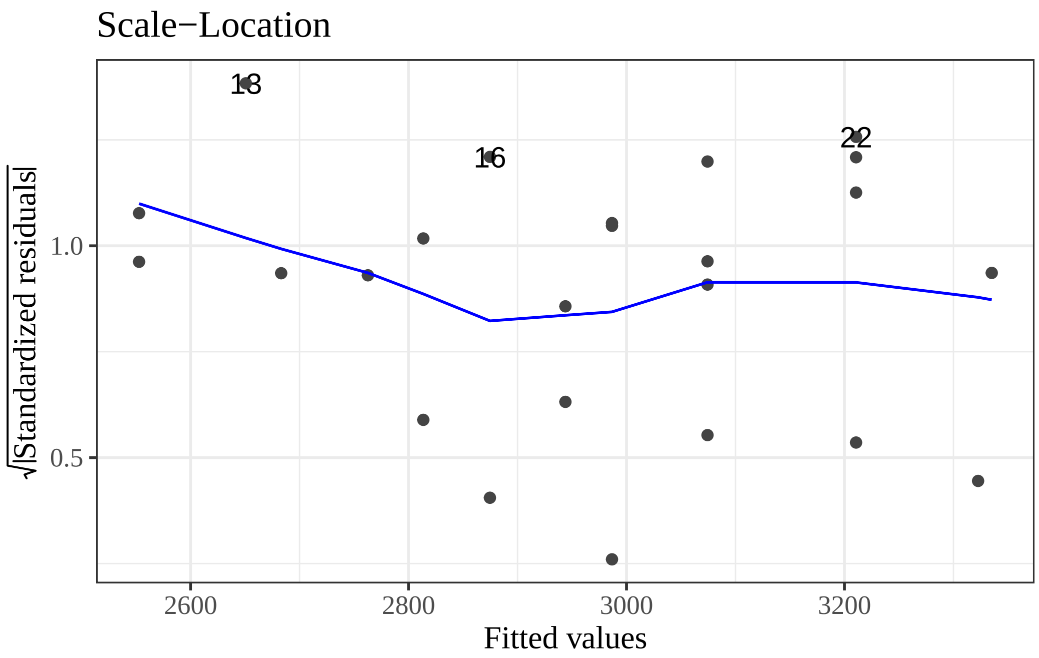


Figure 29: Scale-location plot of `birthweight` data

Here, the blue line doesn't need to be near 0, but it should be flat. If not, the residual variance σ^2 might not be constant, and we might need to transform our outcome Y (or use a model that allows non-constant variance).

Residuals versus leverage

We can also plot our standardized residuals against "leverage", which roughly speaking is a measure of how unusual each x_i value is. Very unusual x_i values can have extreme effects on the model fit, so we might want to remove those observations as outliers, particularly if they have large residuals.

```
autoplot(bw_lm2, which = 5, ncol = 1) |> print()
```

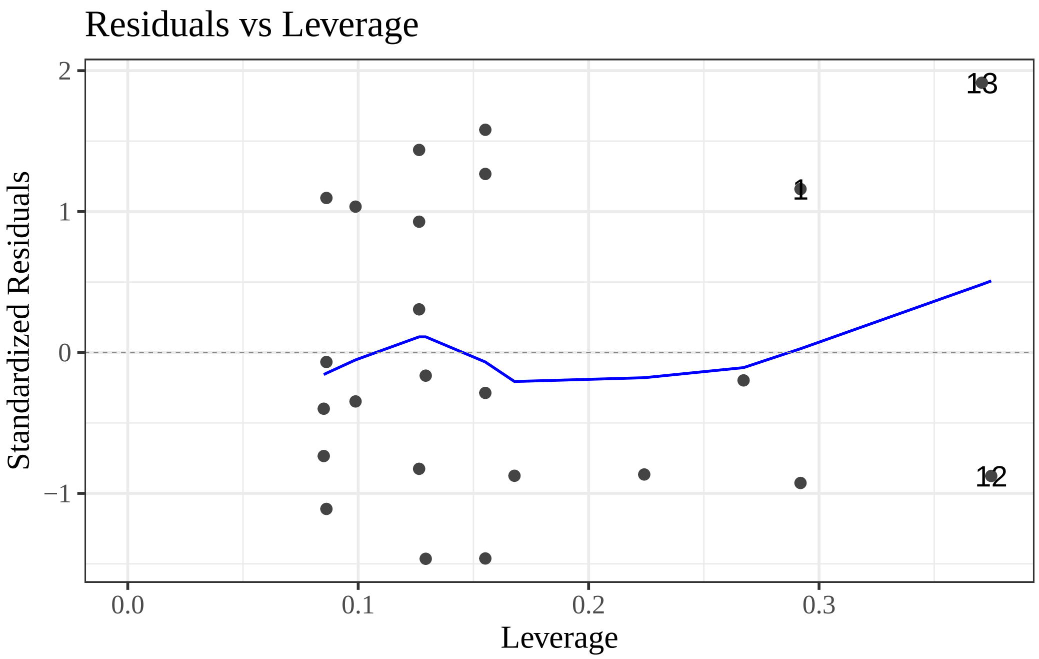


Figure 30: `birthweight` model with interactions (Equation 2): residuals versus leverage

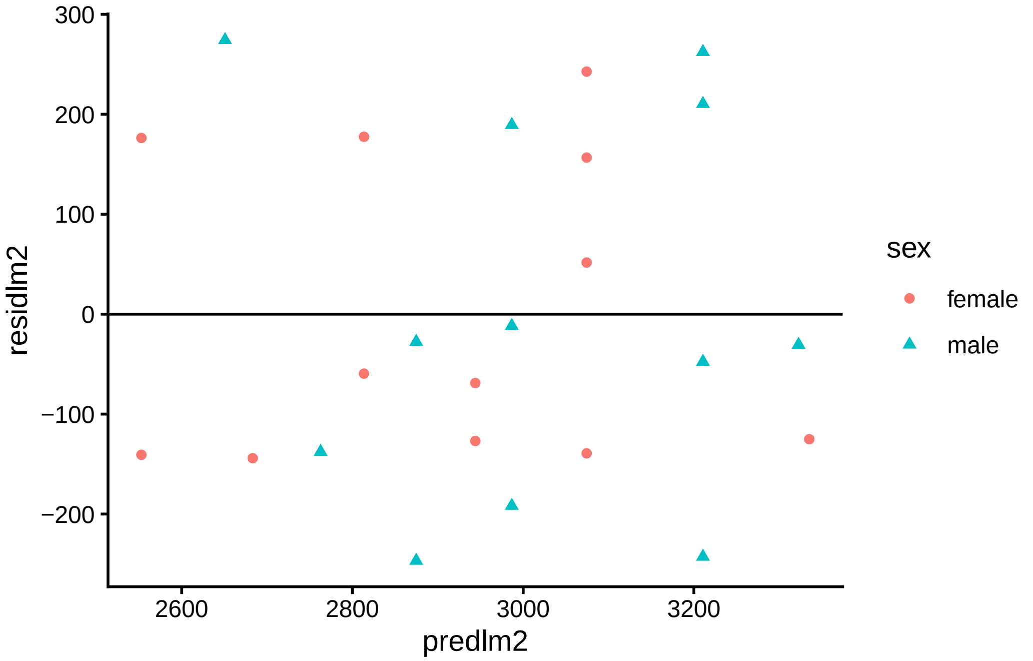
The blue line should be relatively flat and close to 0 here.

1.8.7 Diagnostics constructed by hand

```
bw <-
  bw |>
  mutate(
    predlm2 = predict(bw_lm2),
    residlm2 = weight - predlm2,
    std_resid = residlm2 / sigma(bw_lm2),
    # std_resid_builtin = rstandard(bw_lm2), # uses leverage
    sqrt_abs_std_resid = std_resid |> abs() |> sqrt()
  )
```

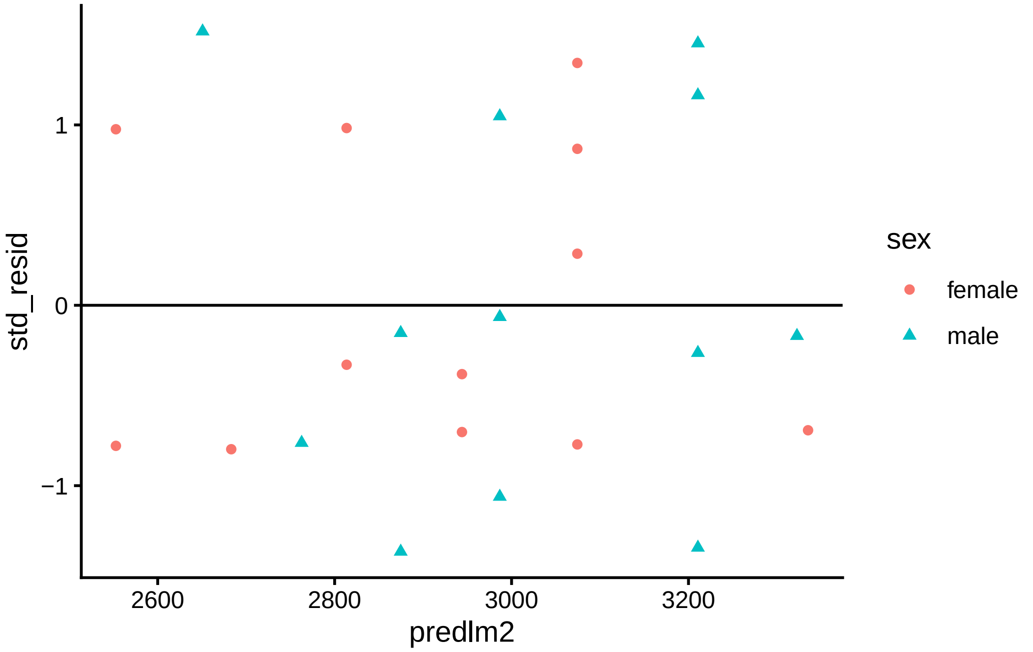
Residuals vs fitted

```
resid_vs_fit <- bw |>
  ggplot(
    aes(x = predlm2, y = residlm2, col = sex, shape = sex)
  ) +
  geom_point() +
  theme_classic() +
  geom_hline(yintercept = 0)

print(resid_vs_fit)
```



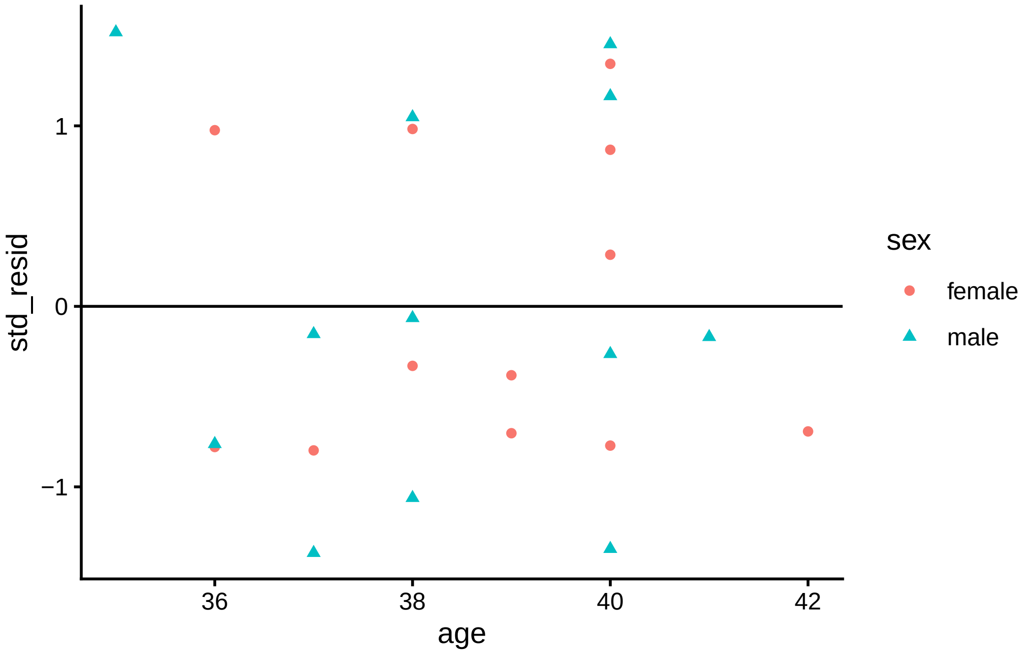
Standardized residuals vs fitted

```
bw |>
  ggplot(
    aes(x = predlm2, y = std_resid, col = sex, shape = sex)
  ) +
  geom_point() +
  theme_classic() +
  geom_hline(yintercept = 0)
```



Standardized residuals vs gestational age

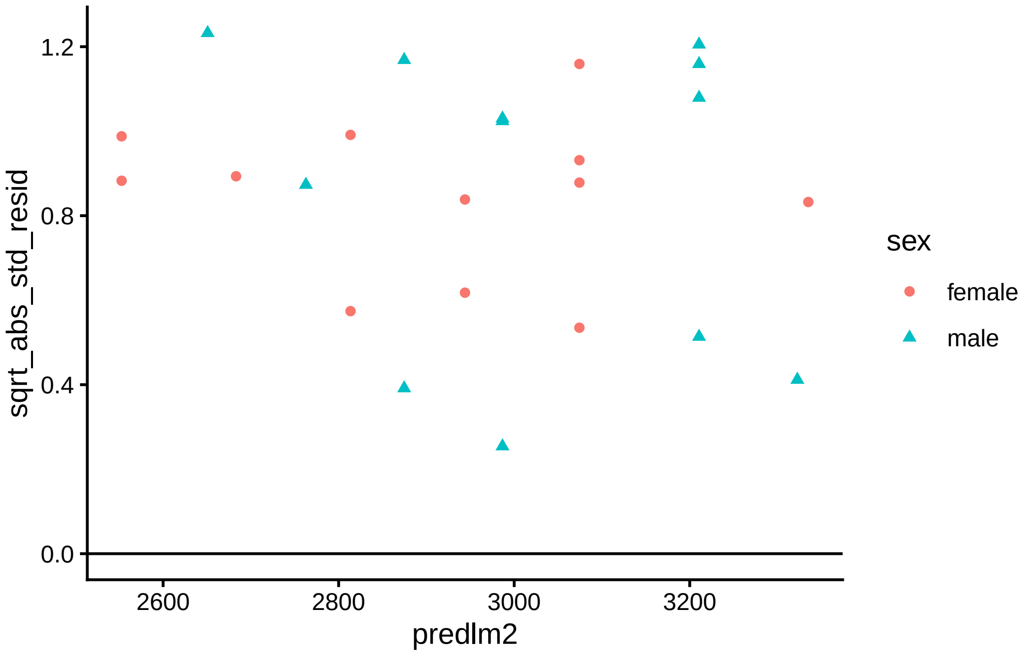
```
bw |>
  ggplot(
    aes(x = age, y = std_resid, col = sex, shape = sex)
  ) +
  geom_point() +
  theme_classic() +
  geom_hline(yintercept = 0)
```



`sqrt(abs(rstandard()))` vs fitted

Compare with `autoplot(bw_lm2, 3)`

```
bw |>
  ggplot(
    aes(x = predlm2, y = sqrt_abs_std_resid, col = sex, shape = sex)
  ) +
  geom_point() +
  theme_classic() +
  geom_hline(yintercept = 0)
```



1.9 Model selection

(adapted from Dobson and Barnett (2018) §6.3.3; for more information on prediction, see James et al. (2013) and Harrell (2015)).

If we have a lot of covariates in our dataset, we might want to choose a small subset to use in our model.

There are a few possible metrics to consider for choosing a "best" model.

1.9.1 Mean squared error

We might want to minimize the **mean squared error**, $\text{E}[(y-\hat{y})^2]$, for new observations that weren't in our data set when we fit the model.

Unfortunately,

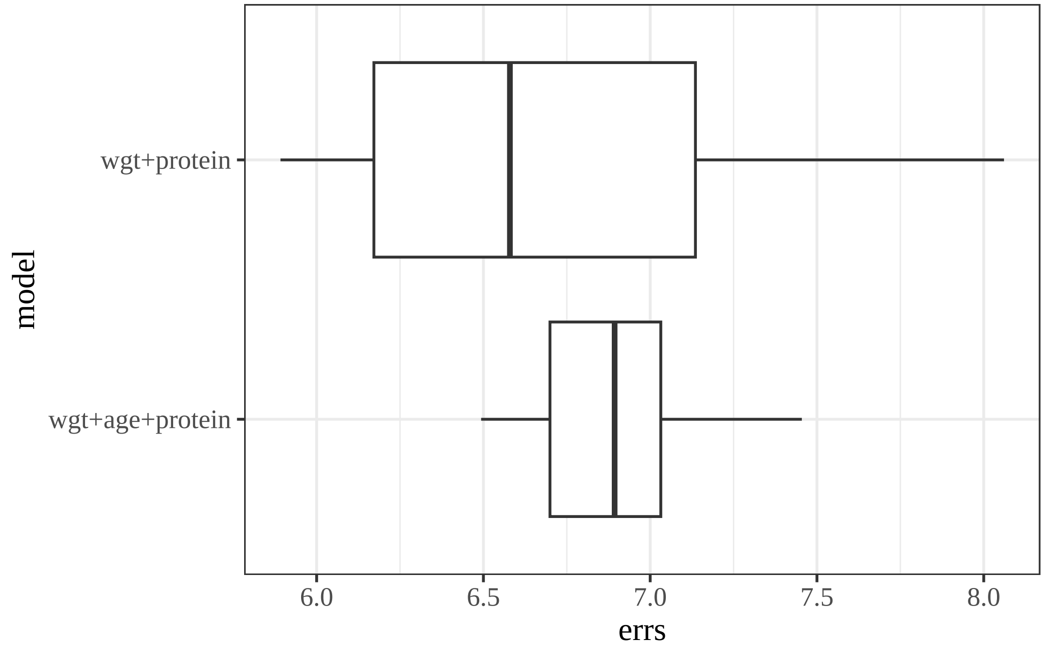
$$\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2$$

gives a biased estimate of $\text{E}[(y-\hat{y})^2]$ for new data. If we want an unbiased estimate, we will have to be clever.

Cross-validation

```
data("carbohydrate", package = "dobson")
library(cvTools)
full_model <- lm(carbohydrate ~ ., data = carbohydrate)
cv_full <-
  full_model |> cvFit(
    data = carbohydrate, K = 5, R = 10,
    y = carbohydrate$carbohydrate
  )

reduced_model <- full_model |> update(formula = ~ . - age)

cv_reduced <-
  reduced_model |> cvFit(
    data = carbohydrate, K = 5, R = 10,
    y = carbohydrate$carbohydrate
  )
```

```
results_reduced <-
  tibble(
    model = "wgt+protein",
    errs = cv_reduced$reps[]
  )
results_full <-
  tibble(
    model = "wgt+age+protein",
    errs = cv_full$reps[]
  )

cv_results <-
  bind_rows(results_reduced, results_full)

cv_results |>
  ggplot(aes(y = model, x = errs)) +
  geom_boxplot()
```



---

comparing metrics

```
compare_results <- tribble(
  ~model, ~cvRMSE, ~r.squared, ~adj.r.squared, ~trainRMSE, ~loglik,
  "full",
  cv_full$cv,
  summary(full_model)$r.squared,
  summary(full_model)$adj.r.squared,
  sigma(full_model),
  logLik(full_model) |> as.numeric(),
  "reduced",
```

```r
  cv_reduced$cv,
  summary(reduced_model)$r.squared,
  summary(reduced_model)$adj.r.squared,
  sigma(reduced_model),
  logLik(reduced_model) |> as.numeric()
)

compare_results
#> # A tibble: 2 x 6
#>   model   cvRMSE r.squared adj.r.squared trainRMSE loglik
#>   <chr>    <dbl>     <dbl>         <dbl>     <dbl>  <dbl>
#> 1 full      6.90     0.481         0.383      5.96  -61.8
#> 2 reduced   6.70     0.445         0.380      5.97  -62.5
```

```r
anova(full_model, reduced_model)
#> # A tibble: 2 x 6
#>   Res.Df   RSS    Df `Sum of Sq`     F `Pr(>F)`
#>    <dbl> <dbl> <dbl>       <dbl> <dbl>    <dbl>
#> 1     16  568.    NA         NA   NA       NA
#> 2     17  606.    -1        -38.4  1.08    0.314
```

---

stepwise regression

```r
library(olsrr)
olsrr:::ols_step_both_aic(full_model)
#>
#>
#>                              Stepwise Summary
#> -------------------------------------------------------------------------
#> Step    Variable        AIC        SBC        SBIC       R2        Adj. R2
#> -------------------------------------------------------------------------
#>  0      Base Model     140.773    142.764    83.068    0.00000    0.00000
#>  1      protein (+)    137.950    140.937    80.438    0.21427    0.17061
#>  2      weight (+)     132.981    136.964    77.191    0.44544    0.38020
#> -------------------------------------------------------------------------
#>
#> Final Model Output
#> ------------------
#>
#>                          Model Summary
#> ---------------------------------------------------------------
#> R                       0.667       RMSE                 5.505
#> R-Squared               0.445       MSE                 30.301
#> Adj. R-Squared          0.380       Coef. Var           15.879
#> Pred R-Squared          0.236       AIC                132.981
#> MAE                     4.593       SBC                136.964
#> ---------------------------------------------------------------
#>  RMSE: Root Mean Square Error
#>  MSE: Mean Square Error
#>  MAE: Mean Absolute Error
#>  AIC: Akaike Information Criteria
#>  SBC: Schwarz Bayesian Criteria
#>
#>                                ANOVA
#> -------------------------------------------------------------------
#>                 Sum of
#>                Squares        DF    Mean Square      F        Sig.
#> -------------------------------------------------------------------
#> Regression     486.778         2        243.389    6.827    0.0067
#> Residual       606.022        17         35.648
#> Total         1092.800        19
#> -------------------------------------------------------------------
#>
#>                                    Parameter Estimates
#> ----------------------------------------------------------------------------------------
#>       model      Beta    Std. Error    Std. Beta      t        Sig      lower     upper
#> ----------------------------------------------------------------------------------------
#> (Intercept)    33.130        12.572                  2.635    0.017     6.607    59.654
#>     protein     1.824         0.623        0.534     2.927    0.009     0.509     3.139
#>      weight    -0.222         0.083       -0.486    -2.662    0.016    -0.397    -0.046
#> ----------------------------------------------------------------------------------------
```

Lasso

$$\arg\max_{\theta} \left\{ \ell(\theta) - \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

```r
library(glmnet)
y <- carbohydrate$carbohydrate
x <- carbohydrate |>
  select(age, weight, protein) |>
  as.matrix()
fit <- glmnet(x, y)
```

```r
autoplot(fit, xvar = "lambda")
```

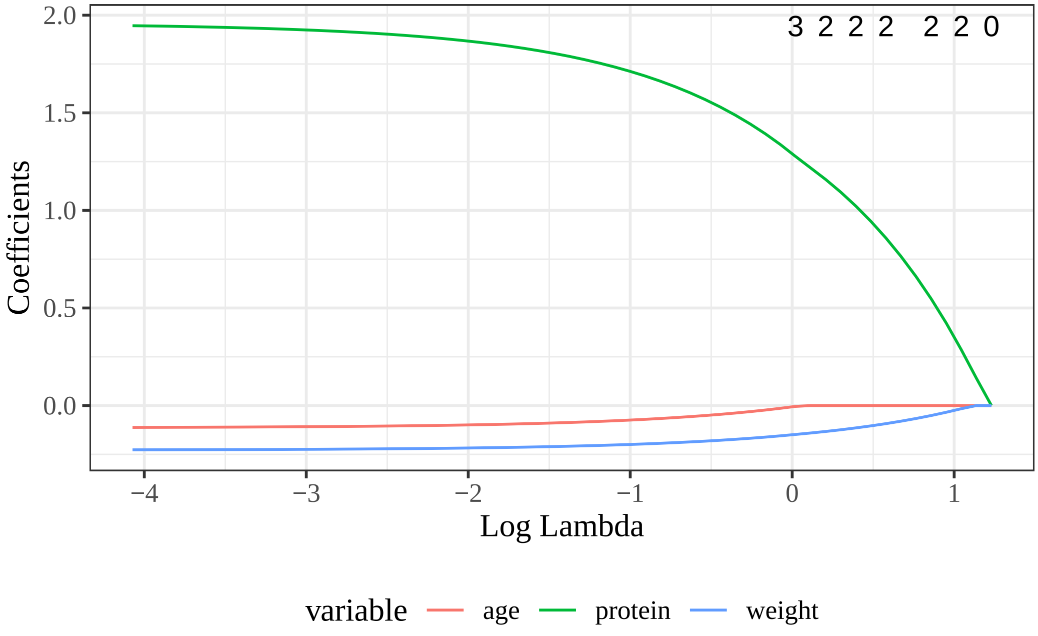
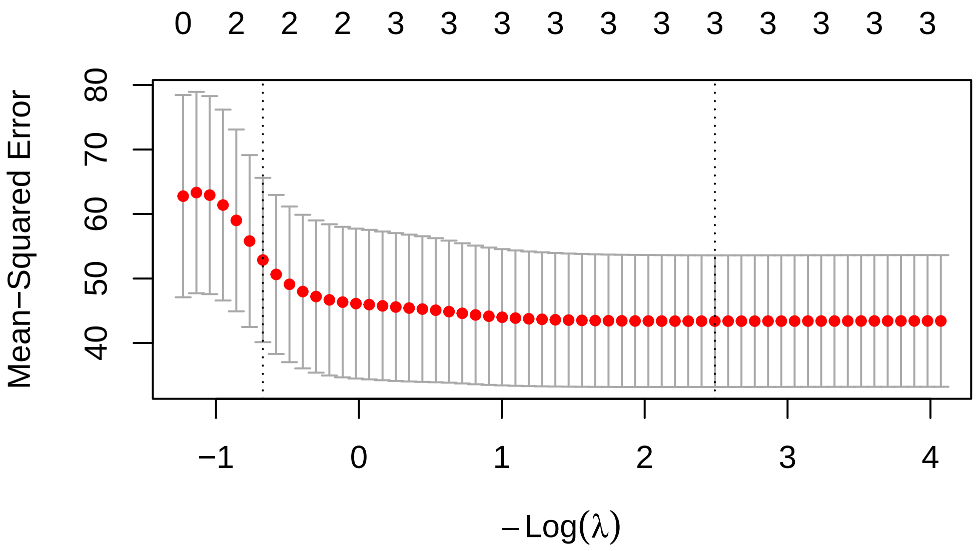


Figure 31: Lasso selection

```r
cvfit <- cv.glmnet(x, y)
plot(cvfit)
```



```r
coef(cvfit, s = "lambda.1se")
#> 4 x 1 sparse Matrix of class "dgCMatrix"
#>              lambda.1se
#> (Intercept) 34.3091615
#> age          .
#> weight      -0.0800142
#> protein      0.7640510
```

1.10 Categorical covariates with more than two levels

1.10.1 Example: `birthweight`

In the birthweight example, the variable `sex` had only two observed values:

```r
unique(bw$sex)
#> [1] female male
#> Levels: female male
```

If there are more than two observed values, we can't just use a single variable with 0s and 1s.

1.10.2

For example, Table 32 shows the (in)famous[12] `iris` data (Anderson (1935)), and Table 33 provides summary statistics. The data include three species: "setosa", "versicolor", and "virginica".

```r
library(table1)
table1(
  x = ~ . | Species,
  data = iris,
  overall = FALSE
)
```

[12] https://www.meganstodel.com/posts/no-to-iris/

Table 32: The `iris` data

```
head(iris)
#> # A tibble: 6 x 5
#>   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
#>          <dbl>       <dbl>        <dbl>       <dbl> <fct>
#> 1          5.1         3.5          1.4         0.2 setosa
#> 2          4.9         3            1.4         0.2 setosa
#> 3          4.7         3.2          1.3         0.2 setosa
#> 4          4.6         3.1          1.5         0.2 setosa
#> 5          5           3.6          1.4         0.2 setosa
#> 6          5.4         3.9          1.7         0.4 setosa
```

Table 33: Summary statistics for the `iris` data

	setosa	**versicolor**	**virginica**
	(N=50)	**(N=50)**	**(N=50)**
Sepal.Length			
Mean (SD)	5.01 (0.352)	5.94 (0.516)	6.59 (0.636)
Median [Min, Max]	5.00 [4.30, 5.80]	5.90 [4.90, 7.00]	6.50 [4.90, 7.90]
Sepal.Width			
Mean (SD)	3.43 (0.379)	2.77 (0.314)	2.97 (0.322)
Median [Min, Max]	3.40 [2.30, 4.40]	2.80 [2.00, 3.40]	3.00 [2.20, 3.80]
Petal.Length			
Mean (SD)	1.46 (0.174)	4.26 (0.470)	5.55 (0.552)
Median [Min, Max]	1.50 [1.00, 1.90]	4.35 [3.00, 5.10]	5.55 [4.50, 6.90]
Petal.Width			
Mean (SD)	0.246 (0.105)	1.33 (0.198)	2.03 (0.275)
Median [Min, Max]	0.200 [0.100, 0.600]	1.30 [1.00, 1.80]	2.00 [1.40, 2.50]

Table 34: `iris` data with numeric coding of species

```r
data(iris) # this step is not always necessary, but ensures you're starting
# from the original version of a dataset stored in a loaded package

iris <-
  iris |>
  tibble() |>
  mutate(
    X = case_when(
      Species == "setosa" ~ 1,
      Species == "virginica" ~ 2,
      Species == "versicolor" ~ 3
    )
  )

iris |>
  distinct(Species, X)
#> # A tibble: 3 x 2
#>   Species        X
#>   <fct>      <dbl>
#> 1 setosa         1
#> 2 versicolor     3
#> 3 virginica      2
```

If we want to model `Sepal.Length` by species, we could create a variable X that represents "setosa" as $X = 1$, "virginica" as $X = 2$, and "versicolor" as $X = 3$.

Then we could fit a model like:

```r
iris_lm1 <- lm(Sepal.Length ~ X, data = iris)
iris_lm1 |>
  parameters() |>
  print_md()
```

Table 35: Model of `iris` data with numeric coding of `Species`

Parameter	Coefficient	SE	95% CI	t(148)	p
(Intercept)	4.91	0.16	(4.60, 5.23)	30.83	$< .001$
X	0.46	0.07	(0.32, 0.61)	6.30	$< .001$

1.10.3 Let's see how that model looks:

```r
iris_plot1 <- iris |>
  ggplot(
    aes(
      x = X,
      y = Sepal.Length
    )
  ) +
  geom_point(alpha = .1) +
  geom_abline(
    intercept = coef(iris_lm1)[1],
    slope = coef(iris_lm1)[2]
  ) +
  theme_bw(base_size = 18)
print(iris_plot1)
```

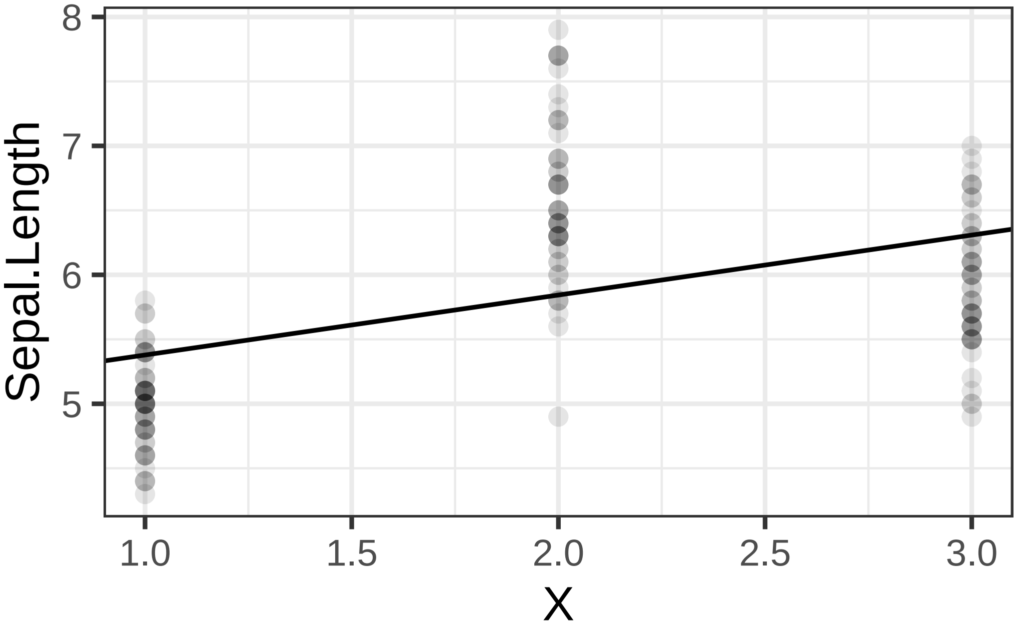


Figure 32: Model of `iris` data with numeric coding of `Species`

We have forced the model to use a straight line for the three estimated means. Maybe not a good idea?

1.10.4 Let's see what R does with categorical variables by default:

```
iris_lm2 <- lm(Sepal.Length ~ Species, data = iris)
iris_lm2 |>
  parameters() |>
  print_md()
```

Table 36: Model of `iris` data with `Species` as a categorical variable

Parameter	Coefficient	SE	95% CI	t(147)	p
(Intercept)	5.01	0.07	(4.86, 5.15)	68.76	< .001
Species (versicolor)	0.93	0.10	(0.73, 1.13)	9.03	< .001
Species (virginica)	1.58	0.10	(1.38, 1.79)	15.37	< .001

1.10.5 Re-parametrize with no intercept

If you don't want the default and offset option, you can use "-1" like we've seen previously:

```
iris_lm2_no_int <- lm(Sepal.Length ~ Species - 1, data = iris)
iris_lm2_no_int |>
  parameters() |>
  print_md()
```

Table 37

Parameter	Coefficient	SE	95% CI	t(147)	p
Species (setosa)	5.01	0.07	(4.86, 5.15)	68.76	< .001
Species (versicolor)	5.94	0.07	(5.79, 6.08)	81.54	< .001
Species (virginica)	6.59	0.07	(6.44, 6.73)	90.49	< .001

1.10.6 Let's see what these new models look like:

```
iris_plot2 <-
  iris |>
  mutate(
    predlm2 = predict(iris_lm2)
  ) |>
  arrange(X) |>
  ggplot(aes(x = X, y = Sepal.Length)) +
  geom_point(alpha = .1) +
  geom_line(aes(y = predlm2), col = "red") +
  geom_abline(
    intercept = coef(iris_lm1)[1],
    slope = coef(iris_lm1)[2]
  ) +
  theme_bw(base_size = 18)

print(iris_plot2)
```

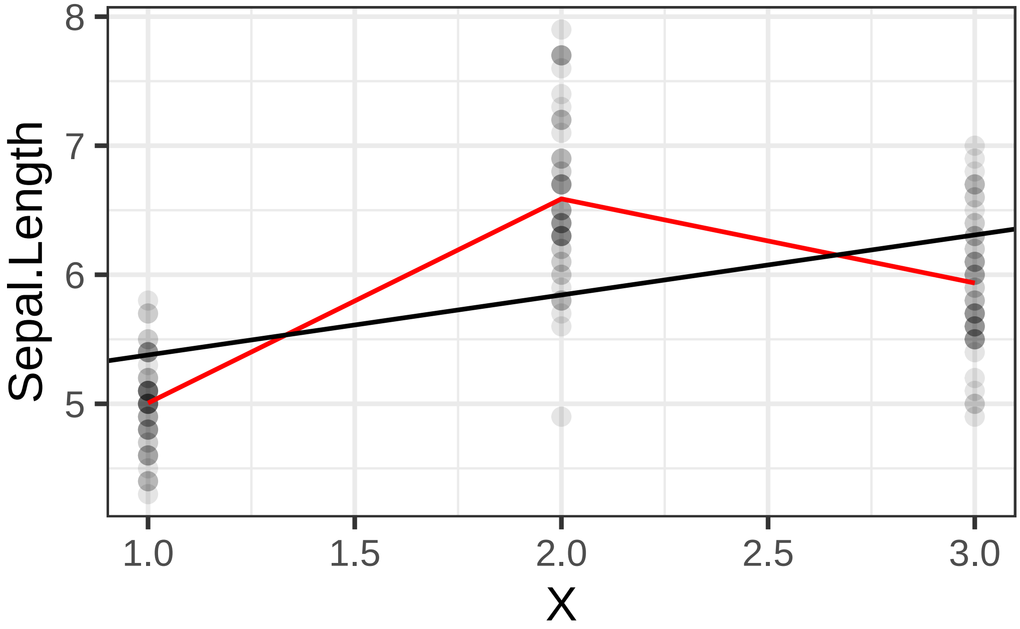


Figure 33

1.10.7 Let's see how R did that:

This format is called a "corner point parametrization" (e.g., in Dobson and Barnett (2018)) or "treatment coding" (e.g., in Dunn and Smyth (2018)).

The default contrasts are controlled by `options("contrasts")`:

Table 38

```
formula(iris_lm2)
#> Sepal.Length ~ Species
model.matrix(iris_lm2) |>
  as_tibble() |>
  unique()
#> # A tibble: 3 x 3
#>   `(Intercept)` Speciesversicolor Speciesvirginica
#>           <dbl>             <dbl>            <dbl>
#> 1             1                 0                0
#> 2             1                 1                0
#> 3             1                 0                1
```

Table 39

```
formula(iris_lm2_no_int)
#> Sepal.Length ~ Species - 1
model.matrix(iris_lm2_no_int) |>
  as_tibble() |>
  unique()
#> # A tibble: 3 x 3
#>   Speciessetosa Speciesversicolor Speciesvirginica
#>           <dbl>             <dbl>            <dbl>
#> 1             1                 0                0
#> 2             0                 1                0
#> 3             0                 0                1
```

```
options("contrasts")
#> $contrasts
#>         unordered           ordered
#> "contr.treatment"      "contr.poly"
```

See `?options` for more details.

This format is called a "group point parametrization" (e.g., in Dobson and Barnett (2018)).

There are more options; see Dobson and Barnett (2018) §6.4.1 and the `codingMatrices` package[13] vignette[14] (Venables (2023)).

1.11 Ordinal covariates

(c.f. Dobson and Barnett (2018) §2.4.4)

We can create ordinal variables in R using the `ordered()` function[15].

Example 1.4.

```
url <- paste0(
  "https://regression.ucsf.edu/sites/g/files/tkssra6706/",
  "f/wysiwyg/home/data/hersdata.dta"
)
```

[13] https://CRAN.R-project.org/package=codingMatrices

[14] https://cran.r-project.org/web/packages/codingMatrices/vignettes/codingMatrices.pdf

[15] or equivalently, `factor(ordered = TRUE)`

Table 40: HERS dataset

```
hers |> head()
#> # A tibble: 6 x 37
#>   HT               age raceth  nonwhite smoking drinkany exercise physact globrat
#>   <dbl+lbl>      <dbl> <dbl+l> <dbl+lb> <dbl+l> <dbl+lb> <dbl+lb> <dbl+l> <dbl+l>
#> 1 0 [placebo]       70 2 [Afr~ 1 [yes]  0 [no]  0 [no]   0 [no]   5 [muc~ 3 [goo~
#> 2 0 [placebo]       62 2 [Afr~ 1 [yes]  0 [no]  0 [no]   0 [no]   1 [muc~ 3 [goo~
#> 3 1 [hormone t~     69 1 [Whi~ 0 [no]   0 [no]  0 [no]   0 [no]   3 [abo~ 3 [goo~
#> 4 0 [placebo]       64 1 [Whi~ 0 [no]   1 [yes] 1 [yes]  0 [no]   1 [muc~ 3 [goo~
#> 5 0 [placebo]       65 1 [Whi~ 0 [no]   0 [no]  0 [no]   0 [no]   2 [som~ 3 [goo~
#> 6 1 [hormone t~     68 2 [Afr~ 1 [yes]  0 [no]  1 [yes]  0 [no]   3 [abo~ 3 [goo~
#> # i 28 more variables: poorfair <dbl+lbl>, medcond <dbl>, htnmeds <dbl+lbl>,
#> #   statins <dbl+lbl>, diabetes <dbl+lbl>, dmpills <dbl+lbl>,
#> #   insulin <dbl+lbl>, weight <dbl>, BMI <dbl>, waist <dbl>, WHR <dbl>,
#> #   glucose <dbl>, weight1 <dbl>, BMI1 <dbl>, waist1 <dbl>, WHR1 <dbl>,
#> #   glucose1 <dbl>, tchol <dbl>, LDL <dbl>, HDL <dbl>, TG <dbl>, tchol1 <dbl>,
#> #   LDL1 <dbl>, HDL1 <dbl>, TG1 <dbl>, SBP <dbl>, DBP <dbl>, age10 <dbl>
```

```
library(haven)
hers <- read_dta(url)
```

Check out `?codingMatrices::contr.diff`

Anderson, Edgar. 1935. "The Irises of the Gaspe Peninsula." *Bulletin of American Iris Society* 59: 2–5.

Chatterjee, Samprit, and Ali S Hadi. 2015. *Regression Analysis by Example*. John Wiley & Sons. https://www.wiley.com/en-us/Regression+Analysis+by+Example%2C+4th+Edition-p-9780470055458.

Dobson, Annette J, and Adrian G Barnett. 2018. *An Introduction to Generalized Linear Models*. 4th ed. CRC press. https://doi.org/10.1201/9781315182780.

Dunn, Peter K, and Gordon K Smyth. 2018. *Generalized Linear Models with Examples in R*. Vol. 53. Springer. https://link.springer.com/book/10.1007/978-1-4419-0118-7.

Faraway, Julian J. 2025. *Linear Models with R*. https://www.routledge.com/Linear-Models-with-R/Faraway/p/book/9781032583983.

Harrell, Frank E. 2015. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. 2nd ed. Springer. https://doi.org/10.1007/978-3-319-19425-7.

Hogg, Robert V., Elliot A. Tanis, and Dale L. Zimmerman. 2015. *Probability and Statistical Inference*. Ninth edition. Boston: Pearson.

Hulley, Stephen, Deborah Grady, Trudy Bush, Curt Furberg, David Herrington, Betty Riggs, Eric Vittinghoff, for the Heart, and Estrogen/progestin Replacement Study (HERS) Research Group. 1998. "Randomized Trial of Estrogen Plus Progestin for Secondary Prevention of Coronary Heart Disease in Postmenopausal Women." *JAMA : The Journal of the American Medical Association* 280 (7): 605–13.

James, Gareth, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. 2013. *An Introduction to Statistical Learning*. Vol. 112. Springer. https://www.statlearning.com/.

Kleinbaum, David G, and Mitchel Klein. 2010. *Logistic Regression: A Self-Learning Text*. 3rd ed. Springer. https://link.springer.com/book/10.1007/978-1-4419-1742-3.

———. 2012. *Survival Analysis: A Self-Learning Text*. 3rd ed. Springer. https://link.springer.com/book/10.1007/978-1-4419-6646-9.

Kleinbaum, David G, Lawrence L Kupper, Azhar Nizam, K Muller, and ES Rosenberg. 2014. *Applied Regression Analysis and Other Multivariable Methods*. 5th ed. Cengage Learning. https://www.cengage.com/c/applied-regression-analysis-and-other-multivariable-methods-5e-kleinbaum/9781285051086/.

Kutner, Michael H, Christopher J Nachtsheim, John Neter, and William Li. 2005. *Applied Linear Statistical Models*. McGraw-Hill.

Polin, Richard A, William W Fox, and Steven H Abman. 2011. *Fetal and Neonatal Physiology*. 4th ed. Elsevier health sciences.

Seber, George AF, and Alan J Lee. 2012. *Linear Regression Analysis*. 2nd ed. John Wiley & Sons. https://www.wiley.com/en-us/Linear+Regression+Analysis%2C+2nd+Edition-p-9781118274422.

Venables, Bill. 2023. *codingMatrices: Alternative Factor Coding Matrices for Linear Model Formulae* (version 0.4.0). https://CRAN.R-project.org/package=codingMatrices.

Vittinghoff, Eric, David V Glidden, Stephen C Shiboski, and Charles E McCulloch. 2012. *Regression Methods in Biostatistics: Linear, Logistic, Survival, and Repeated Measures Models*. 2nd ed. Springer. https://doi.org/10.1007/978-1-4614-1353-0.

Weisberg, Sanford. 2005. *Applied Linear Regression*. Vol. 528. John Wiley & Sons.