Regression Models for Epidemiology

Ezra Morrison

Last modified: 2023-08-21: 11:35:33 (AM)

Contents

Preface

This web-book is derived from my lecture slides for the Spring 2023 session of Epidemiology 204: "Quantitative Epidemiology III: Statistical Models", at UC Davis.

I have drawn these materials from many sources, including but not limited to:

- https://www.taylorfrancis.com/books/mono/10.1201/9781315182780/introduction-generalized-linear-models-adrian-barnett-annette-dobson
- https://dmrocke.ucdavis.edu/Class/EPI204-Spring-2021/EPI204-Spring-2021.html
- https://link.springer.com/book/10.1007/978-1-4614-1353-0

License

0.1 Introduction

Configuring R

Functions from these packages will be used throughout this document:

```
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(plotly) # interactive graphics
library(dplyr) # manipulate data
library(tidyr) # Tools to help to create tidy data
library(haven) # import Stata files
library(pander) # format tables for markdown
library(knitr) # format R output for markdown
library(kableExtra) # more markdown formatting
library(parameters) # format model output tables for markdown
library(reactable) # interactive tables
library(dobson) # datasets from Dobson and Barnett 2018
library(conflicted) # check for conflicting function definitions
```

Here are some R settings I use in this document:

```
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
options('digits' = 4)
conflicts_prefer(dplyr::filter)
conflicts_prefer(ggplot2::autoplot)
```

0.2 Introduction to Epi 204

Welcome to Epidemiology 204: Quantitative Epidemiology III (Statistical Models).

In this course, we will start where Epi 203 left off: with linear regression models.

If you haven't taken Epi 203 or a similar introduction to mathematical statistical inference, please talk to me after class.

0.2.1 What you should know

Epi 202: probability models for different data types

- binomial
- Poisson
- Gaussian
- exponential

Epi 203: inference for one or several homogenous populations

- the maximum likelihood inference framework:
  - likelihood functions
  - log-likelihood functions
  - score functions
  - estimating equations
  - information matrices
  - point estimates
  - standard errors
  - confidence intervals
  - hypothesis tests
  - p-values
- Hypothesis tests for one, two, and >2 groups:
  - t-tests/ANOVA for Gaussian models
  - chi-square tests for binomial and Poisson models
- Some linear regression

Stat 108: linear regression models

- building models for Gaussian outcomes
  - multiple predictors
  - interactions

- regression diagnostics
- fundamentals of R programming; e.g.:
  - R for Data Science (Wickham, Cetinkaya-Rundel, Grolemund 2023)
  - Introductory Statistics with R (Dalgaard 2008)
- RMarkdown or Quarto for formatting homework
- LaTeX for writing math in RMarkdown/Quarto

0.2.2 What we will cover in this course

- Linear (Gaussian) regression models (review and more details)
- Regression models for non-Gaussian outcomes
  - binary
  - count
  - time to event
- Statistical analysis using R

0.3 Regression models

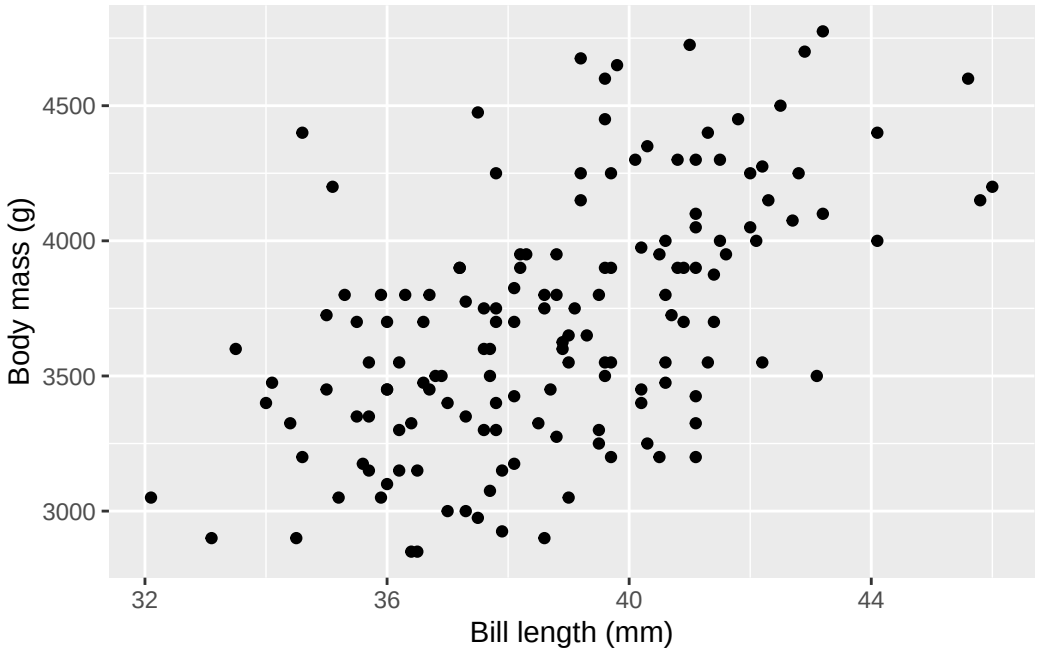
Why do we need them?

- continuous predictors
- not enough data to analyze some subgroups individually

0.3.1 Example: Adelie penguins

```
library(ggplot2)
library(plotly)
library(dplyr)
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)")
```

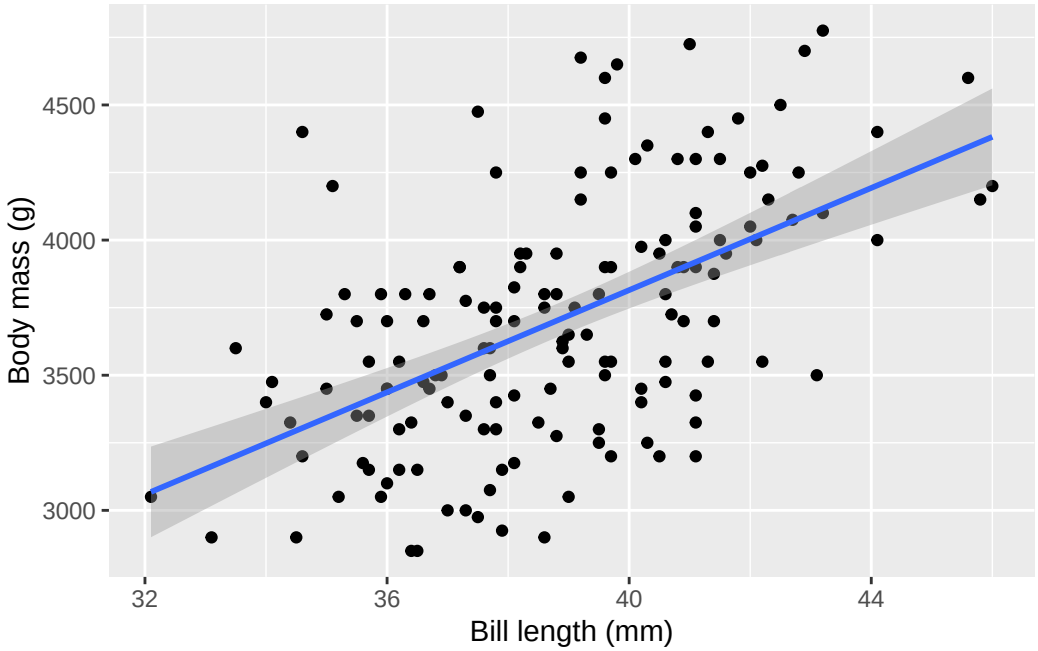
```
ggpenguins |> print()
```



0.3.2 Linear regression

```
ggpenguins2 =
  ggpenguins +
  stat_smooth(method = "lm",
              formula = y ~ x,
              geom = "smooth")
```

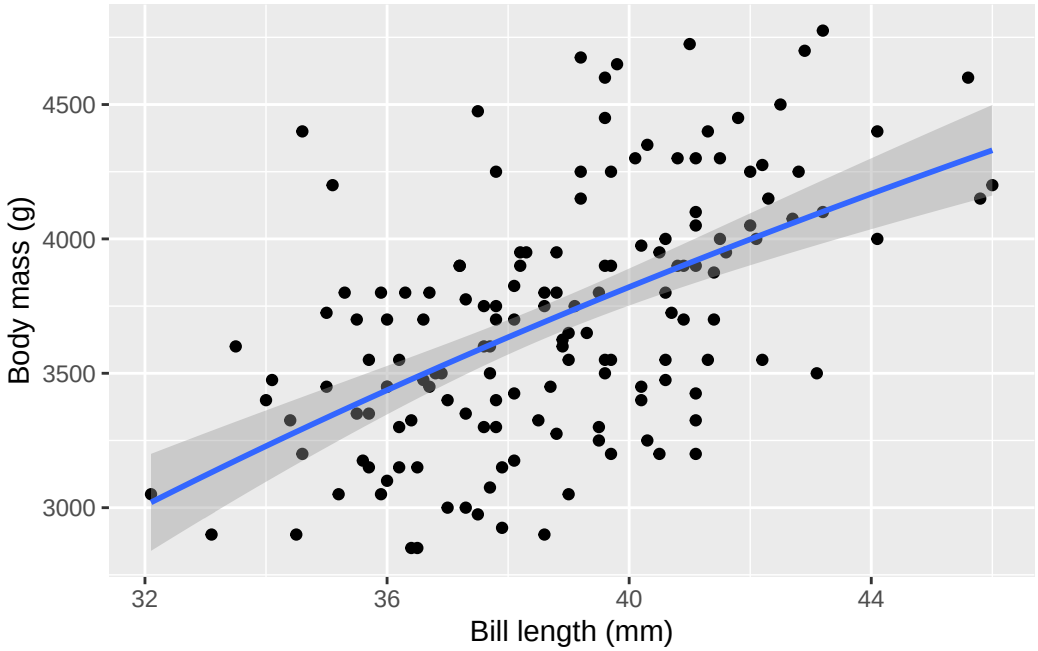
```
ggpenguins2 |> print()
```



0.3.3 Curved regression lines

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

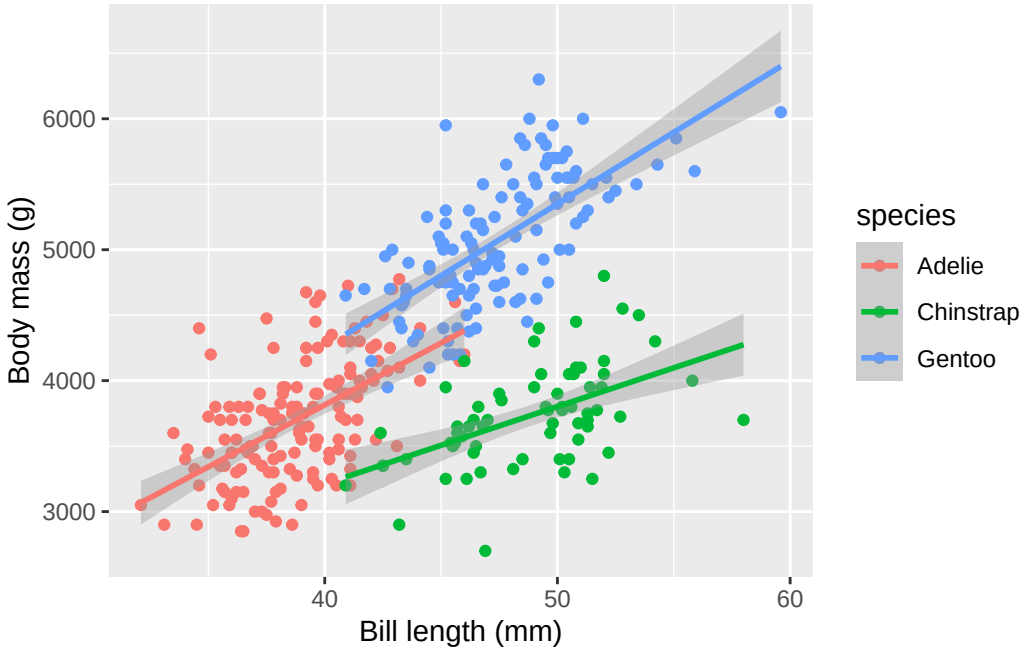
```
ggpenguins2 |> print()
```



0.3.4 Multiple regression

```
ggpenguins =
  palmerpenguins::penguins |>
  ggplot(
    aes(x = bill_length_mm ,
        y = body_mass_g,
        color = species
    )
  ) +
  geom_point() +
  stat_smooth(
    method = "lm",
    formula = y ~ x,
    geom = "smooth") +
  xlab("Bill length (mm)") +
  ylab("Body mass (g)")
```

```
ggpenguins |> print()
```



0.3.5 Modeling non-Gaussian outcomes

```
library(glmx)
data(BeetleMortality)
beetles = BeetleMortality |>
  mutate(
    pct = died/n,
    survived = n - died
  )

plot1 =
  beetles |>
  ggplot(aes(x = dose, y = pct)) +
  geom_point(aes(size = n)) +
  xlab("Dose (log mg/L)") +
  ylab("Mortality rate (%)") +
  scale_y_continuous(labels = scales::percent) +
  # xlab(bquote(log[10]), bquote(CS[2])) +
  scale_size(range = c(1,2))
```

```
print(plot1)
```

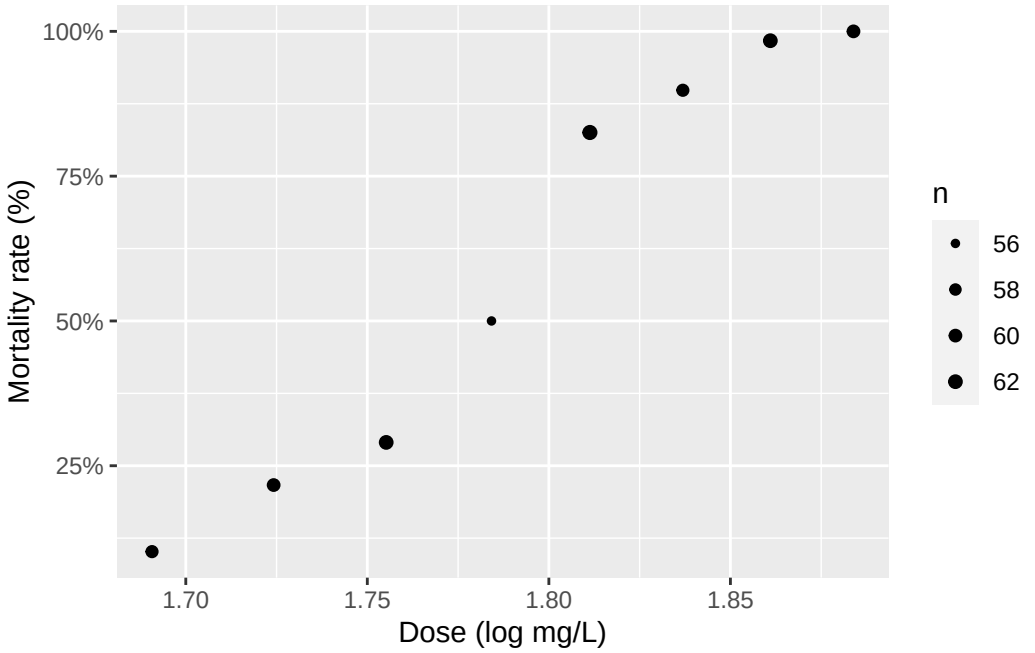


Figure 1: Mortality rates of adult flour beetles after five hours' exposure to gaseous carbon disulphide (Bliss 1935)

0.3.6 Why don't we use linear regression?

```
beetles_long =
  beetles  |>
  reframe(.by = everything(),
          outcome = c(
            rep(1, times = died),
            rep(0, times = survived))
  )

lm1 =
  beetles_long |>
  lm(
    formula = outcome ~ dose,
    data = _)
```

```r
range1 = range(beetles$dose) + c(-.2, .2)

f.linear = function(x) predict(lm1, newdata = data.frame(dose = x))

plot2 =
  plot1 +
  geom_function(fun = f.linear, aes(col = "Straight line")) +
  labs(colour="Model", size = "")
```

```r
print(plot2)
```

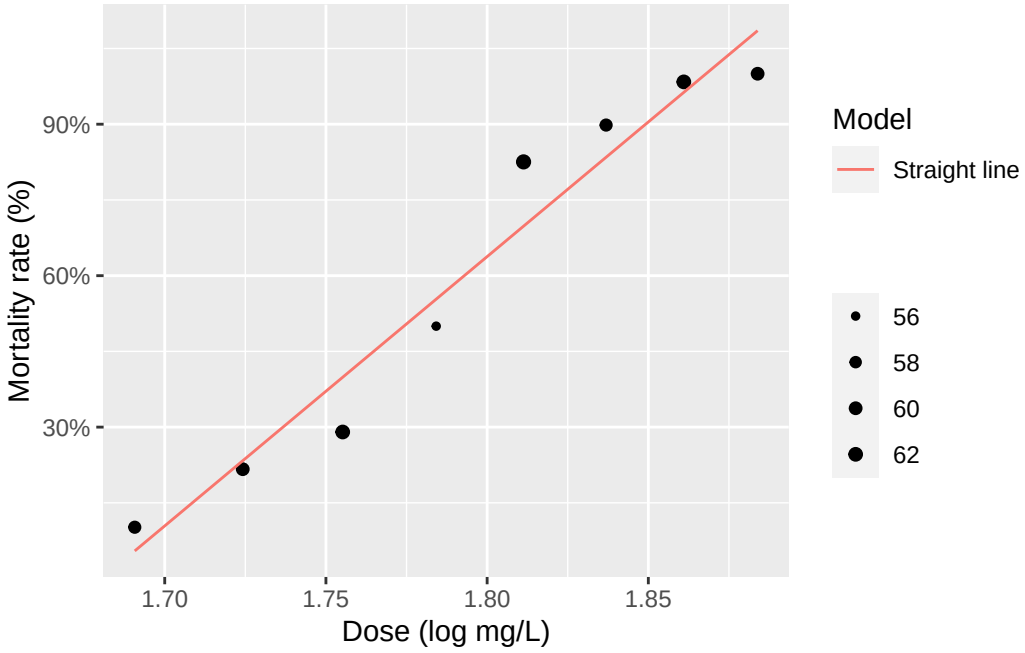


Figure 2: Mortality rates of adult flour beetles after five hours' exposure to gaseous carbon disulphide (Bliss 1935)

0.3.7 Zoom out

```r
print(plot2 + expand_limits(x = c(1.6, 2)))
```

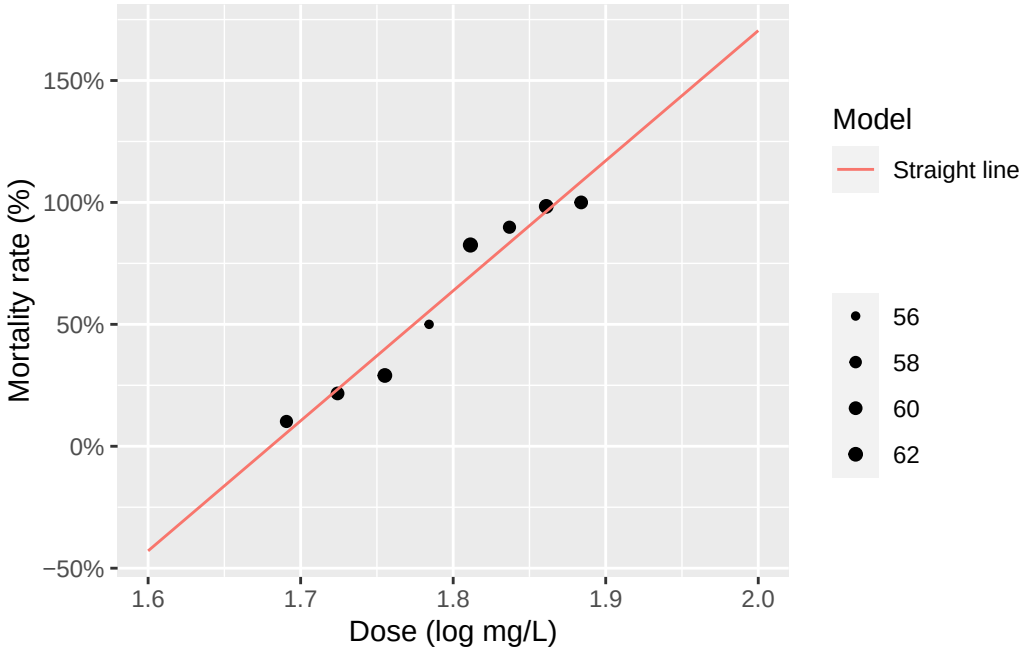


Figure 3: Mortality rates of adult flour beetles after five hours' exposure to gaseous carbon disulphide (Bliss 1935)

0.3.8 log transformation of dose?

```
lm2 =
  beetles_long |>
  lm(formula = outcome ~ log(dose), data = _)

f.linearlog = function(x) predict(lm2, newdata = data.frame(dose = x))

plot3 = plot2 +
  expand_limits(x = c(1.6, 2)) +
  geom_function(fun = f.linearlog, aes(col = "Log-transform dose"))
```

```
print(plot3 + expand_limits(x = c(1.6, 2)))
```

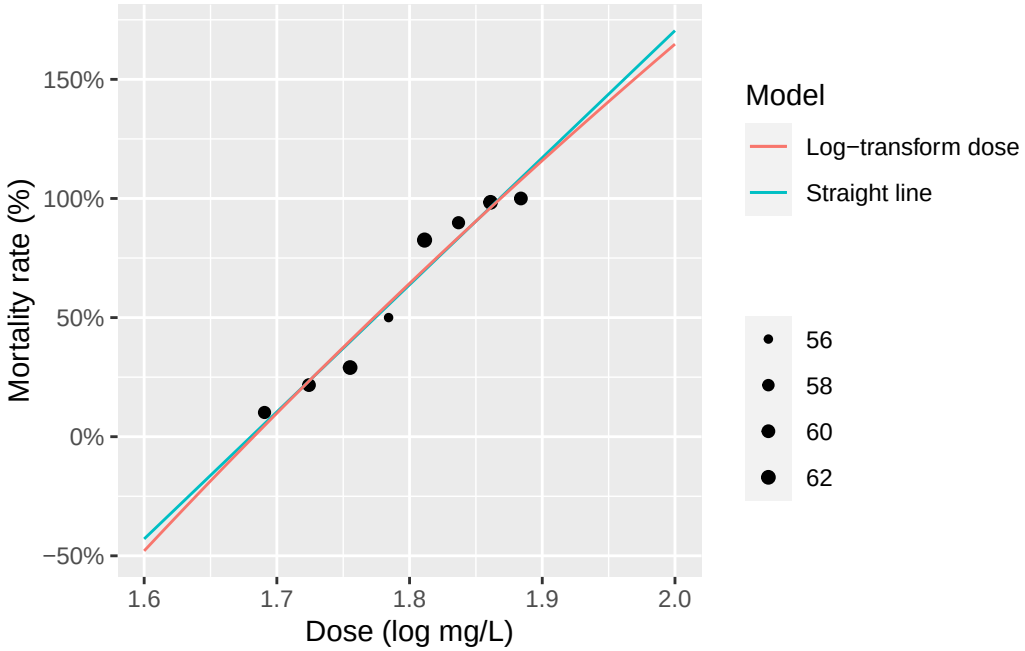


Figure 4: Mortality rates of adult flour beetles after five hours' exposure to gaseous carbon disulphide (Bliss 1935)

0.3.9 Logistic regression

```
glm1 = beetles |>
  glm(formula = cbind(died, survived) ~ dose, family = "binomial")

f = function(x) predict(glm1, newdata = data.frame(dose = x), type = "response")

plot4 = plot3 + geom_function(fun = f, aes(col = "Logistic regression"))
```

```
print(plot4)
```

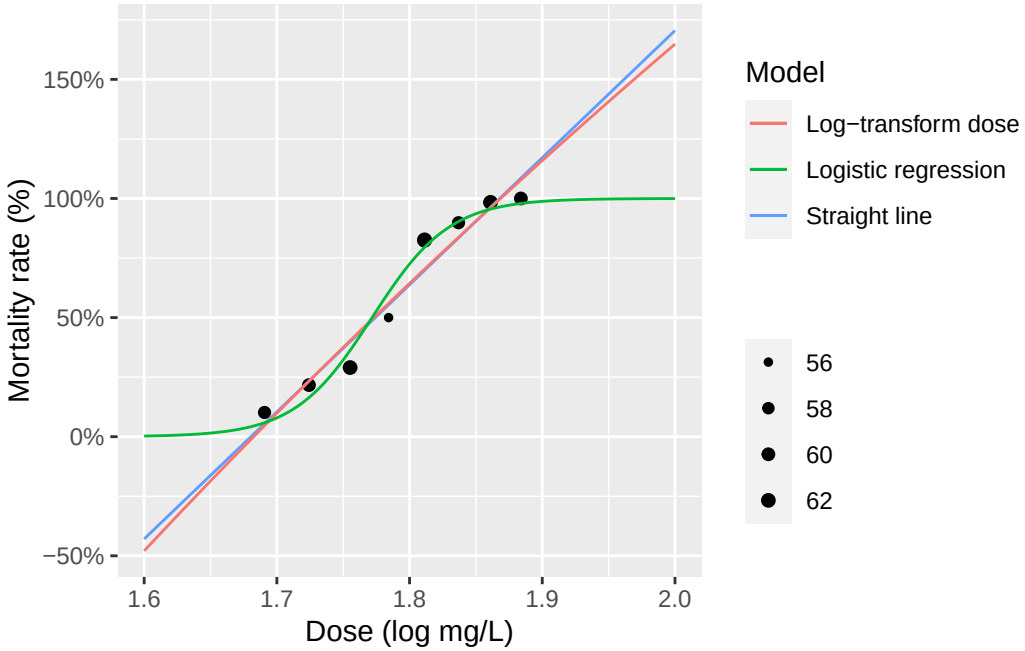


Figure 5: Mortality rates of adult flour beetles after five hours' exposure to gaseous carbon disulphide (Bliss 1935)

0.3.10 Three parts to regression models

- What distribution does the outcome have for a specific subpopulation defined by covariates? (outcome model)
- How does the combination of covariates relate to the mean? (link function)
- How do the covariates combine? (linear predictor, interactions)

Generalized Linear Models

This section is primarily adapted starting from the textbook "An Introduction to Generalized Linear Models" (4th edition, 2018) by Annette J. Dobson and Adrian G. Barnett:

https://www.routledge.com/An-Introduction-to-Generalized-Linear-Models/Dobson-Barnett/p/book/9781138741515

The type of predictive model one uses depends on several issues; one is the type of response.

- Measured values such as quantity of a protein, age, weight usually can be handled in an ordinary linear regression model, possibly after a log transformation.
- Patient survival, which may be censored, calls for a different method (survival analysis, Cox regression).
- If the response is binary, then can we use logistic regression models
- If the response is a count, we can use Poisson regression
- If the count has a higher variance than is consistent with the Poisson, we can use a negative binomial or over-dispersed Poisson
- Other forms of response can generate other types of generalized linear models

We need a linear predictor of the same form as in linear regression x. In theory, such a linear predictor can generate any type of number as a prediction, positive, negative, or zero

We choose a suitable distribution for the type of data we are predicting (normal for any number, gamma for positive numbers, binomial for binary responses, Poisson for counts)

We create a link function which maps the mean of the distribution onto the set of all possible linear prediction results, which is the whole real line (-∞, ∞). The inverse of the link function takes the linear predictor to the actual prediction.

- Ordinary linear regression has identity link (no transformation by the link function) and uses the normal distribution
- If one is predicting an inherently positive quantity, one may want to use the log link since ex is always positive.
- An alternative to using a generalized linear model with a log link, is to transform the data using the log. This is a device that works well with measurement data and may be usable in other cases, but it cannot be used for 0/1 data or for count data that may be 0.

Table 1: R glm() Families

Family	Links
gaussian	**identity**, log, inverse
binomial	**logit**, probit, cauchit, log, cloglog
gamma	**inverse**, identity, log
inverse.gaussian	**1/mu^2**, inverse, identity, log
Poisson	**log**, identity, sqrt
quasi	**identity**, logit, probit, cloglog, inverse, log, 1/mu^2 and sqrt

Family	Links
quasibinomial	**logit**, probit, identity, cloglog, inverse, log, 1/mu^2 and sqrt
quasipoisson	**log**, identity, logit, probit, cloglog, inverse, 1/mu^2 and sqrt

0.4 Linear (Gaussian) Models

> **Note**
>
> This content is adapted from Dobson & Barnett, An Introduction to Generalized Linear Models, 4th edition, Chapters 2-6]

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(dplyr) # manipulate data
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(conflicted) # check for conflicting function definitions
```

0.5 Understanding Gaussian Linear Regression Models

0.5.1 Motivating example: birthweights and gestational age

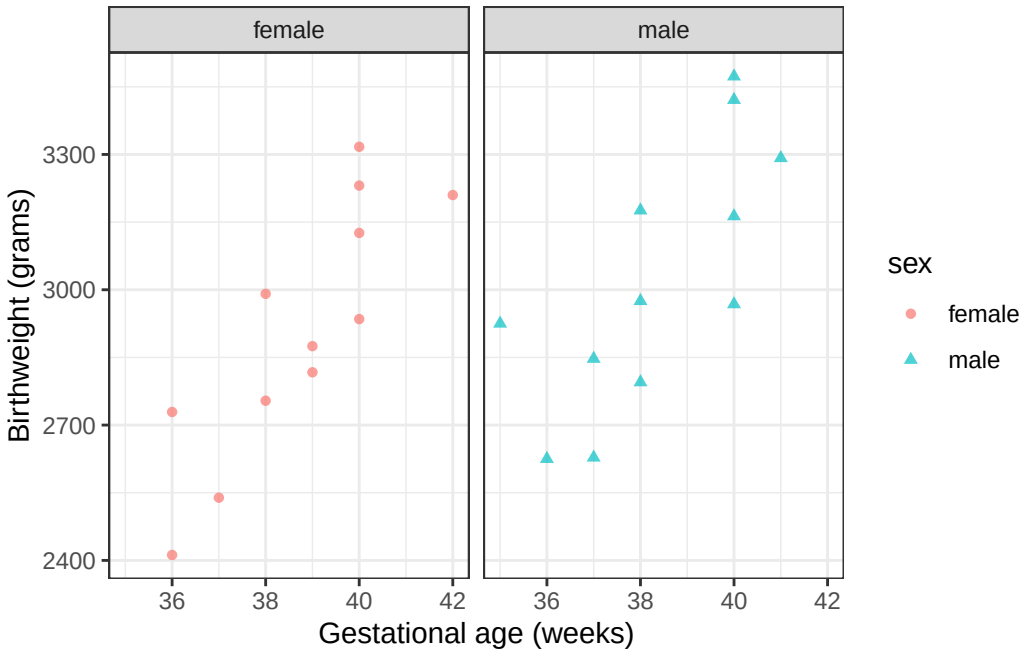
Suppose we want to learn about the distributions of birthweights for (human) babies born at different gestational ages and with different chromosomal sexes (Dobson and Barnett, Example 2.2.2):

```
data("birthweight", package = "dobson")
```

```r
bw =
  birthweight |>
  pivot_longer(
    cols = everything(),
    names_to = c("sex", ".value"),
    names_sep = "s "
  ) |>
  mutate(
    sex = ifelse(sex == "boy", "male", "female"),
    male = (sex == "male") |> as.integer())  |>
  rename(age = `gestational age`)
```

```r
plot1 = bw |>
  ggplot(aes(
    x = age,
    y = weight,
    linetype = sex,
    shape = sex,
    col = sex))  +
  theme_bw() +
  xlab("Gestational age (weeks)") +
  ylab("Birthweight (grams)") +
  # expand_limits(y = 0, x = 0) +
  geom_point(alpha = .7)
```

```r
print(plot1 + facet_wrap(~ sex))
```



0.5.2 Parallel lines regression

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.

0.5.2.1 Notation

Let:

- Y represent birthweight (measured in grams)
- X_1 represent chromosomal sex:
  - $X_1 = 0$ if female (XX)
  - $X_1 = 1$ if male (XY)
- X_2 represent gestational age at birth (measured in weeks).

> **Note**
>
> Female is the **reference level** for the categorical variable X_1 (chromosomal sex). 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.

Now, consider the following model:

$$Y \sim N(\mu(X_1, X_2), \sigma^2)$$

$$\mu(X_1, X_2) \stackrel{\text{def}}{=} E[Y|X_1, X_2] = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

0.5.2.2 Implementing the Model in R

Here's how we can implement this model in R:

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

bw_lm1 |>
  parameters(show_sigma = TRUE) |>
  print_md()
```

Parameter	Coefficient	SE	95% CI	t(21)	p
(Intercept)	-1773.32	794.59	(-3425.75, -120.89)	-2.23	0.037
sex (male)	163.04	72.81	(11.63, 314.45)	2.24	0.036
age	120.89	20.46	(78.34, 163.45)	5.91	$< .001$

Here's how this model looks, superimposed on the data:

```
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)
```

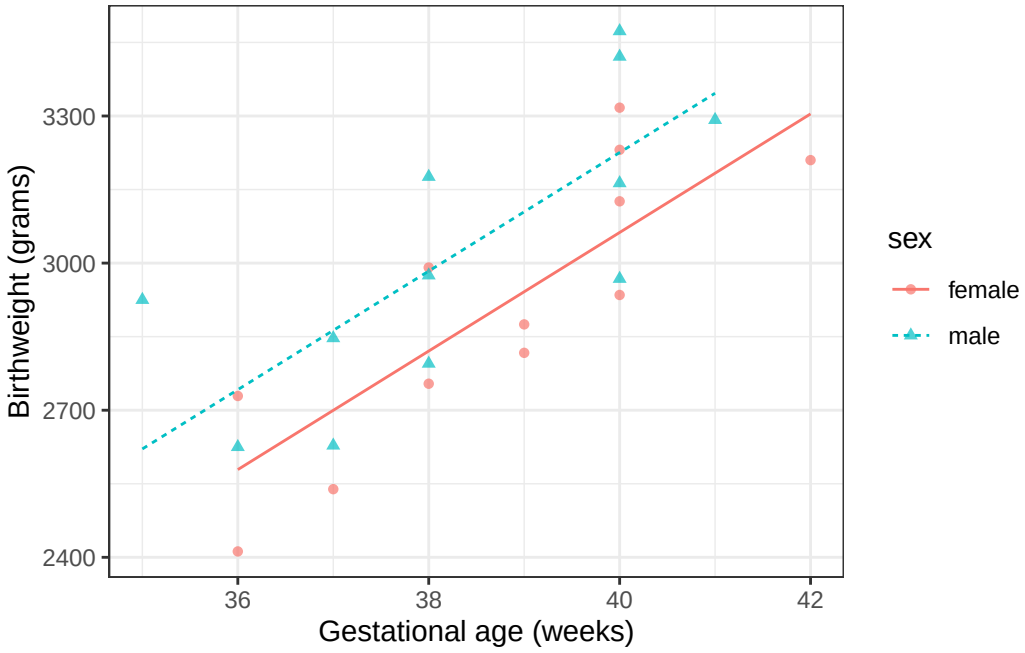


Figure 6: Parallel-slopes model of birthweight

0.5.2.3 Model assumptions and predictions

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

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

```
pred_female = coef(bw_lm1)["(Intercept)"] + coef(bw_lm1)["age"]*36

# print(pred_female)
## built-in prediction:
# predict(bw_lm1, newdata = tibble(sex = "female", age = 36))
```

$$E[Y|X_1 = 0, X_2 = 36] = \beta_0 + \beta_1 \cdot 0 + \beta_2 \cdot 36 = 2578.8739$$

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

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

$$E[Y|X_1 = 1, X_2 = 36] = \beta_0 + \beta_1 \cdot 1 + \beta_2 \cdot 36 = 2741.9132$$

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

$$\begin{aligned} &E[Y|X_1 = 1, X_2 = 36] - E[Y|X_1 = 0, X_2 = 36] \\ &= 2741.9132 - 2578.8739 \\ &= 163.0393 \end{aligned}$$

Shortcut:

$$\begin{aligned} &E[Y|X_1 = 1, X_2 = 36] - E[Y|X_1 = 0, X_2 = 36] \\ &= (\beta_0 + \beta_1 \cdot 1 + \beta_2 \cdot 36) - (\beta_0 + \beta_1 \cdot 0 + \beta_2 \cdot 36) \\ &= \beta_1 \\ &= 163.0393 \end{aligned}$$

Note that age doesn't show up 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.

That's an assumption of the model; it's built-in to the parametric structure, even before we plug in the estimated values of those parameters.

That's why the lines are parallel.

0.5.3 Interactions

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

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

$$E[Y|X_1, X_2] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 (X_1 \cdot X_2)$$

```r
bw_lm2 = lm(weight ~ sex + age + sex:age, data = bw)
```

Here are the estimated parameters (βs):

```
bw_lm2 |>
  parameters() |>
  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) $\times$ age	-18.42	41.76	(-105.52, 68.68)	-0.44	0.664

Here's another way we could rewrite this model (by collecting terms involving X_2):

$$E[Y|X_1, X_2] = \beta_0 + \beta_1 X_1 + (\beta_2 + \beta_3 X_1) X_2$$

> **Note**
>
> If you want to understand a coefficient in a model with interactions, collect terms for the corresponding variable, and you will see what other variables are interacting with the variable you are interested in.

In this case, the coefficient X_2 is interacting with X_1. So the slope of Y with respect to X_2 depends on the value of X_2.

There is no longer a slope; we can't talk about "the slope of birthweight with respect to age". 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".

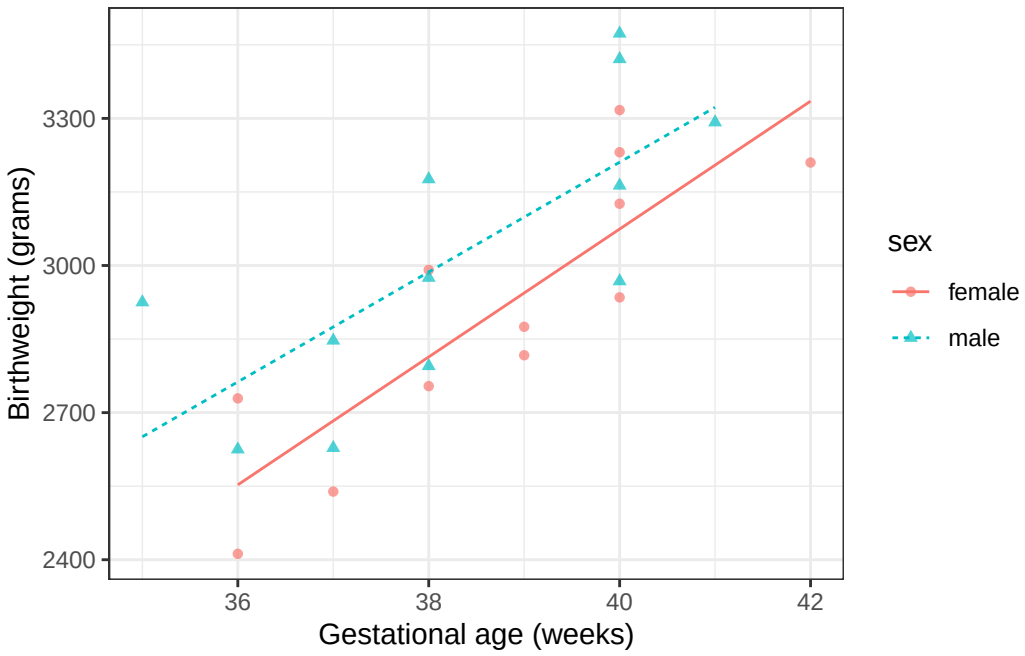
Then: that coefficient is the difference in means per unit change in its corresponding coefficient, when the other collected variables are set to 0.

Here's how this model looks, superimposed on the data:

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

plot1_interact =
```

```
  plot1 %+% bw +
  geom_line(aes(y = predlm2))
```

```
print(plot1_interact)
```



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

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

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

```
pred_female = coef(bw_lm2)["(Intercept)"] + coef(bw_lm2)["age"]*36
```

$$E[Y|X_1 = 0, X_2 = 36] = \beta_0 + \beta_1 \cdot 0 + \beta_2 \cdot 36 + \beta_3 \cdot (0 * 36) = 2552.7333$$

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

```
pred_male =
  coef(bw_lm2)["(Intercept)"] +
  coef(bw_lm2)["sexmale"] +
  coef(bw_lm2)["age"]*36 +
```

```
coef(bw_lm2)["sexmale:age"] * 36
```

$$E[Y|X_1 = 0, X_2 = 36] = \beta_0 + \beta_1 \cdot 1 + \beta_2 \cdot 36 + \beta_3 \cdot 1 \cdot 36 = 2762.7069$$

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

$$\begin{aligned}
&E[Y|X_1 = 1, X_2 = 36] - E[Y|X_1 = 0, X_2 = 36] \\
&= (\beta_0 + \beta_1 \cdot 1 + \beta_2 \cdot 36 + \beta_3 \cdot 1 \cdot 36) \\
&\quad - (\beta_0 + \beta_1 \cdot 0 + \beta_2 \cdot 36 + \beta_3 \cdot 0 \cdot 36) \\
&= \beta_2 + \beta_3 \cdot 36 \\
&= 209.9736
\end{aligned}$$

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.

0.5.4 Stratified regression

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

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

bw_lm_strat |>
  parameters() |>
  print_md()
```

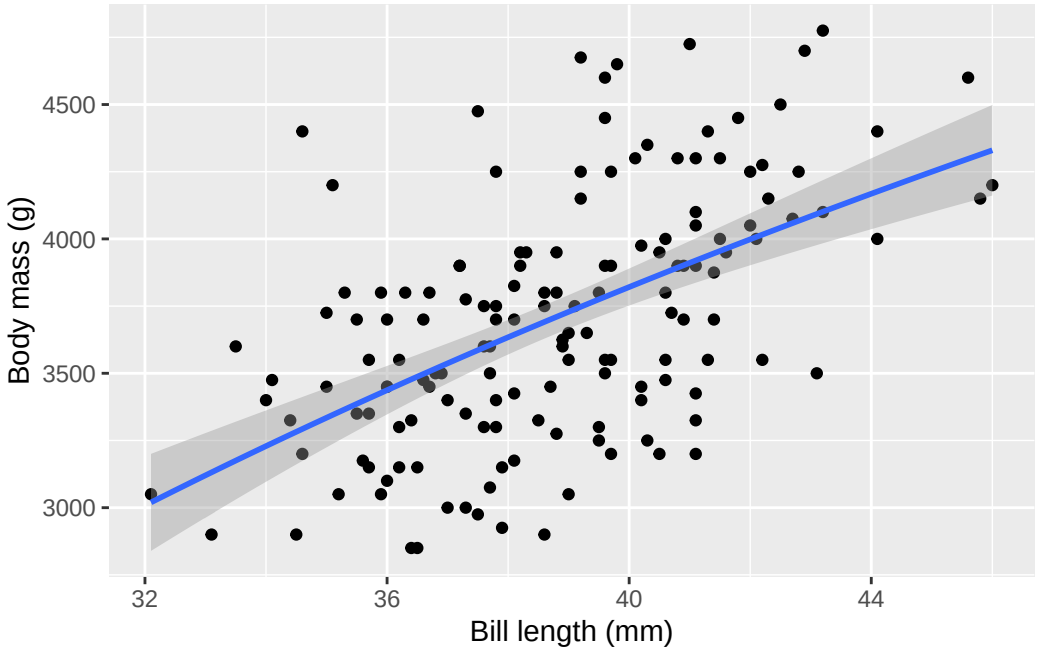
Parameter	Coefficient	SE	95% CI	t(20)	p
sex (female)	-2141.67	1163.60	(-4568.90, 285.56)	-1.84	0.081
sex (male)	-1268.67	1114.64	(-3593.77, 1056.42)	-1.14	0.268
sex (female) × age	130.40	30.00	(67.82, 192.98)	4.35	< .001
sex (male) × age	111.98	29.05	(51.39, 172.57)	3.86	< .001

0.5.5 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)
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()
```



0.6 Estimating Linear Models via Maximum Likelihood

0.6.1 Review of one-sample inference

Previously, we learned how to fit outcome-only models of the form $p(X = x|\theta)$ to iid data $\mathbf{x} = (x_1, ..., x_n)$ using maximum likelihood estimation:

$$\mathcal{L}(\mathbf{x}|\theta) = p(X_1 = x_1, ..., X_n = x_n|\theta) = \prod_{i=1}^{n} p(X = x_i|\theta)$$

$$\ell(x|\theta) = \log\{\mathcal{L}(x|\theta)\}$$

$$\hat{\theta}_{ML} = \arg\max_{\theta} \ell(x|\theta)$$

We learned how to quantify our uncertainty about these maximum likelihood estimates; with sufficient sample size, $\hat{\theta}_{ML}$ has the approximate distribution:

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

For models in the "exponential family" of distributions, which includes the Gaussian, Poisson, Bernoulli, Binomial, Exponential, and Gamma distributions, $\mathcal{I}(\theta) = \text{-}E[l''(X|\theta)]$, so we estimated $\mathcal{I}(\theta)$ using either $\mathcal{I}(\theta)|_{\theta=\hat{\theta}_{ML}}$ or $l''(\mathbf{x}|\theta)|_{\theta=\hat{\theta}_{ML}}$.

Then an asymptotic approximation of a 95% confidence interval for θ_k is

$$\hat{\theta}_{ML} \pm z_{0.975} \times \left[\left(\hat{\mathcal{I}}(\hat{\theta}_{ML})\right)^{-1}\right]_{kk}$$

where z_β the β quantile of the standard Gaussian distribution.

0.6.2 MLEs for Linear Regression

Let's use maximum likelihood again:

$$\mathcal{L}(\mathbf{y}|\mathbf{x}, \beta, \sigma^2) = \prod_{i=1}^{n}(2\pi\sigma^2)^{-1/2}\exp\left\{-\frac{1}{2\sigma^2}(y_i - x_i'\beta)^2\right\}$$

$$\ell(\mathbf{y}|\mathbf{x}, \beta, \sigma^2) \propto -\frac{n}{2}\log\{\sigma^2\} - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i'\beta)^2$$

$$\ell'(\mathbf{y}|\mathbf{x}, \beta, \sigma^2) \propto -\frac{n}{2}\log\{\sigma^2\} - \frac{1}{2\sigma^2}\frac{d}{d\beta}\left(\sum_{i=1}^{n}(y_i - x_i'\beta)^2\right)$$

A few tools from linear algebra will make this analysis go easier (see Fieller, Section 7.2 for details).

$$f_\beta(\mathbf{x}) = (f_\beta(x_1), f_\beta(x_2), ..., f_\beta(x_n))^\top$$

Let $\mathbf{x}$ and β be vectors of length p, or in other words, matrices of length $p \times 1$:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}$$

Then

$$x' \equiv x^\top \equiv [x_1, x_2, ..., x_p]$$

and

$$x'\beta = [x_1, x_2, ..., x_p] \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} = x_1\beta_1 + x_2\beta_2 + ... + x_p\beta_p$$

If $f(\beta)$ is a function that takes β as input and outputs a scalar, such as $f(\beta) = x'\beta$, then:

$$\frac{d}{d\beta} f(\beta) = \begin{bmatrix} \frac{d}{d\beta_1} f(\beta) \\ \frac{d}{d\beta_2} f(\beta) \\ \vdots \\ \frac{d}{d\beta_p} f(\beta) \end{bmatrix}$$

In particular, if $f(\beta) = x'\beta$, then:

$$\frac{d}{d\beta} x'\beta = \begin{bmatrix} \frac{d}{d\beta_1}(x_1\beta_1 + x_2\beta_2 + ... + x_p\beta_p) \\ \frac{d}{d\beta_2}(x_1\beta_1 + x_2\beta_2 + ... + x_p\beta_p) \\ \vdots \\ \frac{d}{d\beta_p}(x_1\beta_1 + x_2\beta_2 + ... + x_p\beta_p) \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} = \mathbf{x}$$

In general:

$$\frac{d}{d\beta} x'\beta = x$$

This looks a lot like non-vector calculus, except that you have to transpose the coefficient.

Similarly,

$$\frac{d}{d\beta} \beta'\beta = 2\beta$$

This is like taking the derivative of x^2.

And finally, if S is a $p \times p$ matrix, then:

$$\frac{d}{d\beta} \beta' S\beta = 2S\beta$$

Again, this is like taking the derivative of cx^2 with respect to c in non-vector calculus.

Thus:

$$\sum_{i=1}^{n}(y_i - f_\beta(x_i))^2 = (\mathbf{y} - X\beta)'(\mathbf{y} - X\beta)$$

$$(\mathbf{y} - X\beta)' = (\mathbf{y}' - (X\beta)') = (\mathbf{y}' - \beta' X')$$

So

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

So

$$\begin{aligned}\frac{d}{d\beta}\left(\sum_{i=1}^{n}(y_i - x_i'\beta)^2\right) &= \frac{d}{d\beta}(\mathbf{y} - X\beta)'(\mathbf{y} - X\beta) \\ &= \frac{d}{d\beta}(y'y - 2y' X\beta + \beta' X' X\beta) \\ &= (-2X' y + 2X' X\beta)\end{aligned}$$

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

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

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

Similarly (not shown):

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

And

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

So:

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

and

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

These are all results you have hopefully seen before, and in the Gaussian linear regression case they are exact, not just approximate.

In our model 2 above, this matrix is:

```
bw_lm2 |> vcov()
```

```
            (Intercept)  sexmale      age sexmale:age
(Intercept)     1353968 -1353968 -34871.0     34871.0
sexmale        -1353968  2596387  34871.0    -67211.0
age              -34871    34871    899.9      -899.9
sexmale:age       34871   -67211   -899.9      1743.5
```

Note that 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
    1163.60     1611.33       30.00       41.76
```

```
bw_lm2 |> summary() |> coef() |> pander()
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-2142	1164	-1.841	0.08057
sexmale	873	1611	0.5418	0.594

	Estimate	Std. Error	t value	Pr(>|t|)
age	130.4	30	4.347	0.0003127
sexmale:age	-18.42	41.76	-0.4411	0.6639

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

0.7 Inference about Gaussian Linear Regression Models

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

$H_0 : \beta_3 = 0$

$H_A : \beta_3 \neq 0$

$P(|\hat{\beta}_3| > |-18.4172| \mid H_0) = ?$

0.7.1 Wald tests and CIs

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

```
bw_lm2 |>
  parameters(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

0.7.2 One-parameter inference by hand

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

0.7.2.1 P-value

```
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(abs(t_stat), df = dfresid, lower = FALSE) * 2
```

$$
\begin{aligned}
P\left(|\hat{\beta}_3| > |-18.4172| \middle| H_0\right) &= P\left(\left|\frac{\hat{\beta}_3}{\widehat{SE}(\hat{\beta}_3)}\right| > \left|\frac{-18.4172}{41.7558}\right| \middle| H_0\right) \\
&= P\left(|T_{20}| > 0.4411 | H_0\right) \\
&= 0.6639
\end{aligned}
$$

This matches the result in the table above.

0.7.2.2 Confidence interval

```
confint_radius_t = se_hat * qt(p = 0.975, df = dfresid, lower = TRUE)
confint_t = beta_hat + c(-1,1) * confint_radius_t
print(confint_t)
```

```
[1] -105.52   68.68
```

This also matches.

0.7.3 Gaussian approximations

Here are the asymptotic (Gaussian approximation) equivalents:

0.7.3.1 P-value

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

print(pval_z)
```

```
[1] 0.6592
```

0.7.3.2 Confidence interval

```
confint_radius_z = se_hat * qnorm(0.975, lower = TRUE)
confint_z =
  beta_hat + c(-1,1) * confint_radius_z
print(confint_z)
```

```
[1] -100.26   63.42
```

0.7.4 Likelihood ratio statistics

```
logLik(bw_lm2)
```

```
'log Lik.' -156.6 (df=5)
```

```
logLik(bw_lm1)
```

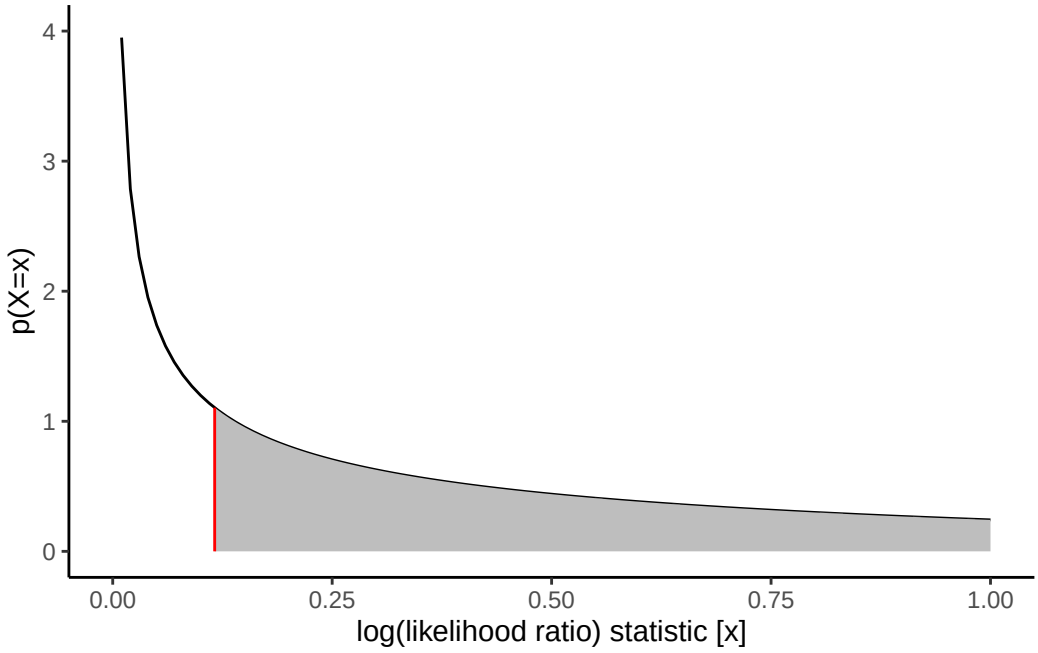
```
'log Lik.' -156.7 (df=4)
```

```
lLR = (logLik(bw_lm2) - logLik(bw_lm1)) |> as.numeric()
delta_df = (bw_lm1$df.residual - df.residual(bw_lm2))

d_lLR = function(x, df = delta_df) dchisq(x, df = df)

x_max = 1

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

```
pchisq(
  q = 2*lLR,
  df = delta_df,
  lower = FALSE)
```

```
[1] 0.6298
```

```
chisq_plot |> print()
```



Now we can get the p-value:

```
pchisq(2*lLR, df = delta_df, lower = FALSE)
```

```
[1] 0.6298
```

0.7.5

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

```r
# built in
library(lmtest)
lrtest(bw_lm2, bw_lm1)
```

```
Likelihood ratio test

Model 1: weight ~ sex + age + sex:age
Model 2: weight ~ sex + age
  #Df LogLik Df Chisq Pr(>Chisq)
1   5   -157
2   4   -157 -1  0.23       0.63
```

0.8 Goodness of fit

0.8.1 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):

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

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:

0.8.1.1 AIC in R

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

```
[1] 323.2
```

```
AIC(bw_lm2)
```

```
[1] 323.2
```

0.8.1.2 BIC in R

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

```
[1] 329
```

```
BIC(bw_lm2)
```

```
[1] 329
```

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

0.8.2 (Residual) Deviance

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

```
bw.X.unique =
  bw |>
  count(sex, age)

n_unique.bw  = nrow(bw.X.unique)
```

For example, in the `birthweight` data, there are $q = 12$ unique patterns:

```
bw.X.unique |>
  pander(
    row.names = rownames(bw.X.unique))
```

	sex	age	n
1	female	36	2

	sex	age	n
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

> **Note**
>
> If a given covariate pattern has more than one observation in a dataset, those observations are called **replicates**.

Then the most complicated model we could fit would have one parameter (a mean) for each of these combinations, plus a variance parameter:

```
lm_max =
  bw |>
  mutate(age = factor(age)) |>
  lm(
    formula = weight ~ sex:age - 1,
    data = _)

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

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

Parameter	Coefficient	SE	95% CI	t(12)	p
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.

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

```
logLik(lm_max)
```

```
'log Lik.' -151.4 (df=13)
```

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

```
lrtest(lm_max, bw_lm2)
```

```
Likelihood ratio test

Model 1: weight ~ sex:age - 1
Model 2: weight ~ sex + age + sex:age
  #Df LogLik Df Chisq Pr(>Chisq)
1  13   -151
2   5   -157 -8  10.4       0.24
```

The likelihood ratio statistic for this test is

$$\lambda = 2 * (\ell_{\text{full}} - \ell) = 10.3554$$

where:

- ℓ_{max} is the log-likelihood of the full model: -151.4016
- ℓ is the log-likelihood of our comparison model (two slopes, two intercepts): -156.5793

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.)

0.8.3 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	$< .001$

This model also has a likelihood:

```
logLik(lm0)
```

```
'log Lik.' -169 (df=2)
```

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

```
lrtest(bw_lm2, lm0)
```

```
Likelihood ratio test

Model 1: weight ~ sex + age + sex:age
Model 2: weight ~ 1
  #Df LogLik Df Chisq Pr(>Chisq)
```

```
1   5   -157
2   2   -169 -3  24.8    1.7e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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

$$\lambda = 2 * (\ell_{\text{full}} - \ell_0) = 35.1067$$

where:

- ℓ_0 is the log-likelihood of the null model: -168.955
- ℓ_{full} is the log-likelihood of the maximal model: -151.4016

In R, this test is:

```r
lrtest(lm_max, lm0)
```

```
Likelihood ratio test

Model 1: weight ~ sex:age - 1
Model 2: weight ~ 1
  #Df LogLik  Df Chisq Pr(>Chisq)
1  13   -151
2   2   -169 -11  35.1    0.00024 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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.

> **Caution**
>
> This definition is a bit different from what I said in class (4/25); then, I said the null deviance was twice the difference in log-likelihood between the model of interest and the null model. The one I have written above is the standard one, and we'll stick to the standard definition going forward.

0.9 Rescaling

0.9.1 Rescale age

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

lm1c = lm(weight ~ sex + `age - 36wks`, data = bw)

lm2c = lm(weight ~ sex + `age - 36wks` + sex:`age - 36wks`, data = bw)

parameters(lm2c, 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

0.10 Prediction

0.10.1 Prediction for linear models

$$\begin{aligned}\hat{Y} &= \hat{E}[Y|X = x] \\ &= x'\hat{\beta} \\ &= \hat{\beta}_0 \cdot 1 + \hat{\beta}_1 x_1 + ... + \hat{\beta}_p x_p\end{aligned}$$

0.10.2 prediction in R

```
X = model.matrix(bw_lm1)
X |> as_tibble() |> print()
```

```
# A tibble: 24 x 3
   `(Intercept)` sexmale   age
           <dbl>   <dbl> <dbl>
 1             1       1    40
 2             1       0    40
 3             1       1    38
 4             1       0    36
 5             1       1    40
 6             1       0    40
 7             1       1    35
 8             1       0    38
 9             1       1    36
10             1       0    42
# i 14 more rows
```

```
print(X[1,])
```

```
(Intercept)     sexmale         age
          1           1          40
```

```
print(coef(bw_lm1))
```

```
(Intercept)     sexmale         age
    -1773.3       163.0       120.9
```

```r
print(X[1,] * coef(bw_lm1))
```

```
(Intercept)     sexmale         age
      -1773         163        4836
```

```r
{X[1,] * coef(bw_lm1)} |> sum() |> print()
```

```
[1] 3225
```

```r
X %*% coef(bw_lm1) |> as.vector()
```

```
 [1] 3225 3062 2984 2579 3225 3062 2621 2821 2742 3304 2863 2942 3346 3062 3225
[16] 2700 2863 2579 2984 2821 3225 2942 2984 3062
```

```r
predict(bw_lm1)
```

```
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
3225 3062 2984 2579 3225 3062 2621 2821 2742 3304 2863 2942 3346 3062 3225 2700
  17   18   19   20   21   22   23   24
2863 2579 2984 2821 3225 2942 2984 3062
```

```r
predict(bw_lm1, se.fit = TRUE)
```

```
$fit
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
3225 3062 2984 2579 3225 3062 2621 2821 2742 3304 2863 2942 3346 3062 3225 2700
  17   18   19   20   21   22   23   24
2863 2579 2984 2821 3225 2942 2984 3062

$se.fit
 [1] 61.46 57.17 51.58 76.03 61.46 57.17 85.25 53.38 69.96 83.89 57.95 51.38
[13] 74.78 57.17 61.46 62.42 57.95 76.03 51.58 53.38 61.46 51.38 51.58 57.17

$df
[1] 21

$residual.scale
[1] 177.1
```

```r
predict(bw_lm1, se.fit = TRUE)$fit
```

```
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
3225 3062 2984 2579 3225 3062 2621 2821 2742 3304 2863 2942 3346 3062 3225 2700
  17   18   19   20   21   22   23   24
2863 2579 2984 2821 3225 2942 2984 3062
```

```r
predict(bw_lm1, se.fit = TRUE)$se.fit
```

```
 [1] 61.46 57.17 51.58 76.03 61.46 57.17 85.25 53.38 69.96 83.89 57.95 51.38
[13] 74.78 57.17 61.46 62.42 57.95 76.03 51.58 53.38 61.46 51.38 51.58 57.17
```

```r
predict(bw_lm1, se.fit = TRUE, interval = "confidence")$fit |> as_tibble()
```

```
# A tibble: 24 x 3
     fit   lwr   upr
   <dbl> <dbl> <dbl>
 1 3225. 3098. 3353.
 2 3062. 2944. 3181.
 3 2984. 2876. 3091.
 4 2579. 2421. 2737.
 5 3225. 3098. 3353.
 6 3062. 2944. 3181.
 7 2621. 2444. 2798.
 8 2821. 2710. 2932.
 9 2742. 2596. 2887.
10 3304. 3130. 3479.
# i 14 more rows
```

```r
predict(bw_lm1, se.fit = TRUE, interval = "predict")$fit |> as_tibble()
```

```
# A tibble: 24 x 3
     fit   lwr   upr
   <dbl> <dbl> <dbl>
 1 3225. 2836. 3615.
 2 3062. 2675. 3449.
 3 2984. 2600. 3367.
```

```
 4 2579. 2178. 2980.
 5 3225. 2836. 3615.
 6 3062. 2675. 3449.
 7 2621. 2212. 3030.
 8 2821. 2436. 3205.
 9 2742. 2346. 3138.
10 3304. 2897. 3712.
# i 14 more rows
```

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). You could also supply `newdata` to get predictions for new combinations of predictors that you didn't see in your original dataset. See `?predict.lm` for more.

0.11 Diagnostics

0.11.1 Residuals

0.11.1.1 Definitions of residuals

- Residuals:
$$e_i = y_i - \hat{E}[Y|X = x]$$
- For Gaussian models, e_i can be seen as an estimate of
$$\epsilon_i = Y_i - \mathrm{E}[Y|X = x_i]$$
- Standardized residuals:
$$r_i = \frac{e_i}{\widehat{SD}(e_i)}$$
- For Gaussian models:
$$\widehat{SD}(e_i) \approx \frac{e_i}{\hat{\sigma}}$$

0.11.1.2 Characteristics of residuals

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:

$$e_i \dot{\sim}_{iid} N(0, \hat{\sigma}^2)$$

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

$$r_i \dot{\sim}_{iid} N(0, 1)$$

0.11.2 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.

First, we need to compute the residuals. R makes this easy:

0.11.2.1 (non-standardized) residuals in R

```
resid(bw_lm2)
```

```
     1       2       3       4       5       6       7       8       9      10
176.27 -140.73 -144.13  -59.53  177.47 -126.93  -68.93  242.67 -139.33   51.67
    11      12      13      14      15      16      17      18      19      20
156.67 -125.13  274.28 -137.71  -27.69 -246.69 -191.67  189.33  -11.67 -242.64
    21      22      23      24
-47.64  262.36  210.36  -30.62
```

```
# check by hand
bw$weight - fitted(bw_lm2)
```

```
     1       2       3       4       5       6       7       8       9      10
176.27 -140.73 -144.13  -59.53  177.47 -126.93  -68.93  242.67 -139.33   51.67
    11      12      13      14      15      16      17      18      19      20
156.67 -125.13  274.28 -137.71  -27.69 -246.69 -191.67  189.33  -11.67 -242.64
    21      22      23      24
-47.64  262.36  210.36  -30.62
```

Success!

0.11.2.2 Standardized residuals in R

```
rstandard(bw_lm2)
```

```
      1       2       3       4       5       6       7       8
 1.15982 -0.92601 -0.87479 -0.34723  1.03507 -0.73473 -0.39901  1.43752
      9      10      11      12      13      14      15      16
-0.82539  0.30606  0.92807 -0.87616  1.91428 -0.86559 -0.16430 -1.46376
     17      18      19      20      21      22      23      24
-1.11016  1.09658 -0.06761 -1.46159 -0.28696  1.58040  1.26717 -0.19805
```

```
resid(bw_lm2)/sigma(bw_lm2)
```

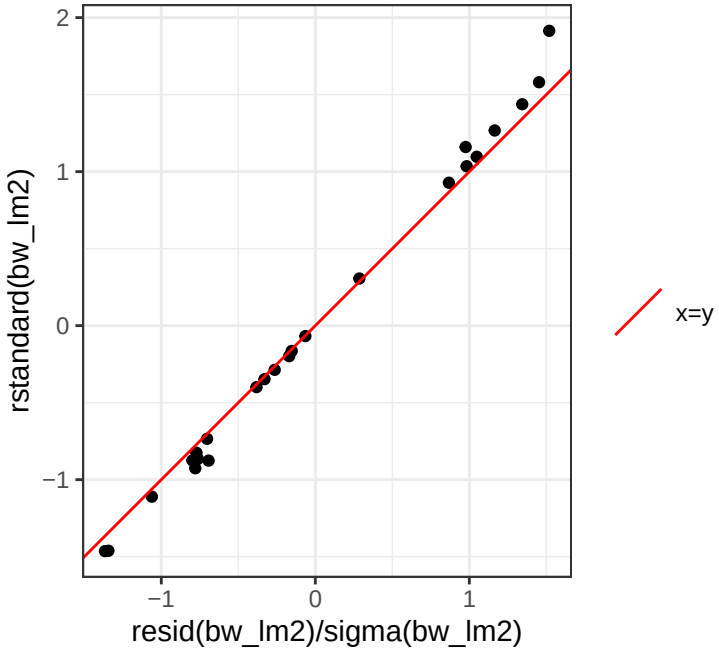
```
      1       2       3       4       5       6       7       8
 0.97593 -0.77920 -0.79802 -0.32962  0.98258 -0.70279 -0.38166  1.34357
      9      10      11      12      13      14      15      16
-0.77144  0.28606  0.86741 -0.69282  1.51858 -0.76244 -0.15331 -1.36584
     17      18      19      20      21      22      23      24
-1.06123  1.04825 -0.06463 -1.34341 -0.26376  1.45262  1.16471 -0.16954
```

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 = weight - fitted_lm2,

    std_resid_lm2 = rstandard(bw_lm2),
    # std_resid_lm2 = resid_lm2 / sigma(bw_lm2)
  )

bw |>
  select(
    sex,
    age,
    weight,
    fitted_lm2,
    resid_lm2,
    std_resid_lm2
  )
```

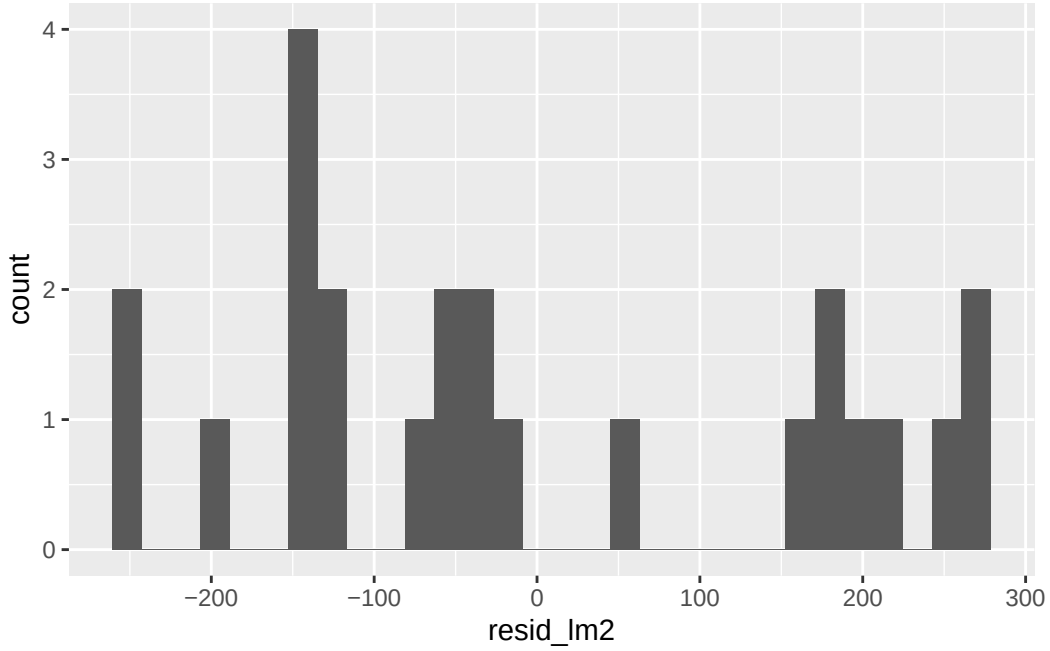
```
# A tibble: 24 x 6
   sex      age weight fitted_lm2 resid_lm2 std_resid_lm2
   <chr>  <dbl>  <dbl>      <dbl>     <dbl>         <dbl>
 1 female    36   2729      2553.     176.          1.16
 2 female    36   2412      2553.    -141.         -0.926
 3 female    37   2539      2683.    -144.         -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:

0.11.2.3 Marginal distribution of (nonstandardized) residuals

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

```
print(resid_marginal_hist)
```

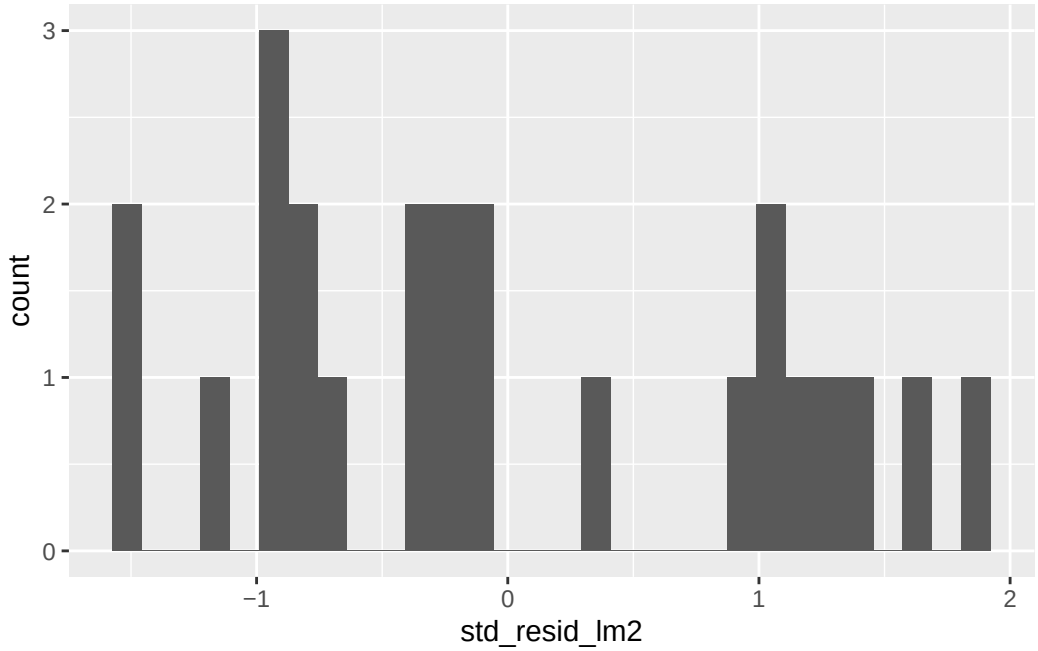


Hard to tell with this small amount of data, but I'm a bit concerned that the histogram doesn't show a bell-curve shape.

Marginal distribution of standardized residuals

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

```
print(std_resid_marginal_hist)
```



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

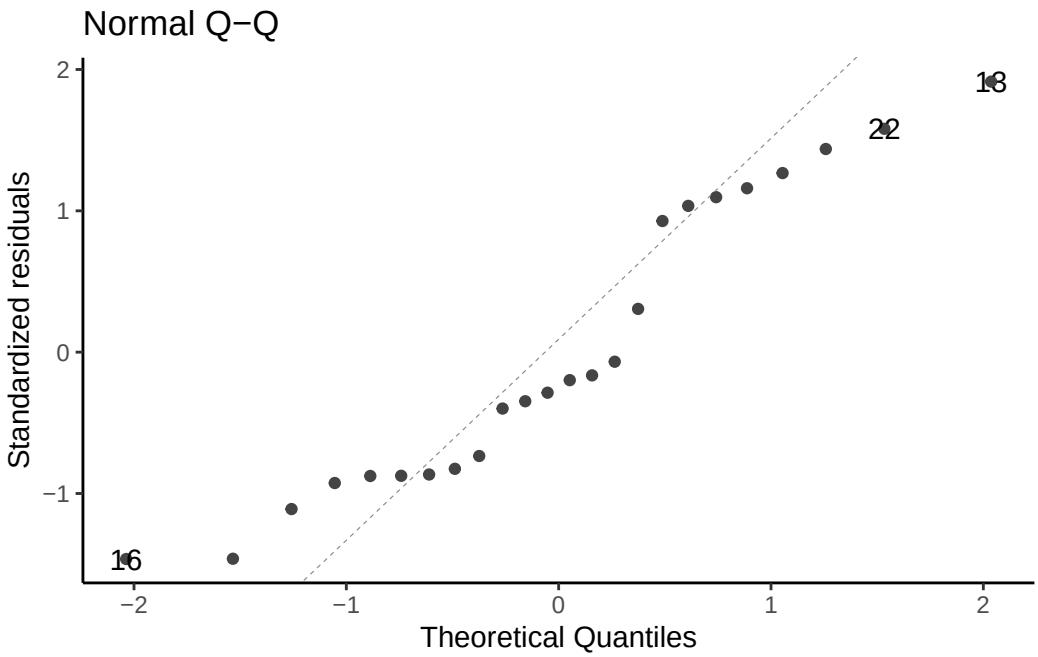
Still hard to tell if the distribution is Gaussian.

0.11.3 QQ plot of standardized residuals

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

```
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.

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

0.11.3.1 QQ plot - how it's built

Let's construct it by hand:

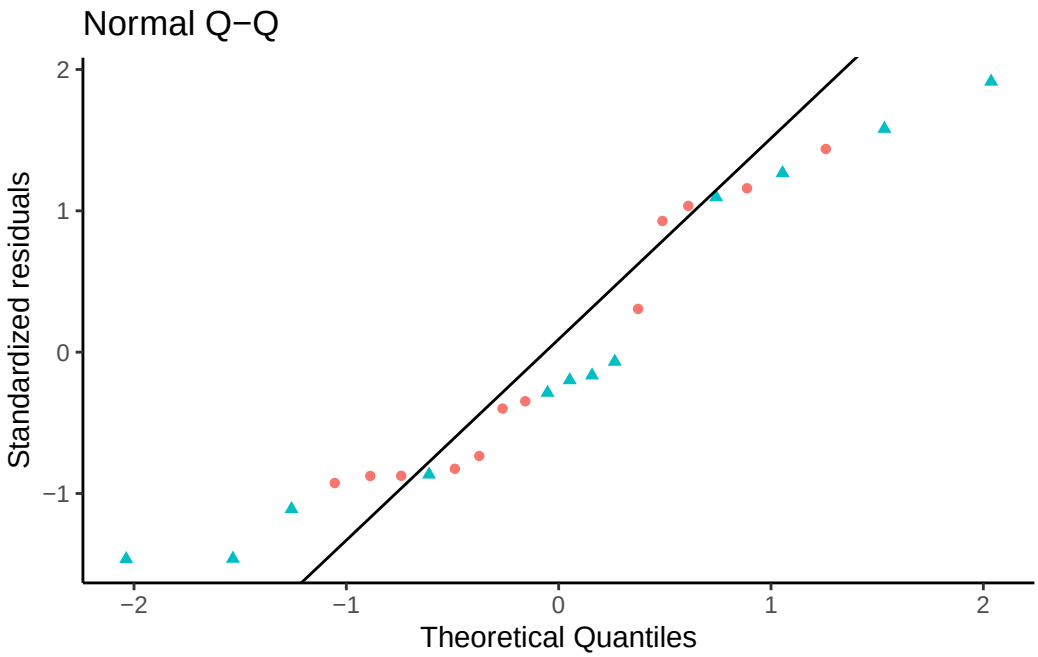
```
bw = bw |>
  mutate(
    p = (rank(std_resid_lm2) - 1/2)/n(), # "Blom's method"
    expected_quantiles_lm2 = qnorm(p)
  )

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")
```

We find the expected line like so:

```
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)
```



0.11.4 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 **regardless** of the value of X.

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

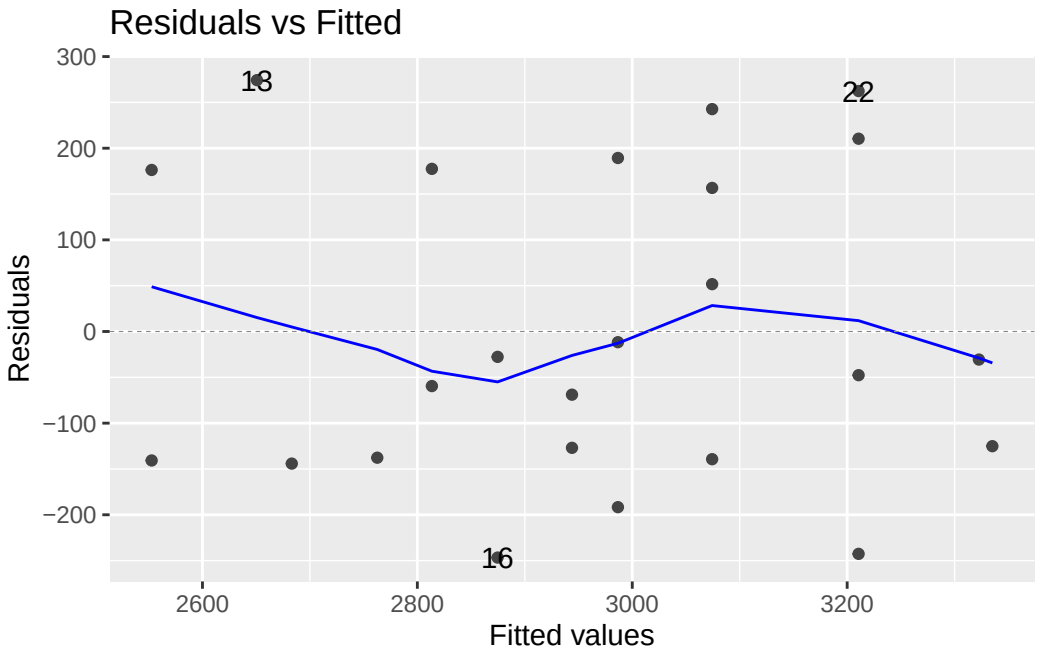
$$\mathrm{E}[Y|X = x] = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p$$

Then the the residuals might have nonzero mean or nonconstant variance for some values of x.

0.11.4.1 Residuals versus fitted values

To look for these issues, we can plot the residuals e_i against the fitted values $\hat{y}_i$:

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



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.

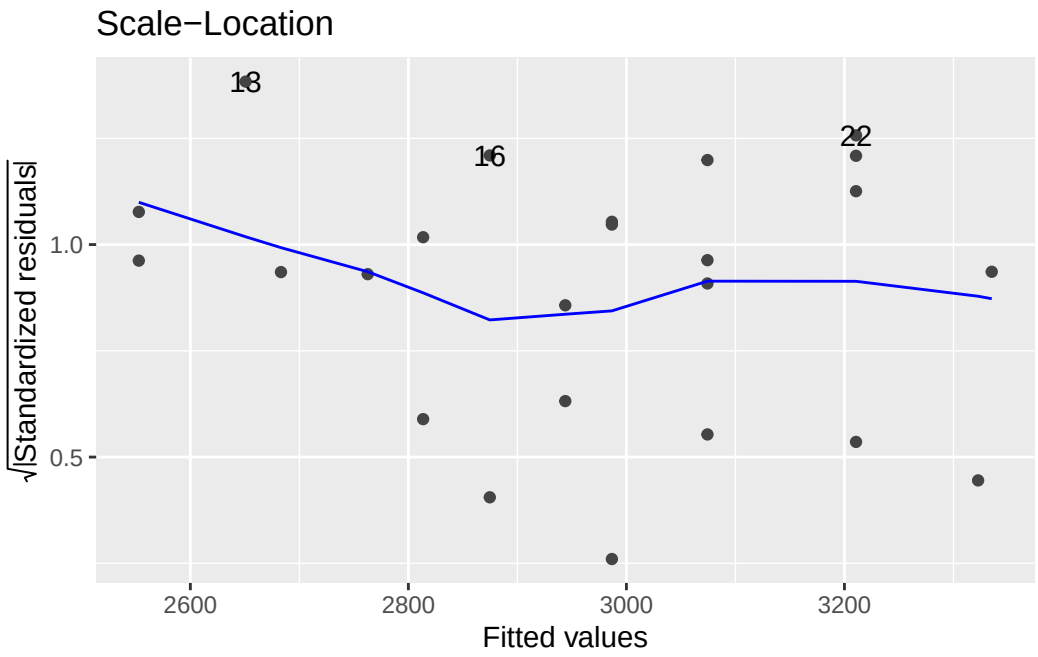
If not, we probably need to change the functional form of linear component of the mean,

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

0.11.4.2 Scale-location plot

We can also plot the square roots of the absolute values of the standardized residuals against the fitted values:

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

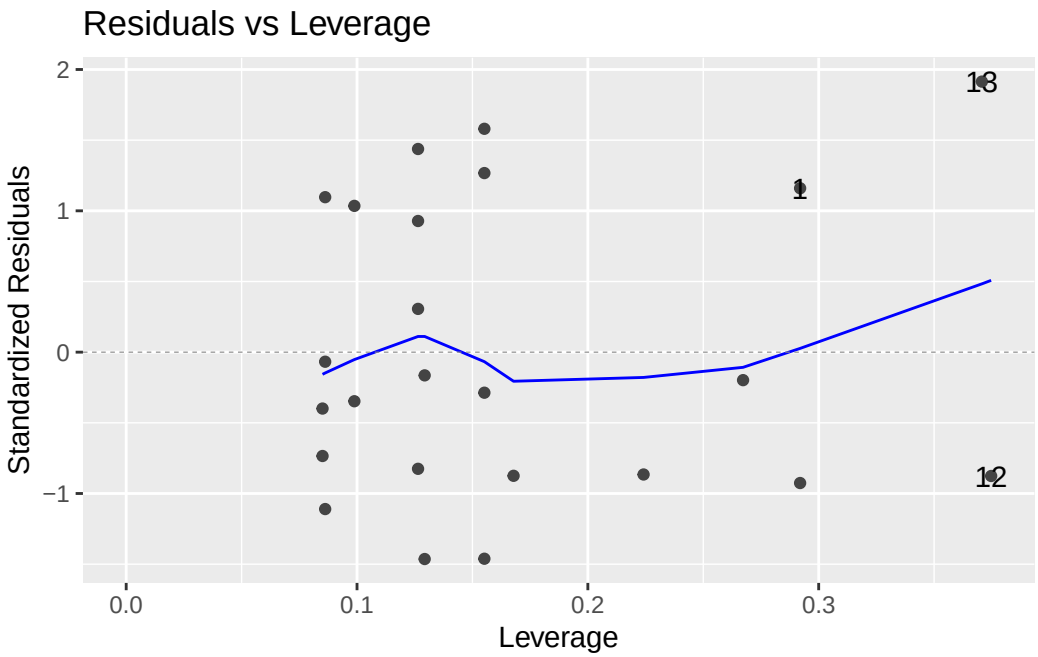


Here, the blue line doesn't need to be near 0, but it should be flat. If not, the residual variance σ^2 might not be constant, and we might need to transform our outcome Y (or use a model that allows non-constant variance).

0.11.4.3 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 ot remove those observations as outliers, particularly if they have large residuals.

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



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

0.11.5 Diagnostics constructed by hand

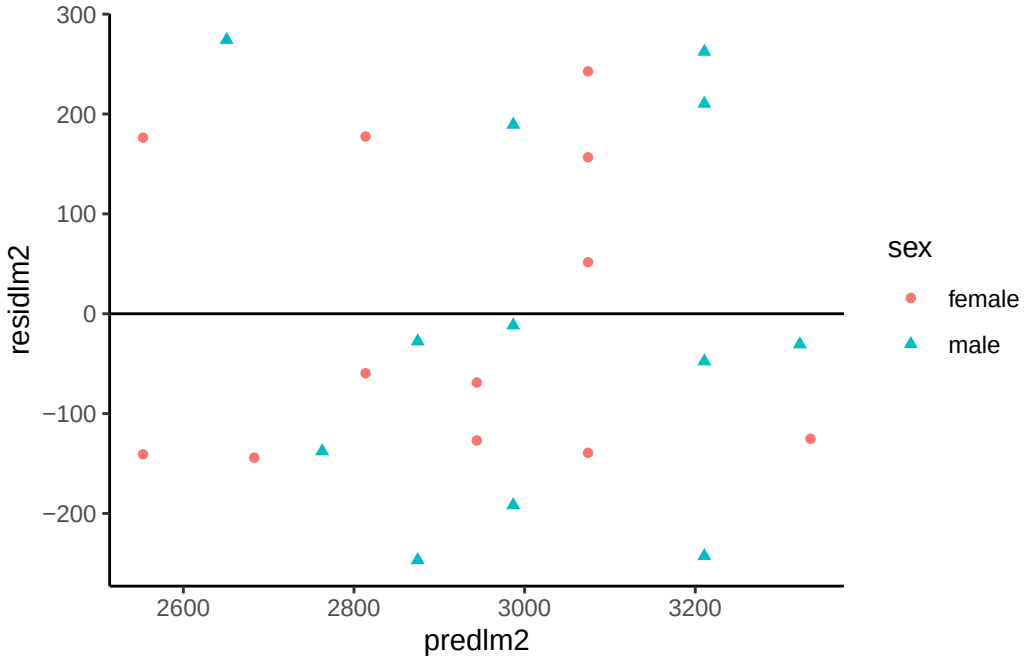
```
bw =
  bw |>
  mutate(
    predlm2 = predict(bw_lm2),
    residlm2 = weight - predlm2,
    std_resid = residlm2 / sigma(bw_lm2),
    # std_resid_builtin = rstandard(bw_lm2), # uses leverage
    sqrt_abs_std_resid = std_resid |> abs() |> sqrt()

  )
```

Residuals vs fitted

```
resid_vs_fit = bw |>
  ggplot(
    aes(x = predlm2, y = residlm2, col = sex, shape = sex)
```

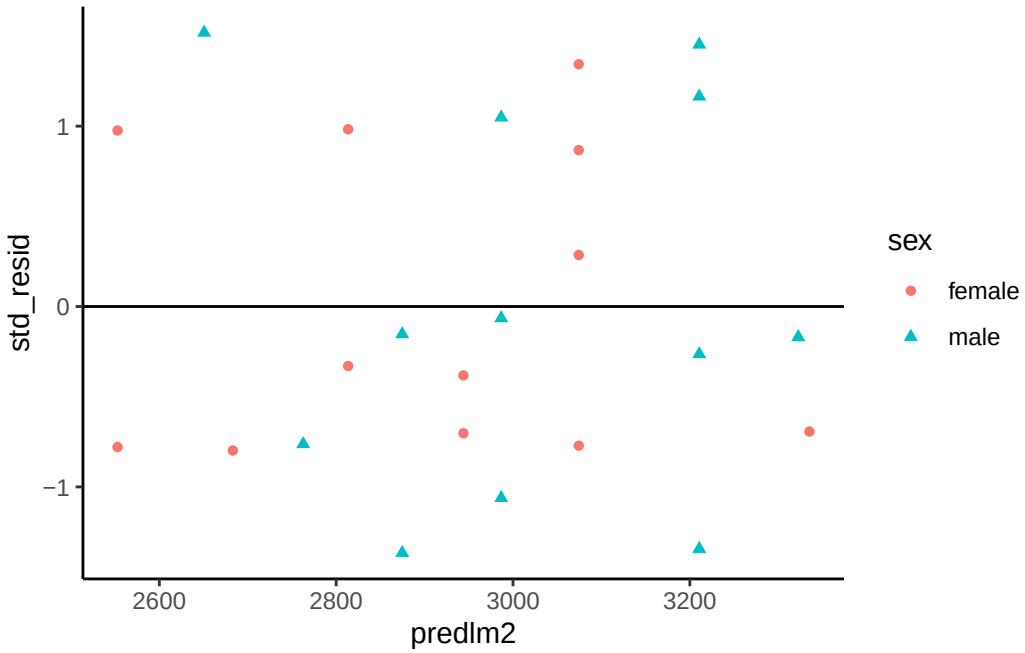
```
  ) +
  geom_point() +
  theme_classic() +
  geom_hline(yintercept = 0)
```

```
print(resid_vs_fit)
```



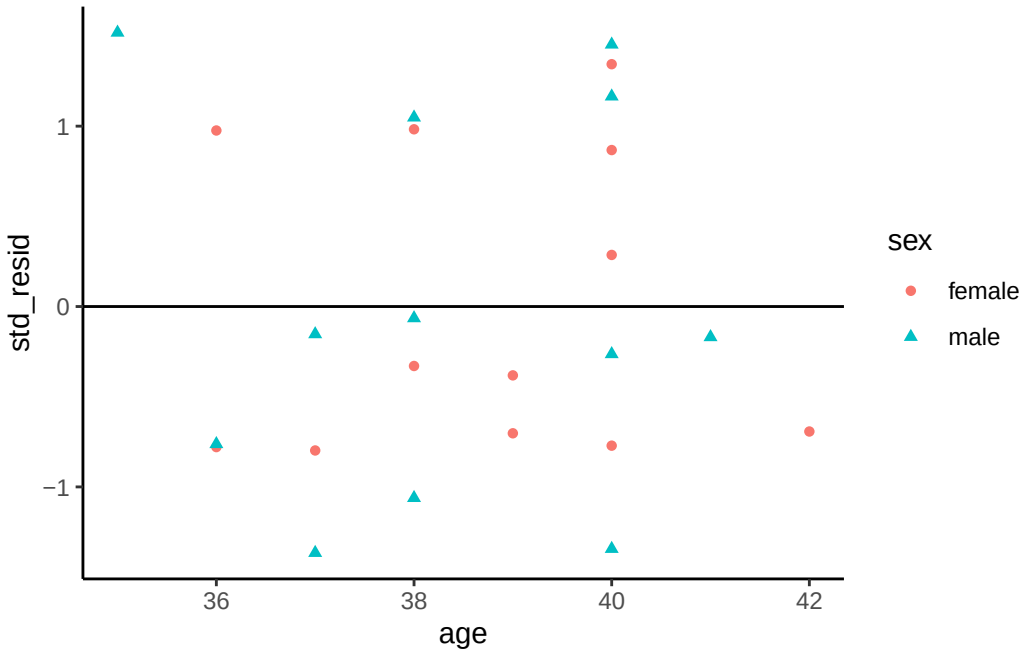
Standardized residuals vs fitted

```
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

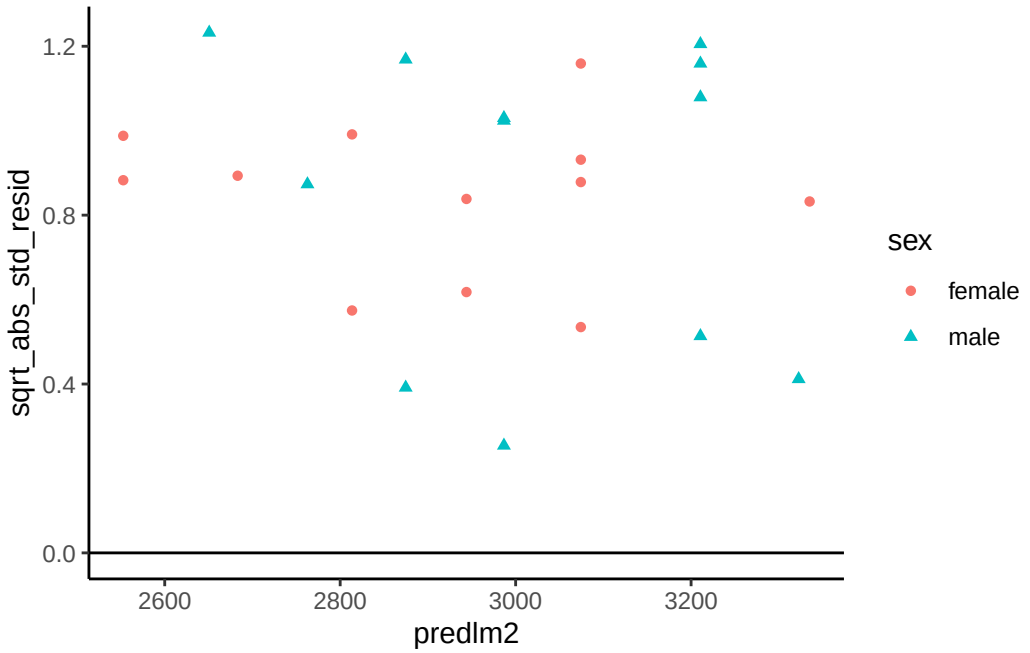
```
bw |>
  ggplot(
    aes(x = age, y = std_resid, col = sex, shape = sex)
  ) +
  geom_point() +
  theme_classic() +
  geom_hline(yintercept = 0)
```



`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)
```



0.12 Model selection

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.

0.12.1 Mean squared error

We might want to minimize the **mean squared error**, $\mathrm{E}[(y - \hat{y})^2]$, 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 $\mathrm{E}[(y - \hat{y})^2]$ for new data. If we want an unbiased estimate, we will have to be clever.

0.12.2 Cross-validation

```
data("carbohydrate", package = "dobson")
library(cvTools)
full.model <- lm(carbohydrate ~ ., data = carbohydrate)
temp =
  cvFit(full.model, data = carbohydrate, K = 5, R = 10,
y = carbohydrate$carbohydrate)
```

0.13 Categorical covariates with more than two levels

0.13.1

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

```
unique(bw$sex)
```

```
[1] "female" "male"
```

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

0.13.2

For example, here's the (in)famous `iris` data:

```
iris |> tibble()
```

```
# A tibble: 150 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
 7          4.6         3.4          1.4         0.3 setosa
```

```
 8          5           3.4          1.5         0.2 setosa
 9          4.4         2.9          1.4         0.2 setosa
10          4.9         3.1          1.5         0.1 setosa
# i 140 more rows
```

```
summary(iris)
```

```
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.30   Min.   :2.00   Min.   :1.00   Min.   :0.1   setosa    :50
 1st Qu.:5.10   1st Qu.:2.80   1st Qu.:1.60   1st Qu.:0.3   versicolor:50
 Median :5.80   Median :3.00   Median :4.35   Median :1.3   virginica :50
 Mean   :5.84   Mean   :3.06   Mean   :3.76   Mean   :1.2
 3rd Qu.:6.40   3rd Qu.:3.30   3rd Qu.:5.10   3rd Qu.:1.8
 Max.   :7.90   Max.   :4.40   Max.   :6.90   Max.   :2.5
```

0.13.3

There are three species:

```
iris$Species |> unique()
```

```
[1] setosa     versicolor virginica
Levels: setosa versicolor virginica
```

0.13.4

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 $X = 3$.

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

```
# A tibble: 3 x 2
  Species        X
  <fct>      <dbl>
1 setosa         1
2 versicolor     3
3 virginica      2
```

Then we could fit a model like:

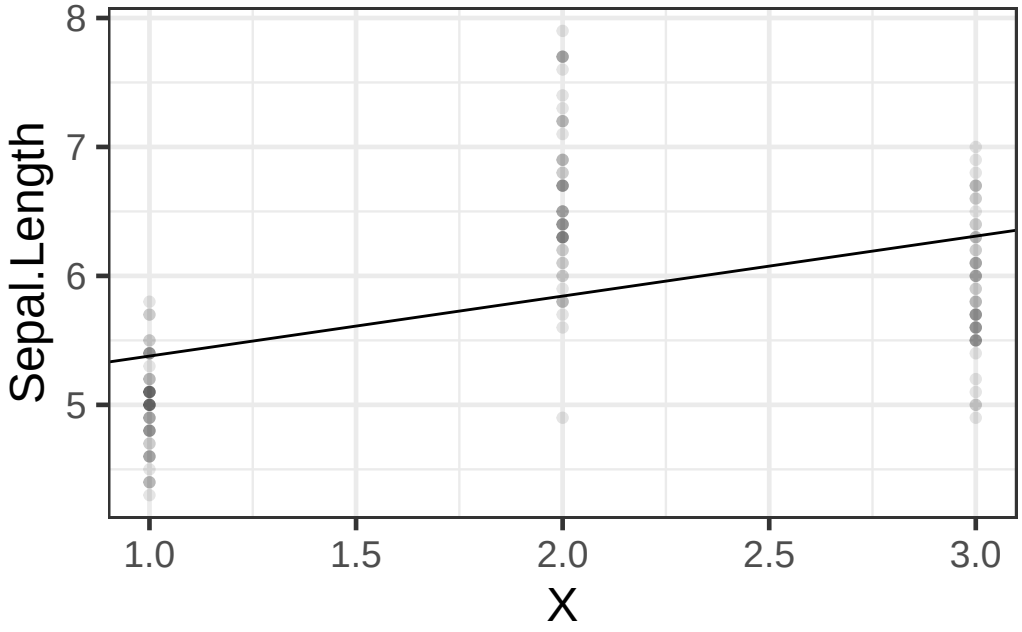
```
iris_lm1 = lm(Sepal.Length ~ X, data = iris)
iris_lm1 |> parameters() |> print_md()
```

Parameter	Coefficient	SE	95% CI	t(148)	p
(Intercept)	4.91	0.16	(4.60, 5.23)	30.83	$< .001$
X	0.47	0.07	(0.32, 0.61)	6.30	$< .001$

0.13.5 Let's see how that model looks:

```
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)
```



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

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

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

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

0.13.7 Re-parametrize with no intercept

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

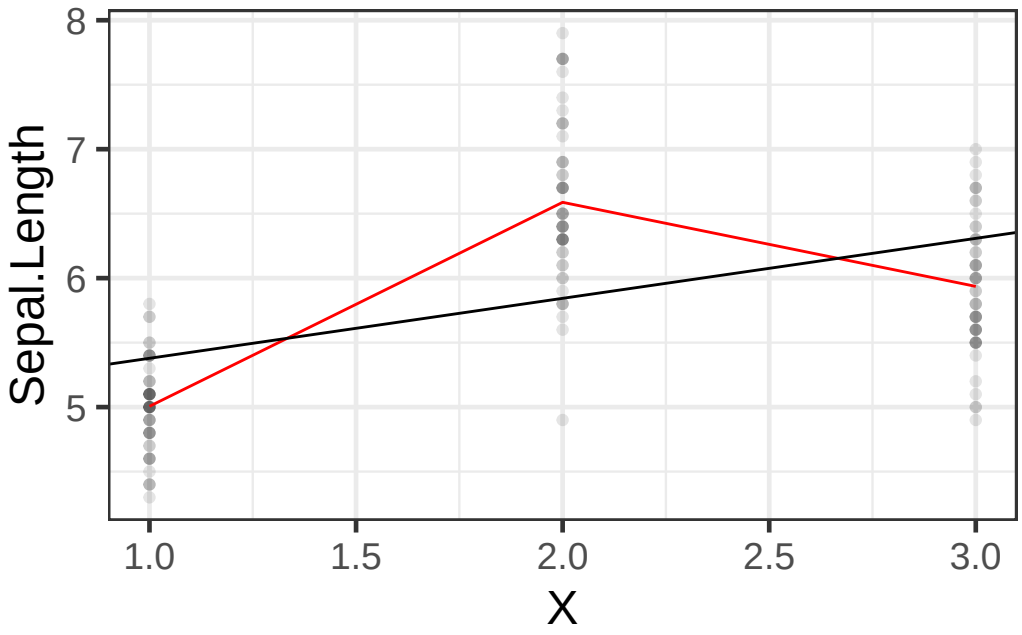
```
iris.lm2b = lm(Sepal.Length ~ Species - 1, data = iris)
iris.lm2b |> parameters() |> print_md()
```

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

0.13.8 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)
```



0.13.9 Let's see how R did that:

```
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
```

This is called a "corner point parametrization".

```
formula(iris.lm2b)
```

```
Sepal.Length ~ Species - 1
```

```
model.matrix(iris.lm2b) |> 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
```

This can be called a "group point parametrization".

There are more options; see Dobson & Barnett §6.4.1.

0.14 Logistic Regression

Acknowledgements

This content is adapted from:

- Dobson & Barnett 2018, "An Introduction to Generalized Linear Models", 4th edition, Chapter 7
- Vittinghoff et al 2012, "Regression Methods in Biostatistics", 2nd edition, Chapter 5
- https://dmrocke.ucdavis.edu/Class/EPI204-Spring-2021/EPI204-Spring-2021.html

Configuring R

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(dplyr) # manipulate data
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(conflicted) # check for conflicting function definitions
```

Here are some R settings I use in this document:

```
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
options('digits' = 4)
```

0.15 Risk Estimation and Prediction

Logistic regression is a method for estimating and predicting the probability of a *binary outcome variable* using one or more *predictors* (a.k.a. *covariates*).

Definition 0.1 (binary variable)**.** A **binary variable** is a random variable which has only two possible values (such as diseased/healthy) in its range.

Exercise 0.1. Recall: what kind of outcome is linear regression used for?

Solution. Linear regression is typically used for continuous outcomes.

You have already seen methods for predicting binary outcomes using one predictor that is also binary (such as exposure/non-exposure).

We will first review one-predictor analyses, with a special focus on risk ratios and odds ratios, which are important concepts for interpreting logistic regression.

Example 0.1 (Oral Contraceptive Use and Heart Attack)**.**

- Research question: does oral contraceptive (OC) use affect the risk of myocardial infarction (MI; a.k.a. heart attack)?

This was an issue when oral contraceptives were first developed, because the original formulations used higher concentrations of hormones. Modern OCs don't have this issue.

Table 15 contains simulated data for an imaginary follow-up (a.k.a. *prospective*) study in which two groups are identified, one using OCs and another not using OCs, and both groups are tracked for three years to determine how many in each groups have MIs.

```r
library(dplyr)
oc_mi =
  tribble(
    ~OC, ~MI, ~Total,
    "OC use", 13, 5000,
    "No OC use", 7, 10000
  ) |>
  mutate(`No MI` = Total - MI) |>
  relocate(`No MI`, .after = MI)

totals =
  oc_mi |>
  summarize(across(c(MI, `No MI`, Total), sum)) |>
  mutate(OC = "Total")

oc_mi = bind_rows(oc_mi, totals)

pander(oc_mi)
```

Table 15: Simulated data from study of oral contraceptive use and heart attack risk

OC	MI	No MI	Total
OC use	13	4987	5000
No OC use	7	9993	10000
Total	20	14980	15000

Exercise 0.2. Review: estimate the probabilities of MI for OC users and non-OC users in Example 0.1.

Solution.

$$\hat{p}(MI|OC) = \frac{13}{5000} = 0.0026$$

$$\hat{p}(MI|\neg OC) = \frac{7}{10000} = 7 \times 10^{-4}$$

Two meanings of "controls"

Depending on context, "controls" can mean either individuals who don't experience an *exposure* of interest, or individuals who don't experience an *outcome* of interest.

Definition 0.2 (cases and controls in retrospective studies)**.** In *retrospective studies*, participants who experience the outcome of interest are called **cases**, while participants who don't experience that outcome are called **controls**.

Definition 0.3 (treatment groups and control groups in prospective studies)**.** In *prospective studies*, the group of participants who experience the treatment or exposure of interest is called the **treatment group**, while the participants who receive the baseline or comparison treatment (for example, clinical trial participants who receive a placebo or a standard-of-care treatment rather than an experimental treatment) are called **controls**.

0.15.1 Comparing probabilities

0.15.1.1 Risk differences

The simplest comparison of two probabilities, π_1, and π_2, is the difference of their values:

Definition 0.4 (Risk difference)**.** The **risk difference** of two probabilities, π_1, and π_2, is the difference of their values:

$$\delta(\pi_1, \pi_2) \stackrel{\text{def}}{=} \pi_1 - \pi_2$$

Example 0.2. In Example 0.1, the estimated risk difference in MI risk between OC users and OC non-users is:

$$\begin{aligned}\hat{\delta}(\pi(OC), \pi(\neg OC)) &= \delta(\hat{\pi}(OC), \hat{\pi}(\neg OC)) \\ &= \hat{\pi}(OC) - \hat{\pi}(\neg OC) \\ &= 0.0026 - 7 \times 10^{-4} \\ &= 0.0019\end{aligned}$$

0.15.1.2 Risk ratios

Definition 0.5 (Relative risk ratios)**.** The **relative risk** of two probabilities π_1 and π_2, also called the **risk ratio**, is the ratio of those probabilities:

$$\rho(\pi_1, \pi_2) = \frac{\pi_1}{\pi_2}$$

Example 0.3. Above, we estimated that:

$$\widehat{p}(MI|OC) = 0.0026$$

$$\widehat{p}(MI|\neg OC) = 7 \times 10^{-4}$$

So we might estimate that the *relative risk* of MI for OC versus non-OC is:

$$\begin{aligned}\widehat{\rho}(OC, \neg OC) &= \frac{\widehat{p}(MI|OC)}{\widehat{p}(MI|\neg OC)} \\ &= \frac{0.0026}{7 \times 10^{-4}} \\ &= 3.7143\end{aligned}$$

Both risk differences and risk ratios are defined by two probabilities, plus a choice of which probability is the **baseline** or **reference** probability (i.e., which probability is the subtrahend of the risk difference or the denominator of the risk ratio). To switch which one is the reference probability, invert the ratio and multiply the difference by -1.

Example 0.4. Above, we estimated that the risk ratio of OC versus non-OC is:

$$\rho(OC, \neg OC) = 3.7143$$

In comparison, the risk ratio for non-OC versus OC is:

$$\begin{aligned}\rho(\neg OC, OC) &= \frac{\widehat{p}(MI|\neg OC)}{\widehat{p}(MI|OC)} \\ &= \frac{7 \times 10^{-4}}{0.0026} \\ &= 0.2692 \\ &= \frac{1}{\rho(OC, \neg OC)}\end{aligned}$$

0.15.2 Odds and probabilities

In logistic regression, we will make use of a transformation (rescaling) of probability, called *odds*.

Definition 0.6 (Odds)**.** The **odds** of an outcome, denoted ω ("omega"), is the probability that the outcome occurs, divided by the probability that it doesn't occur.

That is, if the probability of an outcome is π, then the corresponding odds of that outcome is

$$\omega(\pi) \stackrel{\text{def}}{=} \frac{\pi}{1-\pi}$$

This function, which transforms probabilities into odds, is called the **odds function** (see Figure 7).

```
odds = function(pi) pi/(1-pi)
library(ggplot2)
odds_plot = ggplot() +
  geom_function(fun = odds, aes(col = "odds function")) +
  xlim(0, .5) +
  xlab("Probability") +
  ylab("Odds") +
  geom_abline(aes(intercept = 0, slope = 1, col = "y=x")) +
  theme_bw()
print(odds_plot)
```

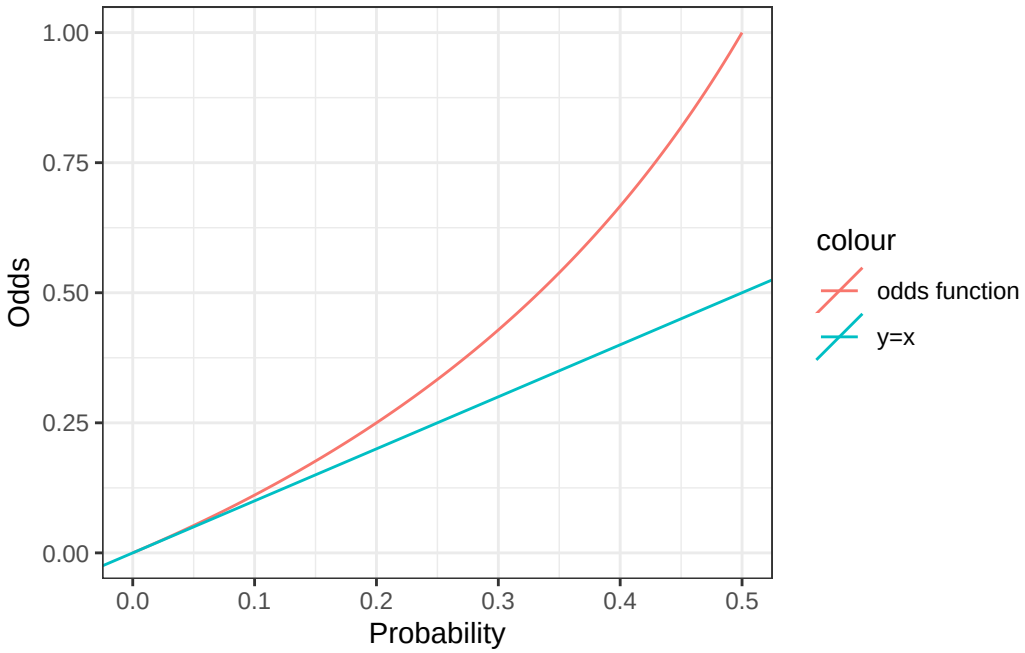


Figure 7: Odds versus probability

Example 0.5 (Calculating odds)**.** In Exercise 0.2, we estimated that the probability of MI, given OC use, is $\pi(OC) \stackrel{\text{def}}{=} \Pr(MI|OC) = 0.0026$. If this estimate is correct, then the odds of MI, given OC use, is:

$$\begin{aligned}\omega(OC) &\stackrel{\text{def}}{=} \frac{\Pr(MI|OC)}{\Pr(\neg MI|OC)} \\ &= \frac{\Pr(MI|OC)}{1-\Pr(MI|OC)} \\ &= \frac{\pi(OC)}{1-\pi(OC)} \\ &= \frac{0.0026}{1-0.0026} \\ &= 0.0026\end{aligned}$$

Exercise 0.3 (Calculating odds)**.** Estimate the odds of MI, for non-OC users.

Solution.

$$\omega_(\neg OC) = 7.0049 \times 10^{-4}$$

0.15.2.1 A shortcut for calculating odds

The usual estimate for a probability of an event is "# events/# observations". We often denote # events as x and # observations as n. So:

$$\hat{\pi} = \frac{x}{n}$$

Thus, the usual estimate for the probability of a nonevent is:

$$\begin{aligned} 1 - \hat{\pi} &= 1 - \frac{x}{n} \\ &= \frac{n}{n} - \frac{x}{n} \\ &= \frac{n - x}{n} \end{aligned}$$

Thus, the estimated odds is:

$$\begin{aligned} \frac{\hat{\pi}}{1 - \hat{\pi}} &= \frac{\left(\frac{x}{n}\right)}{\left(\frac{n-x}{n}\right)} \\ &= \frac{x}{n - x} \end{aligned}$$

That is, odds can be calculated directly as "# events" divided by "# nonevents" (without needing to calculate $\hat{\pi}$ and $1 - \hat{\pi}$ first).

Example 0.6 (calculating odds using the shortcut)**.** In Example 0.5, we calculated

$$\omega(OC) = 0.0026$$

Let's recalculate this result using our shortcut:

$$\begin{aligned} \omega(OC) &= \frac{13}{5000 - 13} \\ &= 0.0026 \end{aligned}$$

Same answer!

0.15.2.2 Odds of rare events

For rare events (small π), odds and probabilities are nearly equal, because $1 - \pi \approx 1$ (see Figure 7).

For example, in Example 0.5, the probability and odds differ by 6.7776×10^{-6}.

Exercise 0.4. What odds value corresponds to the probability $\pi = 0.2$, and what is the numerical difference between these two values?

Solution.

$$\omega = \frac{\pi}{1-\pi} = \frac{.2}{.8} = .25$$

0.15.3 The inverse odds function

Definition 0.7 (inverse odds function)**.** The **inverse odds function**,

$$\pi(\omega) \stackrel{\text{def}}{=} \frac{\omega}{1+\omega}$$

converts odds into their corresponding probabilities (Figure 8).

The inverse-odds function takes an odds as input and produces a probability as output. Its domain of inputs is $[0, \infty)$ and its range of outputs is $[0, 1]$.

```
odds_inv = function(omega) omega / (1 + omega)
ggplot() +
  geom_function(fun = odds_inv, aes(col = "inverse-odds")) +
  xlab("Odds") +
  ylab("Probability") +
  xlim(0,5) +
  ylim(0,1) +
  geom_abline(aes(intercept = 0, slope = 1, col = "x=y"))
```

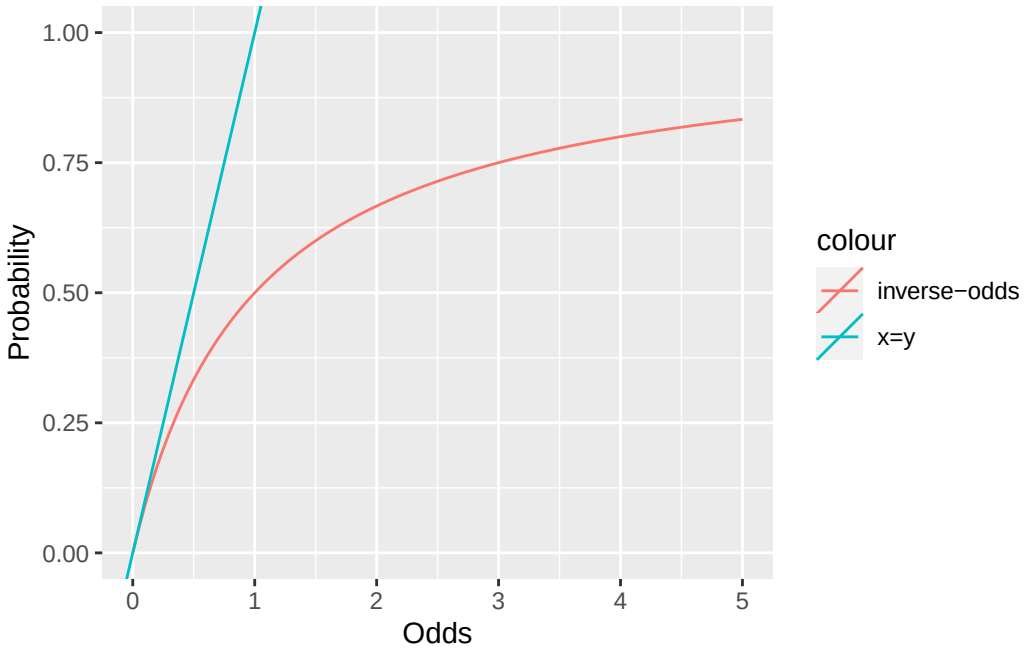


Figure 8: The inverse odds function, $\pi(\omega)$

! Important

An equivalent expression for the inverse odds function is

$$\pi(\omega) = (1 - \omega^{-1})^{-1}$$

.

Exercise 0.5. Prove that the expression above is equivalent to Definition 0.7.

Exercise 0.6. What probability corresponds to an odds of $\omega = 1$, and what is the numerical difference between these two values?

Solution.

$$\pi(1) = \frac{1}{1+1} = \frac{1}{2} = .5$$
$$1 - \pi(1) = 1 - .5 = .5$$

0.15.4 Odds ratios

Now that we have defined odds, we can introduce another way of comparing event probabilities: odds ratios.

Definition 0.8 (Odds ratio)**.** The **odds ratio** for two odds ω_1, ω_2 is their ratio:

$$\theta(\omega_1, \omega_2) = \frac{\omega_1}{\omega_2}$$

Example 0.7 (Calculating odds ratios)**.** In Example 0.1, the odds ratio for OC users versus OC-non-users is:

$$\begin{aligned}\theta(\omega(OC), \omega(\neg OC)) &= \frac{\omega(OC)}{\omega(\neg OC)} \\ &= \frac{0.0026}{7 \times 10^{-4}} \\ &= 3.7143\end{aligned}$$

When the outcome is rare (i.e., its probability is small) for both groups being compared in an odds ratio, the odds of the outcome will be similar to the probability of the outcome, and thus the risk ratio will be similar to the odds ratio.

For example, in Example 0.1, the outcome is rare for both OC and non-OC participants, so the odds for both groups are similar to the corresponding probabilities, and the odds ratio is similar the risk ratio.

0.15.4.1 A shortcut for calculating odds ratio estimates

The general form of a two-by-two table is shown in Table 16.

Table 16: A generic 2x2 table

	Event	Non-Event	Total
Exposed	a	b	a+b
Non-exposed	c	d	c+d
Total	a+c	b+d	a+b+c+d

From this table, we have:

- $\hat{\pi}(Event|Exposed) = a/(a+b)$

- $\hat{\pi}(\neg Event|Exposed) = b/(a+b)$
- $\hat{\omega}(Event|Exposed) = \frac{\left(\frac{a}{a+b}\right)}{\left(\frac{b}{a+b}\right)} = \frac{a}{b}$
- $\hat{\omega}(Event|\neg Exposed) = \frac{c}{d}$ (see Exercise 0.7)
- $\theta(Exposed, \neg Exposed) = \frac{\frac{a}{b}}{\frac{c}{d}} = \frac{ad}{bc}$

Exercise 0.7. Given Table 16, show that $\hat{\omega}(Event|\neg Exposed) = \frac{c}{d}$.

0.15.4.2 Properties of odds ratios

Odds ratios have a special property: we can swap a covariate with the outcome, and the odds ratio remains the same.

Example 0.8. In Example 0.1, we have:

$$\begin{aligned}
\theta(MI; OC) &\stackrel{\text{def}}{=} \frac{\omega(MI|OC)}{\omega(MI|\neg OC)} \\
&\stackrel{\text{def}}{=} \frac{\left(\frac{\Pr(MI|OC)}{\Pr(\neg MI|OC)}\right)}{\left(\frac{\Pr(MI|\neg OC)}{\Pr(\neg MI|\neg OC)}\right)} \\
&= \frac{\left(\frac{\Pr(MI,OC)}{\Pr(\neg MI,OC)}\right)}{\left(\frac{\Pr(MI,\neg OC)}{\Pr(\neg MI,\neg OC)}\right)} \\
&= \left(\frac{\Pr(MI,OC)}{\Pr(\neg MI,OC)}\right)\left(\frac{\Pr(\neg MI,\neg OC)}{\Pr(MI,\neg OC)}\right) \\
&= \left(\frac{\Pr(MI,OC)}{\Pr(MI,\neg OC)}\right)\left(\frac{\Pr(\neg MI,\neg OC)}{\Pr(\neg MI,OC)}\right) \\
&= \left(\frac{\Pr(OC,MI)}{\Pr(\neg OC,MI)}\right)\left(\frac{\Pr(\neg OC,\neg MI)}{\Pr(OC,\neg MI)}\right) \\
&= \left(\frac{\Pr(OC|MI)}{\Pr(\neg OC|MI)}\right)\left(\frac{\Pr(\neg OC|\neg MI)}{\Pr(OC|\neg MI)}\right) \\
&= \frac{\left(\frac{\Pr(OC|MI)}{\Pr(\neg OC|MI)}\right)}{\left(\frac{\Pr(OC|\neg MI)}{\Pr(\neg OC|\neg MI)}\right)} \\
&\stackrel{\text{def}}{=} \frac{\omega(OC|MI)}{\omega(OC|\neg MI)} \\
&\stackrel{\text{def}}{=} \theta(OC; MI)
\end{aligned}$$

Exercise 0.8. For Table 16, show that $\hat{\theta}(Exposed, Unexposed) = \hat{\theta}(Event, \neg Event)$.

0.15.5 Effect of study design

Table 15 simulates a follow-up study in which two populations were followed and the number of MI's was observed. The risks are $P(MI|OC)$ and $P(MI|\neg OC)$ and we can estimate these risks from the data.

But suppose we had a case-control study in which we had 100 women with MI and selected a comparison group of 100 women without MI (matched as groups on age, etc.). Then MI is not random, and we cannot compute P(MI|OC) and we cannot compute the risk ratio. However, the odds ratio however can be computed.

The disease odds ratio is the odds for the disease in the exposed group divided by the odds for the disease in the unexposed group, and we cannot validly compute and use these separate parts.

But we can validly compute and use the exposure odds ratio, which is the odds for exposure in the disease group divided by the odds for exposure in the non-diseased group (because exposure can be treated as random). And these two odds ratios are mathematically equivalent, as we saw in Section 0.15.4.2.

Exercise 0.9. Calculate the odds ratio of MI with respect to OC use, assuming that Table 15 comes from a case-control study. Confirm that the result is the same as in Example 0.7.

Solution.

- $\omega(OC|MI) = P(OC|MI)/(1–P(OC|MI) = \frac{13}{7} = 1.8571$
- $\omega(OC|\neg MI) = P(OC|\neg MI)/(1–P(OC|\neg MI) = \frac{4987}{9993} = 0.499$
- $\theta(OC, MI) = \frac{\omega(OC|MI)}{\omega(OC|\neg MI)} = \frac{13/7}{4987/9993} = 3.7214$

This is the same estimate we calculated in Example 0.7.

0.15.5.1 Cross-Sectional Studies

- If a cross-sectional study is a probability sample of a population (which it rarely is) then we can estimate risks.
- If it is a sample, but not an unbiased probability sample, then we need to treat it in the same way as a case-control study.
- We can validly estimate odds ratios in either case.
- But we can usually not validly estimate risks and risk ratios.

0.16 Introduction to logistic regression

- In Example 0.1, we estimated the risk and the odds of MI for two discrete cases, as to whether of not the individual used oral contraceptives.
- If the predictor is quantitative (dose) or there is more than one predictor, the task becomes more difficult.
- In this case, we will use logistic regression, which is a generalization of the linear regression models you have been using that can account for a binary response instead of a continuous one.

0.16.1 Binary outcomes models - one group, no covariates

$$
\begin{aligned}
p(Y=1) &= \pi \\
p(Y=0) &= 1-\pi \\
p(Y=y) &= \pi^y(1-\pi)^{1-y} \\
\mathbf{y} &= (y_1, ..., y_n) \\
\mathcal{L}(\pi;\mathbf{y}) &= \pi^{\sum y_i}(1-\pi)^{n-\sum y_i} \\
\ell(\pi,\mathbf{y}) &= \left(\sum y_i\right)\log\{\pi\} + \left(n-\sum y_i\right)\log\{1-\pi\} \\
&= \left(\sum y_i\right)(\log\{\pi\} - \log\{1-\pi\}) + n\cdot\log\{1-\pi\} \\
&= \left(\sum y_i\right)\log\left\{\frac{\pi}{1-\pi}\right\} + n\cdot\log\{1-\pi\}
\end{aligned}
$$

0.16.2 Binary outcomes - general

$$
\begin{aligned}
p(Y_i=1) &= \pi_i \\
p(Y_i=0) &= 1-\pi_i \\
p(Y_i=y) &= (\pi_i)^y(1-\pi_i)^{1-y} \\
\mathbf{y} &= (y_1, ..., y_n) \\
\mathcal{L}(\pi;\mathbf{y}) &= \prod_{i=1}^{n}(\pi_i)^{y_i}(1-\pi_i)^{1-y_i} \\
\ell(\pi,\mathbf{y}) &= \sum_{i=1}^{n} y_i\log\{\pi_i\} + (1-y_i)\log\{1-\pi_i\}
\end{aligned}
$$

0.16.3 Modeling π_i as a function of X_i

If there are only a few distinct X_i values, we can model each one separately.

Otherwise, we need regression.

$$\begin{aligned}\pi(x) &\equiv \mathrm{E}(Y = 1|X = x) \\ &= f(x^\top \beta)\end{aligned}$$

Typically, we use $f(\eta) = \text{expit}(\eta)$.

Definition 0.9 (expit function)**.** The **expit** function (Figure 9), also known as the **inverse-logit** or **logistic** function, is:

$$\begin{aligned}\text{expit}(\eta) &\stackrel{\text{def}}{=} \frac{\exp\{\eta\}}{1+\exp\{\eta\}} \\ &= (1+\exp\{-\eta\})^{-1}\end{aligned}$$

```
expit = function(eta) exp(eta)/(1+exp(eta))
library(ggplot2)
expit_plot =
  ggplot() +
  geom_function(fun = expit) +
  xlim(-5, 5) +
  ylim(0,1) +
  ylab(expression(expit(eta))) +
  xlab(expression(eta)) +
  theme_bw()
print(expit_plot)
```

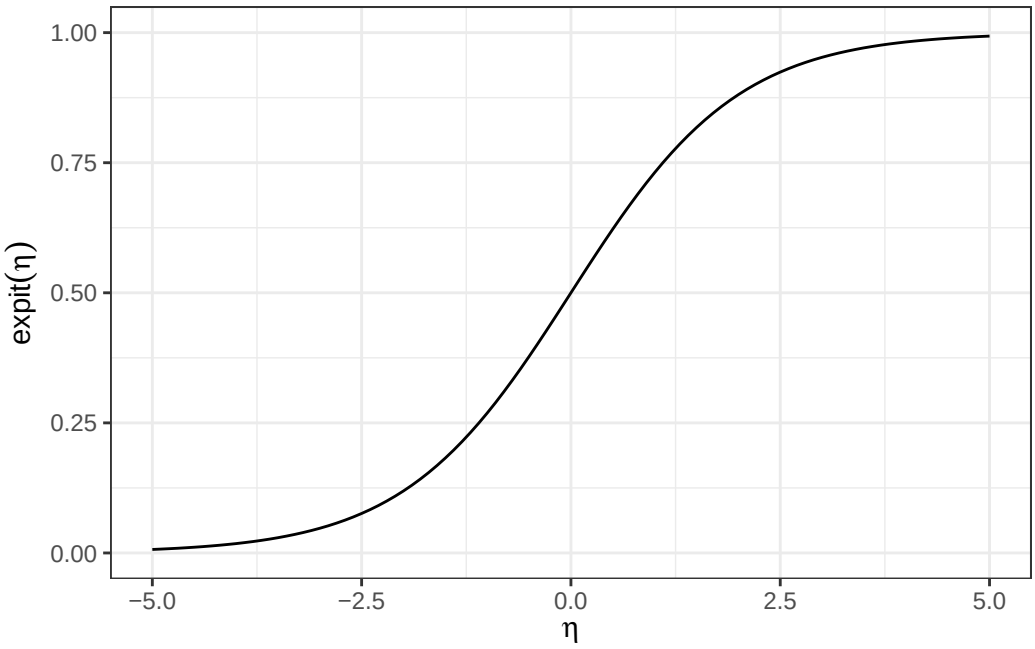


Figure 9: The expit function

Definition 0.10 (logit function)**.** The inverse of the `expit` function is the `logit` function:

$$g(p) = f^{-1}(p) = \text{logit}(p) = \log\left\{\frac{p}{1-p}\right\}$$

```
logit = function(p) log(odds(p))

logit_plot =
  ggplot() +
  geom_function(fun = logit) +
  xlim(.01, .99) +
  ylab("logit(p)") +
  xlab("p") +
  theme_bw()
print(logit_plot)
```

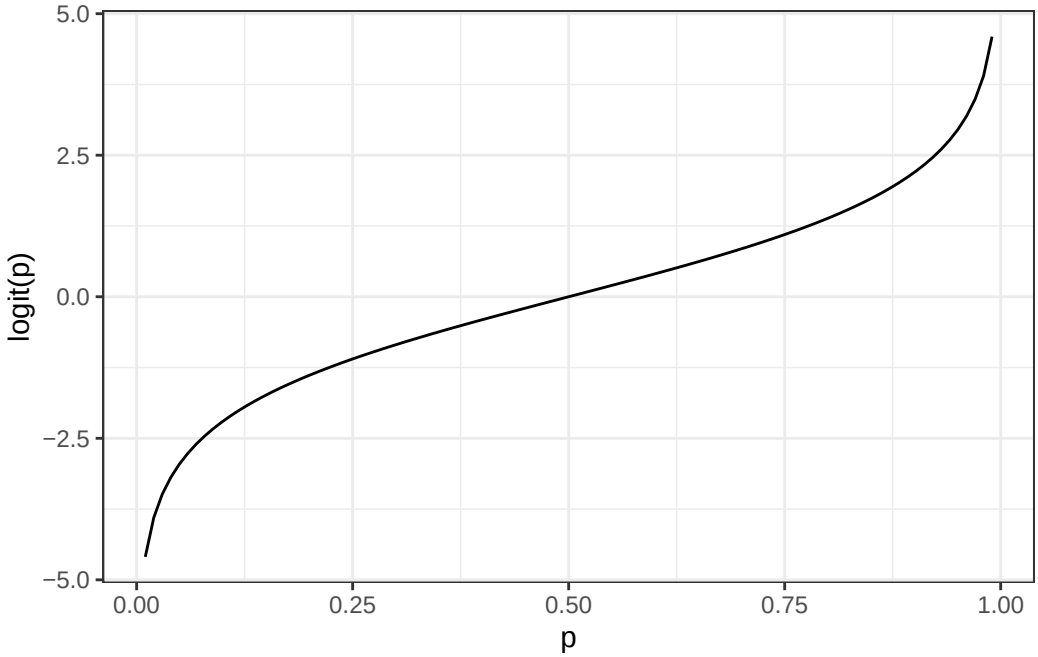


Figure 10: the logit function

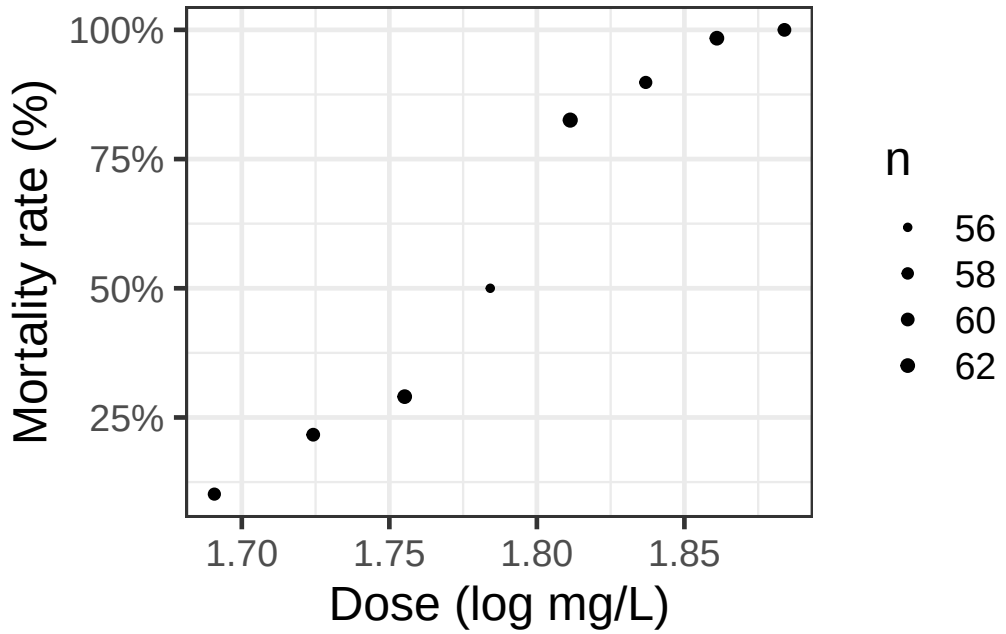
0.16.4 Diagram of expit and logit

$$\overbrace{\Big[\pi \overset{\text{def}}{=} \Pr(Y=1)\Big] \underset{\frac{\omega}{1+\omega}}{\overset{\frac{\pi}{1-\pi}}{\rightleftarrows}} \Big[\omega \overset{\text{def}}{=} \text{odds}(Y=1)\Big] \underset{\exp\{\eta\}}{\overset{\log\{\omega\}}{\rightleftarrows}} \Big[\eta \overset{\text{def}}{=} \text{log-odds}(Y=1)\Big]}^{\text{logit}(\pi)}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{expit}(\eta)}$$

0.16.5 Meet the beetles

```
library(glmx)

data(BeetleMortality, package = "glmx")
beetles = BeetleMortality |>
  mutate(
    pct = died/n,
    survived = n - died
  )
```

```r
plot1 =
  beetles |>
  ggplot(aes(x = dose, y = pct)) +
  geom_point(aes(size = n)) +
  xlab("Dose (log mg/L)") +
  ylab("Mortality rate (%)") +
  scale_y_continuous(labels = scales::percent) +
  scale_size(range = c(1,2)) +
  theme_bw(base_size = 18)

print(plot1)
```

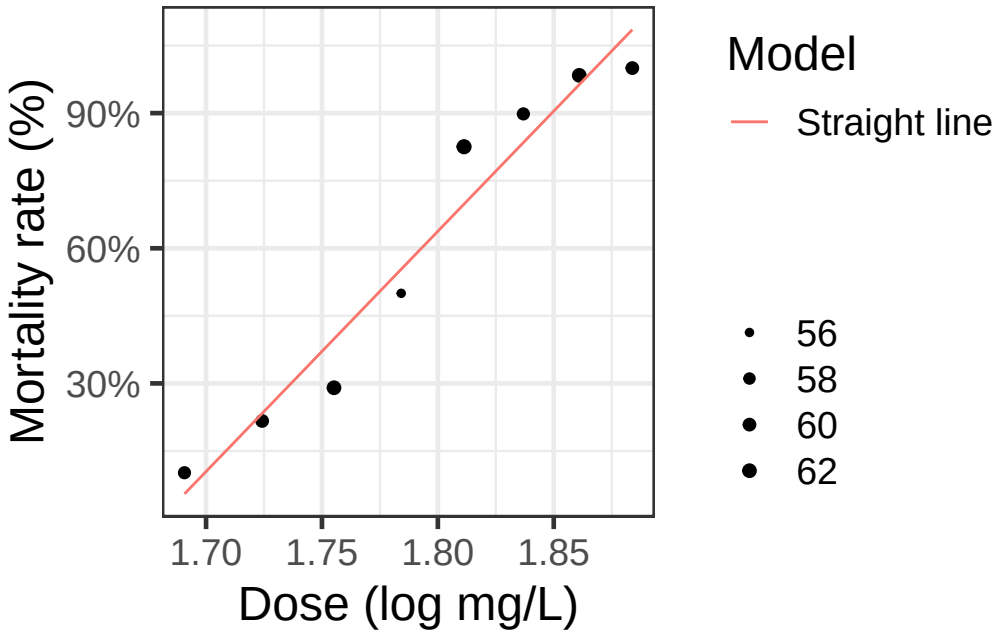


Mortality rates of adult flour beetles after five hours' exposure to gaseous carbon disulphide (Bliss 1935)

0.16.6 Why don't we use linear regression?

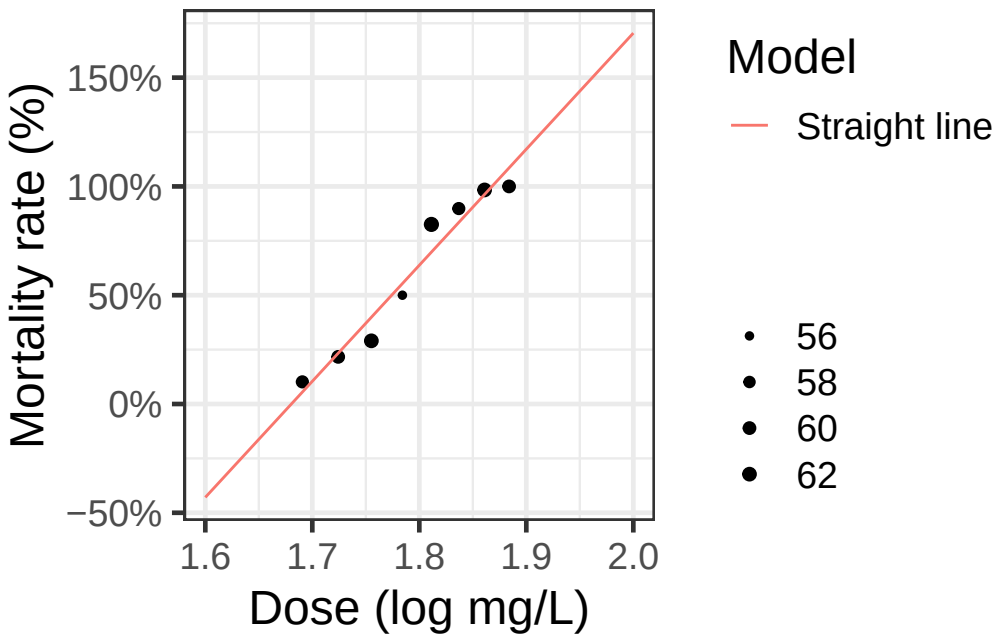
```
beetles_glm_grouped = beetles |>
  glm(formula = cbind(died, survived) ~ dose, family = "binomial")

lm1 =
  beetles  |>
  reframe(
    .by = everything(),
    outcome = c(
      rep(1, times = died),
      rep(0, times = survived))
  ) |>
  lm(
    formula = outcome ~ dose,
    data = _)

lm2 =
  beetles  |>
  reframe(
    .by = everything(),
    outcome = c(
      rep(1, times = died),
      rep(0, times = survived))
  ) |>
  lm(
    formula = outcome ~ log(dose),
    data = _)

range1 = range(beetles$dose) + c(-.2, .2)
f = function(x) predict(beetles_glm_grouped, newdata = data.frame(dose = x), type = "respo
f.linear = function(x) predict(lm1, newdata = data.frame(dose = x))
f.linearlog = function(x) predict(lm2, newdata = data.frame(dose = x))

plot2 =
  plot1 +
  geom_function(
    fun = f.linear,
    aes(col = "Straight line")) +
  labs(colour="Model", size = "")
```

```
plot2 |> print()
```



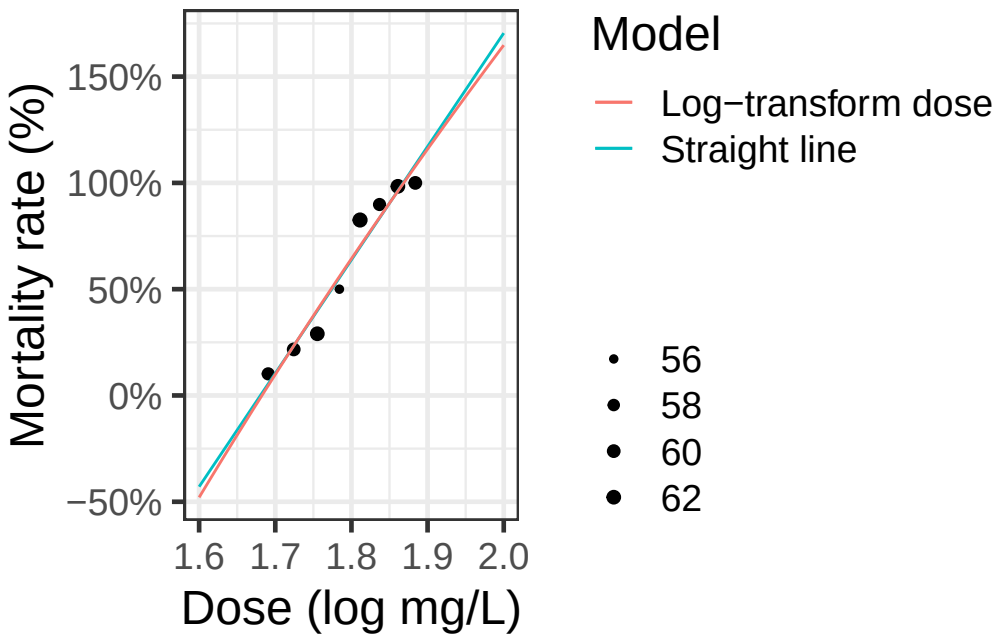
0.16.7 Zoom out

```
(plot2 + expand_limits(x = c(1.6, 2))) |> print()
```



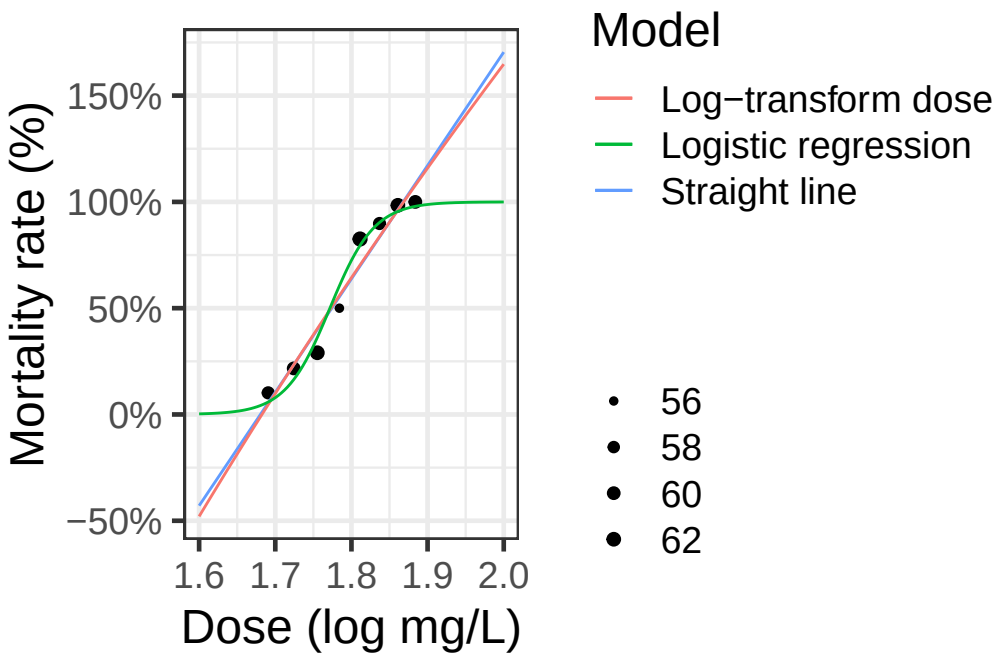
0.16.8 log transformation of dose?

```
plot3 = plot2 +
  expand_limits(x = c(1.6, 2)) +
  geom_function(fun = f.linearlog, aes(col = "Log-transform dose"))
(plot3 + expand_limits(x = c(1.6, 2))) |> print()
```



0.16.9 Logistic regression

```
plot4 = plot3 + geom_function(fun = f, aes(col = "Logistic regression"))
plot4 |> print()
```



0.16.10 Three parts to regression models

- What distribution does the outcome have for a specific subpopulation defined by covariates? (outcome model)
- How does the combination of covariates relate to the mean? (link function)
- How do the covariates combine? (linear predictor, interactions)

0.16.11 Logistic regression in R

```
beetles_glm_grouped =
  beetles |>
  glm(
    formula = cbind(died, survived) ~ dose,
    family = "binomial")

beetles_glm_grouped |>
  parameters() |>
  print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
(Intercept)	-60.72	5.18	(-71.44, -51.08)	-11.72	$< .001$
dose	34.27	2.91	(28.85, 40.30)	11.77	$< .001$

Fitted values:

```
fitted.values(beetles_glm_grouped)
```

```
     1      2      3      4      5      6      7      8
0.0586 0.1640 0.3621 0.6053 0.7952 0.9032 0.9552 0.9790
```

```
predict(beetles_glm_grouped, type = "response")
```

```
     1      2      3      4      5      6      7      8
0.0586 0.1640 0.3621 0.6053 0.7952 0.9032 0.9552 0.9790
```

```
predict(beetles_glm_grouped, type = "link")
```

```
      1       2       3       4       5       6       7       8
-2.7766 -1.6286 -0.5662  0.4277  1.3564  2.2337  3.0596  3.8444
```

```
fit_y = beetles$n * fitted.values(beetles_glm_grouped)
```

0.16.12 Individual observations

```
beetles_long =
  beetles  |>
  reframe(
    .by = everything(),
    outcome = c(
      rep(1, times = died),
      rep(0, times = survived))
  )
beetles_long |> tibble() |> print()
```

```
# A tibble: 481 x 6
    dose  died     n   pct survived outcome
   <dbl> <int> <int> <dbl>    <int>   <dbl>
 1  1.69     6    59 0.102       53       1
 2  1.69     6    59 0.102       53       1
 3  1.69     6    59 0.102       53       1
 4  1.69     6    59 0.102       53       1
 5  1.69     6    59 0.102       53       1
 6  1.69     6    59 0.102       53       1
 7  1.69     6    59 0.102       53       0
 8  1.69     6    59 0.102       53       0
 9  1.69     6    59 0.102       53       0
10  1.69     6    59 0.102       53       0
# i 471 more rows
```

Here's the model with individual data

```
beetles_glm_ungrouped =
  beetles_long |>
  glm(
    formula = outcome ~ dose,
    family = "binomial")

beetles_glm_ungrouped |> parameters() |> print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
(Intercept)	-60.72	5.18	(-71.44, -51.08)	-11.72	< .001
dose	34.27	2.91	(28.85, 40.30)	11.77	< .001

Here's the previous version again:

```
beetles_glm_grouped |> parameters() |> print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
(Intercept)	-60.72	5.18	(-71.44, -51.08)	-11.72	< .001
dose	34.27	2.91	(28.85, 40.30)	11.77	< .001

They seem the same! But not quite:

```
logLik(beetles_glm_grouped)
```

```
'log Lik.' -18.72 (df=2)
```

```
logLik(beetles_glm_ungrouped)
```

```
'log Lik.' -186.2 (df=2)
```

The difference is due to the binomial coefficient $\binom{n}{x}$ which isn't included in the individual-observations (Bernoulli) version of the model.

0.17 Multiple logistic regression

0.17.1 Coronary heart disease (WCGS) study data

Let's use the data from the following study to explore multiple logistic regression:

0.17.1.1 Summary of study

From Vittinghoff et al 2012:

"The **Western Collaborative Group Study (WCGS)** was a large epidemiological study designed to investigate the association between the"type A" behavior pattern and coronary heart disease (CHD) (Rosenman et al. 1964)."

From Wikipedia, "Type A and Type B personality theory":

"The hypothesis describes Type A individuals as outgoing, ambitious, rigidly organized, highly status-conscious, impatient, anxious, proactive, and concerned with time management….

The hypothesis describes Type B individuals as a contrast to those of Type A. Type B personalities, by definition, are noted to live at lower stress levels. They typically work steadily and may enjoy achievement, although they have a greater tendency to disregard physical or mental stress when they do not achieve."

0.17.1.2 Study design

from `?faraway::wcgs:`

3154 healthy young men aged 39-59 from the San Francisco area were assessed for their personality type. All were free from coronary heart disease at the start of the research. Eight and a half years later change in CHD status was recorded.

Details (from `faraway::wcgs`*)*

The WCGS began in 1960 with 3,524 male volunteers who were employed by 11 California companies. Subjects were 39 to 59 years old and free of heart disease as determined by electrocardiogram. After the initial screening, the study population dropped to 3,154 and the number of companies to 10 because of various exclusions. The cohort comprised both blue- and white-collar employees.

At baseline the following information was collected:

- socio-demographic including:
- age
- education
- marital status
- income
- occupation
- physical and physiological including:
- height
- weight
- blood pressure
- electrocardiogram
- corneal arcus;
- biochemical including:
- cholesterol and lipoprotein fractions;
- medical and family history and use of medications;
- behavioral data including
- Type A interview,
- smoking,
- exercise
- alcohol use.

Later surveys added data on:

- anthropometry
- triglycerides
- Jenkins Activity Survey
- caffeine use

Average follow-up continued for 8.5 years with repeat examinations.

Reference: Coronary Heart Disease in the Western Collaborative Group Study Final Follow-up Experience of 8 1/2 Years Ray H. Rosenman, MD; Richard J. Brand, PhD; C. David Jenkins, PhD; Meyer Friedman, MD; Reuben Straus, MD; Moses Wurm, MD JAMA. 1975;233(8):872-877. doi:10.1001/jama.1975.03260080034016.

0.17.2 Load the data

Here, I load the data:

```
## load the data directly from a UCSF website:
# library(haven)
# url = paste0(
#     # I'm breaking up the url into two chunks for readability
#     "https://regression.ucsf.edu/sites/g/files/",
#     "tkssra6706/f/wysiwyg/home/data/wcgs.dta")
# wcgs = haven::read_dta(url)


# I presaved the data in my project's `data` folder
library(here) # provides the `here()` function
library(fs) # provides the `path()` function
here::here() |>
  fs::path('data/wcgs.rda') |>
  load()
```

0.17.3 Now let's do some data cleaning

```
library(arsenal) # provides `set_labels()`

wcgs = wcgs |>
  mutate(
    age = age |>
      arsenal::set_labels("Age (years)"),

    arcus =
      arcus |>
      as.logical() |>
      arsenal::set_labels("Arcus Senilis"),
```

```
    time169 =
      time169 |>
      as.numeric() |>
      arsenal::set_labels("Observation (follow up) time (days)"),

    dibpat =
      dibpat |>
      as_factor() |>
      relevel(ref = "Type A") |>
      arsenal::set_labels("Behavioral Pattern"),

    typchd69 = typchd69 |>
      labelled(
        label = "Type of CHD Event",
        labels =
          c(
            "None" = 0,
            "infdeath" = 1,
            "silent" = 2,
            "angina" = 3)),

    # turn stata-style labelled variables in to R-style factors:
    across(
      where(is.labelled),
      haven::as_factor)
  )
```

0.17.4 What's in the data

Here's a table of the data:

```
wcgs |>
  select(-c(id, uni, t1)) |>
  tableby(chd69 ~ ., data = _) |>
  summary(
    pfootnote = TRUE,
    title =
      "Baseline characteristics by CHD status at end of follow-up")
```

Table 20: Baseline characteristics by CHD status at end of follow-up

	No (N=2897)	Yes (N=257)	Total (N=3154)	p value
Age (years)				< 0.001[1]
Mean (SD)	46.082 (5.457)	48.490 (5.801)	46.279 (5.524)	
Range	39.000 - 59.000	39.000 - 59.000	39.000 - 59.000	
Arcus Senilis				< 0.001[2]
N-Miss	0	2	2	
FALSE	2058 (71.0%)	153 (60.0%)	2211 (70.1%)	
TRUE	839 (29.0%)	102 (40.0%)	941 (29.9%)	
Behavioral Pattern				< 0.001[2]
A1	234 (8.1%)	30 (11.7%)	264 (8.4%)	
A2	1177 (40.6%)	148 (57.6%)	1325 (42.0%)	
B3	1155 (39.9%)	61 (23.7%)	1216 (38.6%)	
B4	331 (11.4%)	18 (7.0%)	349 (11.1%)	
Body Mass Index (kg/m2)				< 0.001[1]
Mean (SD)	24.471 (2.561)	25.055 (2.579)	24.518 (2.567)	
Range	11.191 - 37.653	19.225 - 38.947	11.191 - 38.947	
Total Cholesterol				< 0.001[1]
N-Miss	12	0	12	
Mean (SD)	224.261 (42.217)	250.070 (49.396)	226.372 (43.420)	
Range	103.000 - 400.000	155.000 - 645.000	103.000 - 645.000	
Diastolic Blood Pressure				< 0.001[1]
Mean (SD)	81.723 (9.621)	85.315 (10.311)	82.016 (9.727)	
Range	58.000 - 150.000	64.000 - 122.000	58.000 - 150.000	
Behavioral Pattern				< 0.001[2]
Type A	1411 (48.7%)	178 (69.3%)	1589 (50.4%)	
Type B	1486 (51.3%)	79 (30.7%)	1565 (49.6%)	
Height (inches)				0.290[1]
Mean (SD)	69.764 (2.539)	69.938 (2.410)	69.778 (2.529)	

	No (N=2897)	Yes (N=257)	Total (N=3154)	p value
Range	60.000 - 78.000	63.000 - 77.000	60.000 - 78.000	
Ln of Systolic Blood Pressure				< 0.001[1]
Mean (SD)	4.846 (0.110)	4.900 (0.125)	4.850 (0.112)	
Range	4.585 - 5.438	4.605 - 5.298	4.585 - 5.438	
Ln of Weight				< 0.001[1]
Mean (SD)	5.126 (0.123)	5.155 (0.118)	5.128 (0.123)	
Range	4.357 - 5.670	4.868 - 5.768	4.357 - 5.768	
Cigarettes per day				< 0.001[1]
Mean (SD)	11.151 (14.329)	16.665 (15.657)	11.601 (14.518)	
Range	0.000 - 99.000	0.000 - 60.000	0.000 - 99.000	
Systolic Blood Pressure				< 0.001[1]
Mean (SD)	128.034 (14.746)	135.385 (17.473)	128.633 (15.118)	
Range	98.000 - 230.000	100.000 - 200.000	98.000 - 230.000	
Current smoking				< 0.001[2]
No	1554 (53.6%)	98 (38.1%)	1652 (52.4%)	
Yes	1343 (46.4%)	159 (61.9%)	1502 (47.6%)	
Observation (follow up) time (days)				< 0.001[1]
Mean (SD)	2775.158 (562.205)	1654.700 (859.297)	2683.859 (666.524)	
Range	238.000 - 3430.000	18.000 - 3229.000	18.000 - 3430.000	
Type of CHD Event				
None	0 (0.0%)	0 (0.0%)	0 (0.0%)	
infdeath	2897 (100.0%)	0 (0.0%)	2897 (91.9%)	
silent	0 (0.0%)	135 (52.5%)	135 (4.3%)	
angina	0 (0.0%)	71 (27.6%)	71 (2.3%)	
4	0 (0.0%)	51 (19.8%)	51 (1.6%)	
Weight (lbs)				< 0.001[1]
Mean (SD)	169.554 (21.010)	174.463 (21.573)	169.954 (21.096)	

	No (N=2897)	Yes (N=257)	Total (N=3154)	p value
Range	78.000 - 290.000	130.000 - 320.000	78.000 - 320.000	
Weight Category				< 0.001[2]
< 140	217 (7.5%)	15 (5.8%)	232 (7.4%)	
140-170	1440 (49.7%)	98 (38.1%)	1538 (48.8%)	
170-200	1049 (36.2%)	122 (47.5%)	1171 (37.1%)	
> 200	191 (6.6%)	22 (8.6%)	213 (6.8%)	
RECODE of age (Age)				< 0.001[2]
35-40	512 (17.7%)	31 (12.1%)	543 (17.2%)	
41-45	1036 (35.8%)	55 (21.4%)	1091 (34.6%)	
46-50	680 (23.5%)	70 (27.2%)	750 (23.8%)	
51-55	463 (16.0%)	65 (25.3%)	528 (16.7%)	
56-60	206 (7.1%)	36 (14.0%)	242 (7.7%)	

1. Linear Model ANOVA
2. Pearson's Chi-squared test

0.17.5 Data by age and personality type

For now, we will look at the interaction between age and personality type (`dibpat`). To make it easier to visualize the data, we summarize the event rates for each combination of age:

```
chd_grouped_data =
  wcgs |>
  summarize(
    .by = c(age, dibpat),
    n = n(),
    `p(chd)` = mean(chd69 == "Yes") |>
      labelled(label = "CHD Event by 1969"),
    `odds(chd)` = `p(chd)`/(1-`p(chd)`),
    `logit(chd)` = log(`odds(chd)`)
  )

chd_grouped_data
```

```
# A tibble: 42 x 6
```

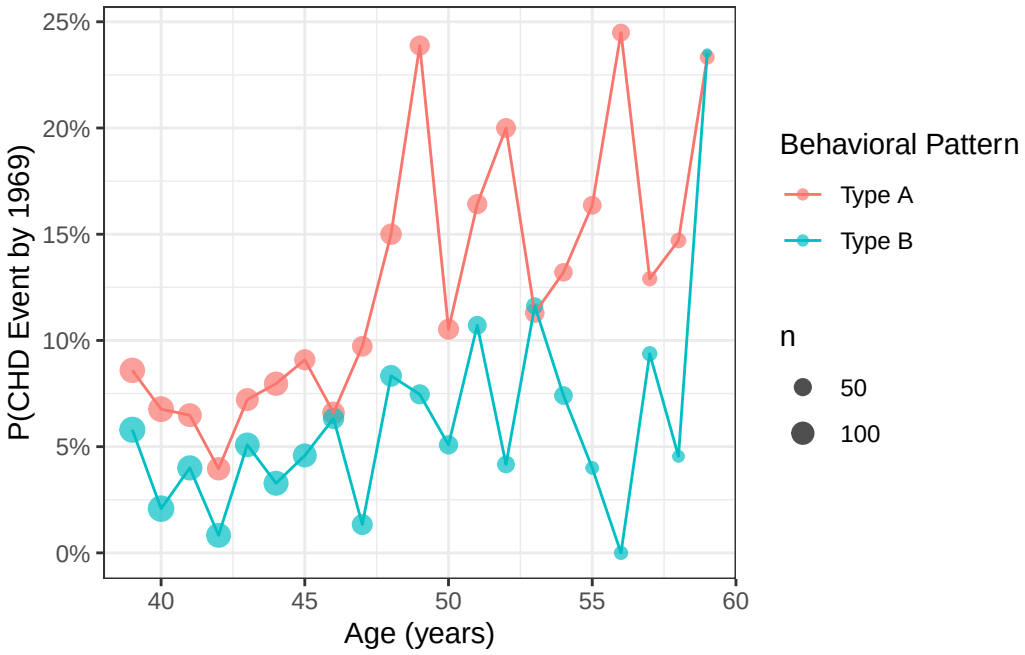
```
     age dibpat     n `p(chd)`  `odds(chd)` `logit(chd)`
   <dbl> <fct>  <int> <dbl+lbl>       <dbl>        <dbl>
 1    50 Type A    76 0.105          0.118        -2.14
 2    51 Type A    67 0.164          0.196        -1.63
 3    59 Type A    30 0.233          0.304        -1.19
 4    44 Type A   113 0.0796         0.0865       -2.45
 5    47 Type A    72 0.0972         0.108        -2.23
 6    40 Type A   133 0.0677         0.0726       -2.62
 7    41 Type A   108 0.0648         0.0693       -2.67
 8    43 Type A    97 0.0722         0.0778       -2.55
 9    54 Type A    53 0.132          0.152        -1.88
10    48 Type A    80 0.15           0.176        -1.73
# i 32 more rows
```

0.17.6 Graphical exploration

0.17.6.1 Probability scale

```
library(ggplot2)
library(ggeasy)
library(scales)
chd_plot_probs =
  chd_grouped_data |>
  ggplot(
    aes(
      x = age,
      y = `p(chd)`,
      col = dibpat)
  ) +
  geom_point(aes(size = n), alpha = .7) +
  scale_size(range = c(1,4)) +
  geom_line() +
  theme_bw() +
  ylab("P(CHD Event by 1969)") +
  scale_y_continuous(labels = scales::label_percent()) +
  ggeasy::easy_labs()
```

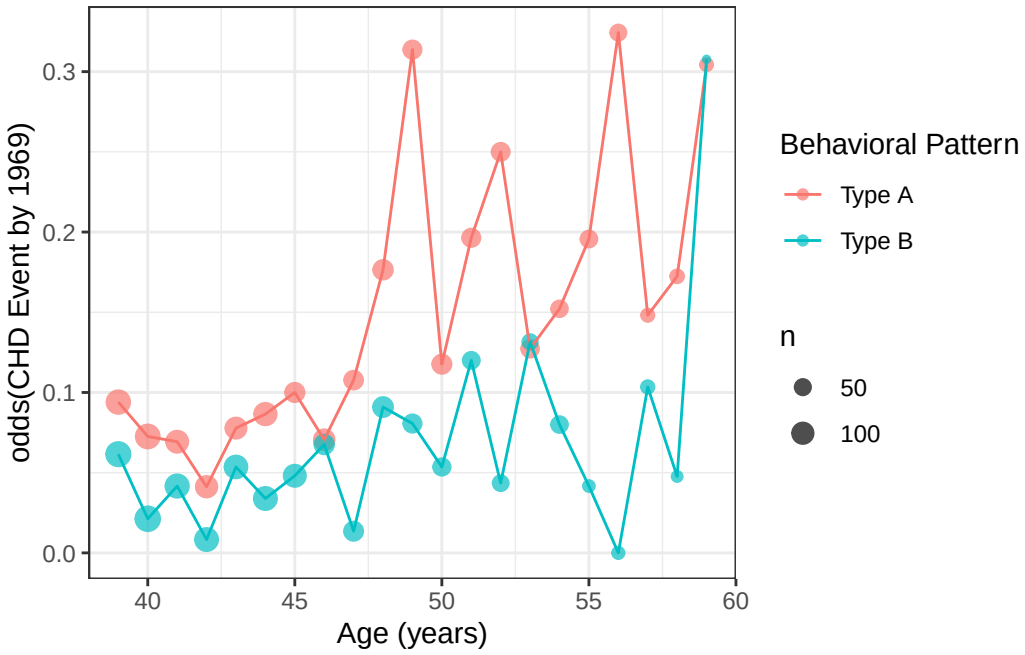
```
print(chd_plot_probs)
```



0.17.6.2 Odds scale

```
chd_plot_odds =
  chd_grouped_data |>
  ggplot(
    aes(
      x = age,
      y = `odds(chd)`,
      col = dibpat)
  ) +
  geom_point(aes(size = n), alpha = .7) +
  scale_size(range = c(1,4)) +
  geom_line() +
  theme_bw() +
  ylab("odds(CHD Event by 1969)") +
  ggeasy::easy_labs()
```

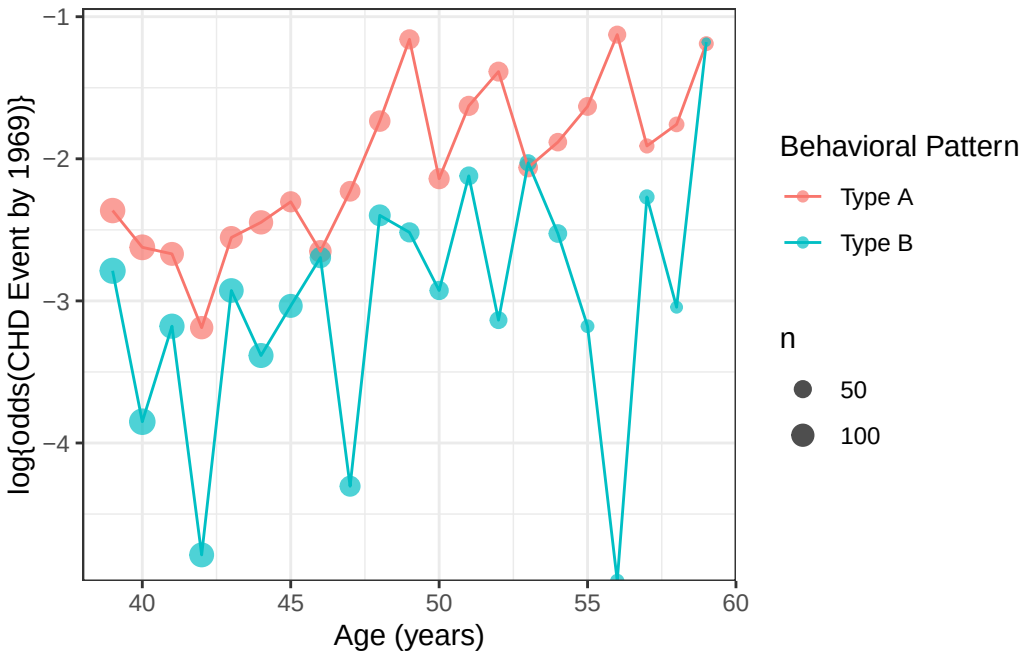
```
print(chd_plot_odds)
```



0.17.6.3 Log-odds (logit) scale

```
chd_plot_logit =
  chd_grouped_data |>
  ggplot(
    aes(
      x = age,
      y = `logit(chd)`,
      col = dibpat)
  ) +
  geom_point(aes(size = n), alpha = .7) +
  scale_size(range = c(1,4)) +
  geom_line() +
  theme_bw() +
  ylab("log{odds(CHD Event by 1969)}") +
  ggeasy::easy_labs()
```

```
print(chd_plot_logit)
```



0.17.7 Logistic regression models for CHD data

Here, we fit stratified models for CHD by personality type.

```
chd_glm_strat = glm(
  "formula" = chd69 == "Yes" ~ dibpat + dibpat:age - 1,
  "data" = wcgs,
  "family" = binomial(link = "logit")
)

chd_glm_strat |> parameters() |> print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
dibpat (Type A)	-5.50	0.67	(-6.83, -4.19)	-8.18	$< .001$
dibpat (Type B)	-5.80	0.98	(-7.73, -3.90)	-5.95	$< .001$
dibpat (Type A) × age	0.07	0.01	(0.05, 0.10)	5.24	$< .001$
dibpat (Type B) × age	0.06	0.02	(0.02, 0.10)	3.01	0.003

We can get the corresponding odds ratios (e^{β}s) by passing `exponentiate = TRUE` to `parameters()`:

```
chd_glm_strat |>
  parameters(exponentiate = TRUE) |>
  print_md()
```

Parameter	Odds Ratio	SE	95% CI	z	p
dibpat (Type A)	4.09e-03	2.75e-03	(1.08e-03, 0.02)	-8.18	< .001
dibpat (Type B)	3.02e-03	2.94e-03	(4.40e-04, 0.02)	-5.95	< .001
dibpat (Type A) × age	1.07	0.01	(1.05, 1.10)	5.24	< .001
dibpat (Type B) × age	1.06	0.02	(1.02, 1.11)	3.01	0.003

0.17.8 Models superimposed on data

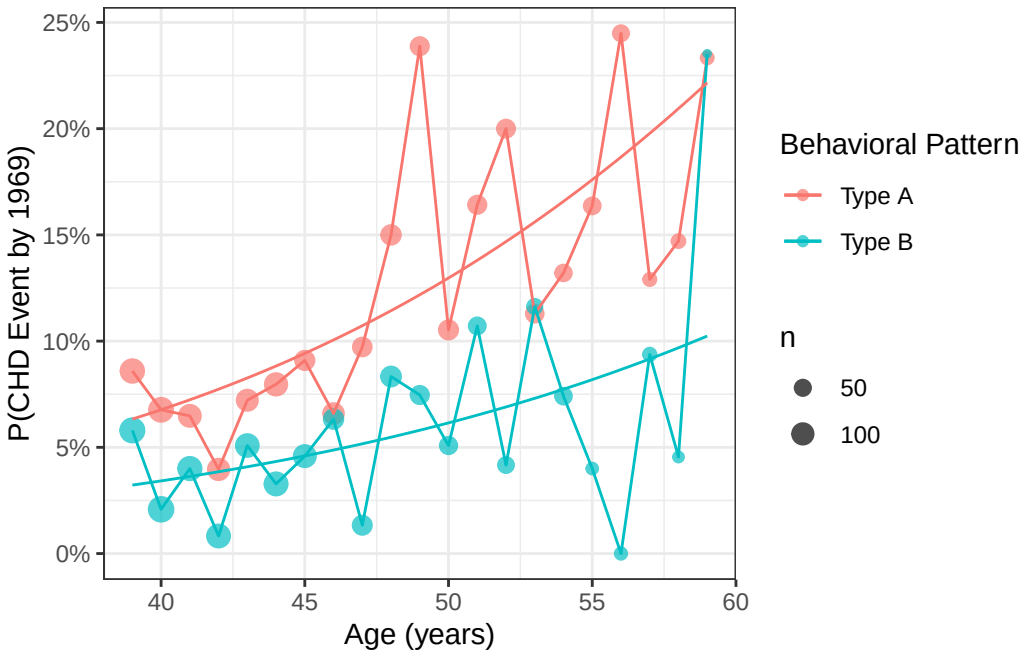
We can graph our fitted models on each scale (probability, odds, log-odds).

0.17.8.1 probability scale

```
curve_type_A = function(x)
{
  chd_glm_strat |> predict(
    type = "response",
    newdata = tibble(age = x, dibpat = "Type A"))
}

curve_type_B = function(x)
{
  chd_glm_strat |> predict(
    type = "response",
    newdata = tibble(age = x, dibpat = "Type B"))
}

chd_plot_probs_2 =
  chd_plot_probs +
  geom_function(
    fun = curve_type_A,
    aes(col = "Type A")
  ) +
  geom_function(
    fun = curve_type_B,
```

```
    aes(col = "Type B")
  )
```

```
print(chd_plot_probs_2)
```

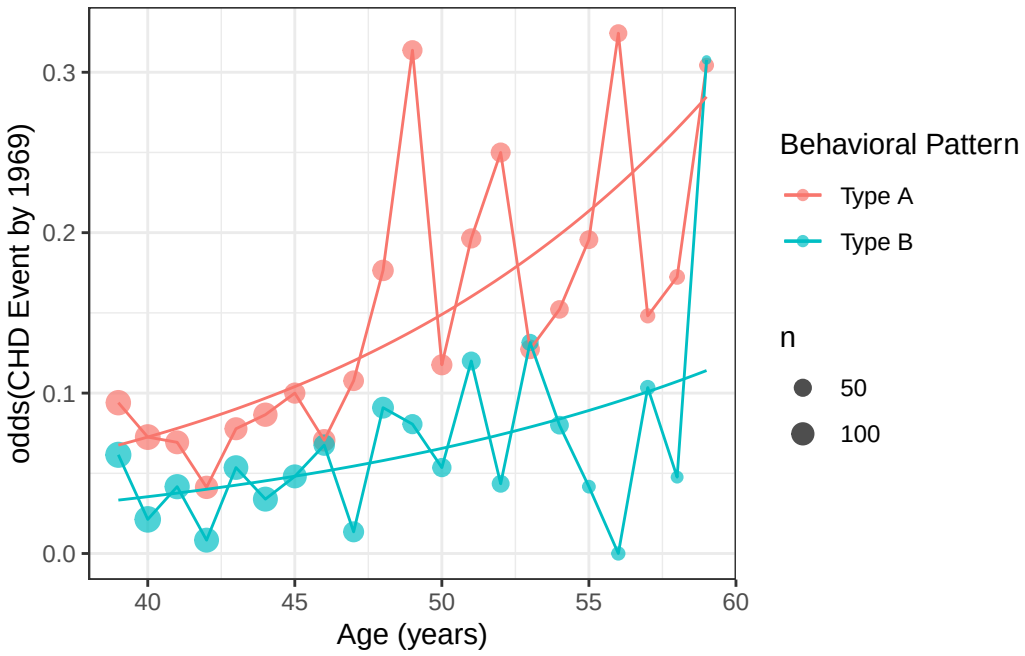


0.17.8.2 odds scale

```
curve_type_A = function(x)
{
  chd_glm_strat |> predict(
    type = "link",
    newdata = tibble(age = x, dibpat = "Type A")) |> exp()
}
curve_type_B = function(x)
{
  chd_glm_strat |> predict(
    type = "link",
    newdata = tibble(age = x, dibpat = "Type B")) |> exp()
}
```

```
chd_plot_odds_2 =
  chd_plot_odds +
  geom_function(
    fun = curve_type_A,
    aes(col = "Type A")
  ) +
  geom_function(
    fun = curve_type_B,
    aes(col = "Type B")
  )
```

```
print(chd_plot_odds_2)
```

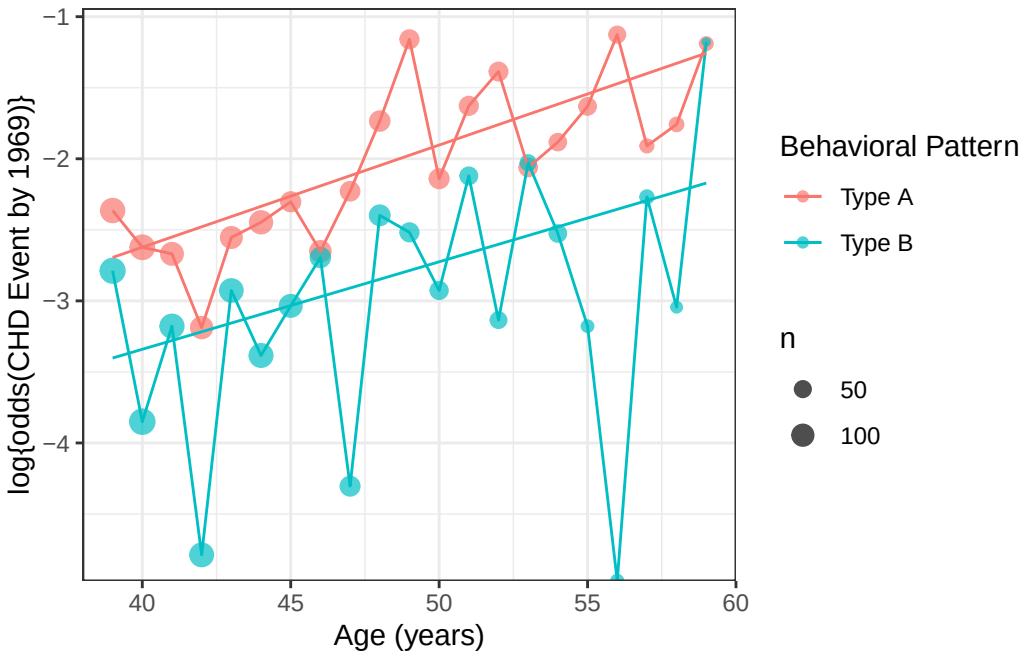


0.17.8.3 log-odds (logit) scale

```
curve_type_A = function(x)
{
  chd_glm_strat |> predict(
    type = "link",
    newdata = tibble(age = x, dibpat = "Type A"))
```

```
}
curve_type_B = function(x)
{
  chd_glm_strat |> predict(
    type = "link",
    newdata = tibble(age = x, dibpat = "Type B"))
}

chd_plot_logit_2 =
  chd_plot_logit +
  geom_function(
    fun = curve_type_A,
    aes(col = "Type A")
  ) +
  geom_function(
    fun = curve_type_B,
    aes(col = "Type B")
  )
```

```
print(chd_plot_logit_2)
```



0.17.9 reference-group and contrast parametrization

We can also use the corner-point parametrization (with reference groups and contrasts):

```
chd_glm_contrasts =
  wcgs |>
  glm(
    "data" = _,
    "formula" = chd69 == "Yes" ~ dibpat*age,
    "family" = binomial(link = "logit")
  )

chd_glm_contrasts |>
  parameters() |>
  print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
(Intercept)	-5.50	0.67	(-6.83, -4.19)	-8.18	$< .001$
dibpat (Type B)	-0.30	1.18	(-2.63, 2.02)	-0.26	0.797
age	0.07	0.01	(0.05, 0.10)	5.24	$< .001$
dibpat (Type B) $\times$ age	-0.01	0.02	(-0.06, 0.04)	-0.42	0.674

Compare with what we had before:

```
chd_glm_strat |>
  parameters() |>
  print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
dibpat (Type A)	-5.50	0.67	(-6.83, -4.19)	-8.18	$< .001$
dibpat (Type B)	-5.80	0.98	(-7.73, -3.90)	-5.95	$< .001$
dibpat (Type A) $\times$ age	0.07	0.01	(0.05, 0.10)	5.24	$< .001$
dibpat (Type B) $\times$ age	0.06	0.02	(0.02, 0.10)	3.01	0.003

If I give you model 1, how would you get the coefficients of model 2?

0.18 Fitting logistic regression models

0.18.1

In general, the estimating equation $\ell'(\beta; \mathbf{x}) = 0$ cannot be solved analytically.

Instead, we have to use a variant of the Newton-Raphson method, which was discussed briefly in Epi 203. We won't go over it in this class; if you need to learn it, see Chapter 4 of Dobson and Barnett.

For now, all you need to know is that we make an iterative series of guesses, and each guess helps us make the next guess better (higher log-likelihood).

You can see some information about this process like so:

```
options(digits = 8)
temp =
  wcgs |>
  glm(
    control = glm.control(trace = TRUE),
    "data" = _,
    "formula" = chd69 == "Yes" ~ dibpat*age,
    "family" = binomial(link = "logit")
  )
```

```
Deviance = 1775.7899 Iterations - 1
Deviance = 1708.5396 Iterations - 2
Deviance = 1704.0434 Iterations - 3
Deviance = 1703.9833 Iterations - 4
Deviance = 1703.9832 Iterations - 5
Deviance = 1703.9832 Iterations - 6
```

After each iteration of the fitting procedure, the deviance $(2(\ell_{\text{full}} - \ell(\hat{\beta}))$) is printed. You can see that the algorithm took six iterations to converge to a solution where the likelihood wasn't changing much anymore.

0.19 Model comparisons for logistic models

0.19.1 Deviance test

We can compare the maximized log-likelihood of our model, $\ell(\hat{\beta}; \mathbf{x})$, versus the log-likelihood of the full model (aka saturated model aka maximal model), ℓ_{full}, which has one parameter

per covariate pattern. With enough data, $2(\ell_{\text{full}} - \ell(\hat{\beta}; \mathbf{x})) \dot{\sim} \chi^2(N-p)$, where N is the number of distinct covariate patterns and p is the number of β parameters in our model. A significant p-value for this **deviance** statistic indicates that there's some detectable pattern in the data that our model isn't flexible enough to catch.

> **Caution**
>
> The deviance statistic needs to have a large amount of data **for each covariate pattern** for the χ^2 approximation to hold. A guideline from Dobson is that if there are q distinct covariate patterns $x_1 ..., x_q$, with $n_1, ..., n_q$ observations per pattern, then the expected frequencies $n_k \cdot \pi(x_k)$ should be at least 1 for every pattern $k \in 1 : q$.

If you have covariates measured on a continuous scale, you may not be able to use the deviance tests to assess goodness of fit.

0.19.2 Hosmer-Lemeshow test

If our covariate patterns produce groups that are too small, a reasonable solution is to make bigger groups by merging some of the covariate-pattern groups together.

Hosmer and Lemeshow (1980) proposed that we group the patterns by their predicted probabilities according to the model of interest. For example, you could group all of the observations with predicted probabilities of 10% or less together, then group the observations with 11%-20% probability together, and so on; $g = 10$ categories in all.

Then we can construct a statistic

$$X^2 = \sum_{c=1}^{g} \frac{(o_c - e_c)^2}{e_c}$$

where o_c is the number of events *observed* in group c, and e_c is the number of events expected in group c (based on the sum of the fitted values $\hat{\pi}_i$ for observations in group c).

If each group has enough observations in it, you can compare X^2 to a χ^2 distribution; by simulation, the degrees of freedom has been found to be approximately $g - 2$.

For our CHD model, this procedure would be:

```
wcgs =
  wcgs |>
  mutate(
    pred_probs_glm1 = chd_glm_strat |> fitted(),
    pred_prob_cats1 =
      pred_probs_glm1 |>
      cut(breaks = seq(0, 1, by = .1),
```

```
          include.lowest = TRUE))

HL_table =
  wcgs |>
  summarize(
    .by = pred_prob_cats1,
    n = n(),
    o = sum(chd69 == "Yes"),
    e = sum(pred_probs_glm1)
  )

HL_table |> pander()
```

pred_prob_cats1	n	o	e
(0.1,0.2]	785	116	108
(0.2,0.3]	64	12	13.77
[0,0.1]	2305	129	135.2

```
X2 = HL_table |>
  summarize(
    `X^2` = sum((o-e)^2/e)
  ) |>
  pull(`X^2`)
print(X2)
```

```
[1] 1.1102871
```

```
pval1 = pchisq(X2, lower = FALSE, df = nrow(HL_table) - 2)
```

Our statistic is $X^2 = 1.1103$; $p(\chi^2(1) > 1.1103) = 0.292$, which is our p-value for detecting a lack of goodness of fit.

Unfortunately that grouping plan left us with just three categories with any observations, so instead of grouping by 10% increments of predicted probability, typically analysts use deciles of the predicted probabilities:

```
wcgs =
  wcgs |>
  mutate(
```

```
    pred_probs_glm1 = chd_glm_strat |> fitted(),
    pred_prob_cats1 =
      pred_probs_glm1 |>
      cut(breaks = quantile(pred_probs_glm1, seq(0, 1, by = .1)),
          include.lowest = TRUE))

HL_table =
  wcgs |>
  summarize(
    .by = pred_prob_cats1,
    n = n(),
    o = sum(chd69 == "Yes"),
    e = sum(pred_probs_glm1)
  )

HL_table |> pander()
```

pred_prob_cats1	n	o	e
(0.114,0.147]	275	48	36.81
(0.147,0.222]	314	51	57.19
(0.0774,0.0942]	371	27	32.56
(0.0942,0.114]	282	30	29.89
(0.0633,0.069]	237	17	15.97
(0.069,0.0774]	306	20	22.95
(0.0487,0.0633]	413	27	24.1
(0.0409,0.0487]	310	14	14.15
[0.0322,0.0363]	407	16	13.91
(0.0363,0.0409]	239	7	9.48

```
X2 = HL_table |>
  summarize(
    `X^2` = sum((o-e)^2/e)
  ) |>
  pull(`X^2`)

print(X2)
```

```
[1] 6.7811383
```

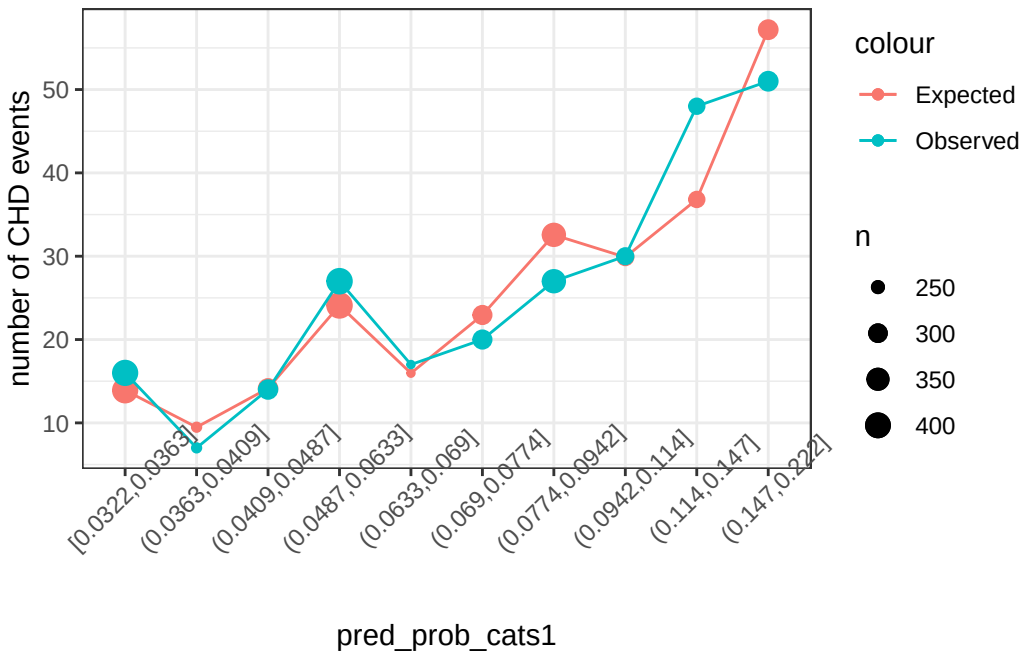
```
pval1 = pchisq(X2, lower = FALSE, df = nrow(HL_table) - 2)
```

Now we have more evenly split categories. The p-value is 0.5604, still not significant.

Graphically, we have compared:

```
HL_plot =
  HL_table |>
  ggplot(aes(x = pred_prob_cats1)) +
  geom_line(aes(y = e, x = pred_prob_cats1, group = "Expected", col = "Expected")) +
  geom_point(aes(y = e, size = n, col = "Expected")) +
  geom_point(aes(y = o, size = n, col = "Observed")) +
  geom_line(aes(y = o, col = "Observed", group = "Observed")) +
  scale_size(range = c(1,4)) +
  theme_bw() +
  ylab("number of CHD events") +
  theme(axis.text.x = element_text(angle = 45))
```

```
print(HL_plot)
```



0.19.3 Comparing models

- AIC $= -2 * \ell(\hat{\theta}) + 2 * p$ [lower is better]
- BIC $= -2 * \ell(\hat{\theta}) + p * \log(n)$ [lower is better]
- likelihood ratio [higher is better]

0.20 Residual-based diagnostics

0.20.1 Logistic regression residuals only work for grouped data

Residuals only work if there is more than one observation for most covariate patterns.

Here we will create the grouped-data version of our CHD model from the WCGS study:

```
wcgs_grouped =
  wcgs |>
  summarize(
    .by = c(dibpat, age),
    n = n(),
    chd = sum(chd69 == "Yes"),
    `!chd` = sum(chd69 == "No")
  )

chd_glm_strat_grouped = glm(
  "formula" = cbind(chd, `!chd`) ~ dibpat + dibpat:age - 1,
  "data" = wcgs_grouped,
  "family" = binomial(link = "logit")
)

chd_glm_strat_grouped |> parameters() |> print_md()
```

Parameter	Log-Odds	SE	95% CI	z	p
dibpat (Type A)	-5.50	0.67	(-6.83, -4.19)	-8.18	$< .001$
dibpat (Type B)	-5.80	0.98	(-7.73, -3.90)	-5.95	$< .001$
dibpat (Type A) $\times$ age	0.07	0.01	(0.05, 0.10)	5.24	$< .001$
dibpat (Type B) $\times$ age	0.06	0.02	(0.02, 0.10)	3.01	0.003

0.20.2 (Response) residuals

$$e_k \stackrel{\text{def}}{=} \bar{y}_k - \hat{\pi}(x_k)$$

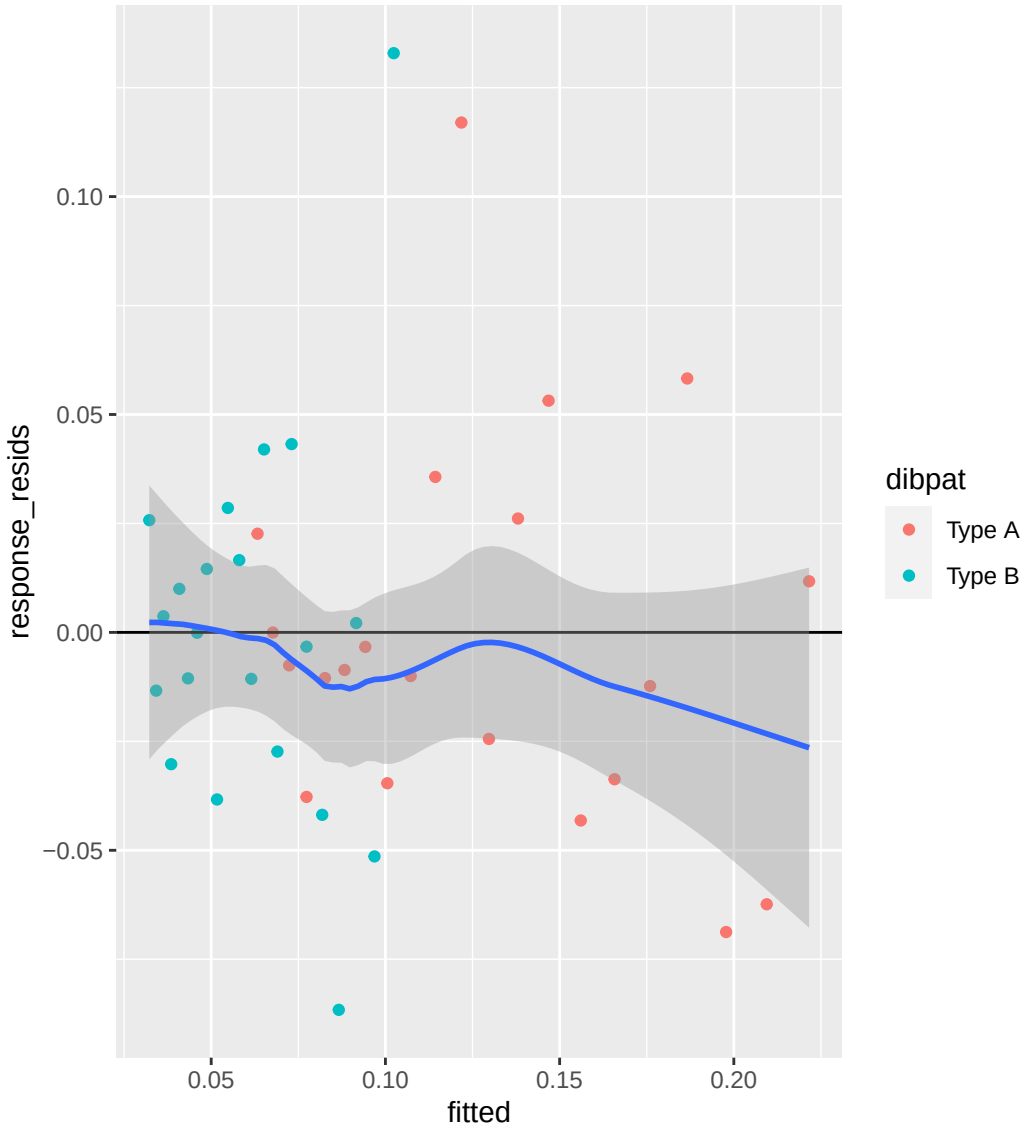
(k indexes the covariate patterns)

We can graph these residuals e_k against the fitted values $\hat{\pi}(x_k)$:

```
wcgs_grouped =
  wcgs_grouped |>
  mutate(
    fitted = chd_glm_strat_grouped |> fitted(),
    fitted_logit = fitted |> logit(),
    response_resids =
      chd_glm_strat_grouped |> resid(type = "response")
  )

wcgs_response_resid_plot =
  wcgs_grouped |>
  ggplot(
    mapping = aes(
      x = fitted,
      y = response_resids
    )
  ) +
  geom_point(
    aes(col = dibpat)
  ) +
  geom_hline(yintercept = 0) +
  geom_smooth( #<1>
    se = TRUE,  #<1>
    method.args = list( #<1>
      span=2/3, #<1>
      degree=1, #<1>
      family="symmetric", #<1>
      iterations=3), #<1>
    method = stats::loess) #<1>
```

① Don't worry about these options for now; I chose them to match `autoplot()` as closely as I can. `plot.glm` and `autoplot` use `stats::lowess` instead of `stats::loess`; `stats::lowess` is older, hard to use with `geom_smooth`, and hard to match exactly with `stats::loess`; see https://support.bioconductor.org/p/2323/.]

```
wcgs_response_resid_plot |> print()
```



We can see a slight fan-shape here: observations on the right have larger variance (as expected since $var(\bar{y}) = \pi(1-\pi)/n$ is maximized when $\pi = 0.5$).

0.20.3 Pearson residuals

The fan-shape in the response residuals plot isn't necessarily a concern here, since we haven't made an assumption of constant residual variance, as we did for linear regression.

However, we might want to divide by the standard error in order to make the graph easier to interpret. Here's one way to do that:

The Pearson (chi-squared) residual for covariate pattern k is:

$$X_k = \frac{\bar{y}_k - \hat{\pi}_k}{\sqrt{\hat{\pi}_k(1-\hat{\pi}_k)/n_k}}$$

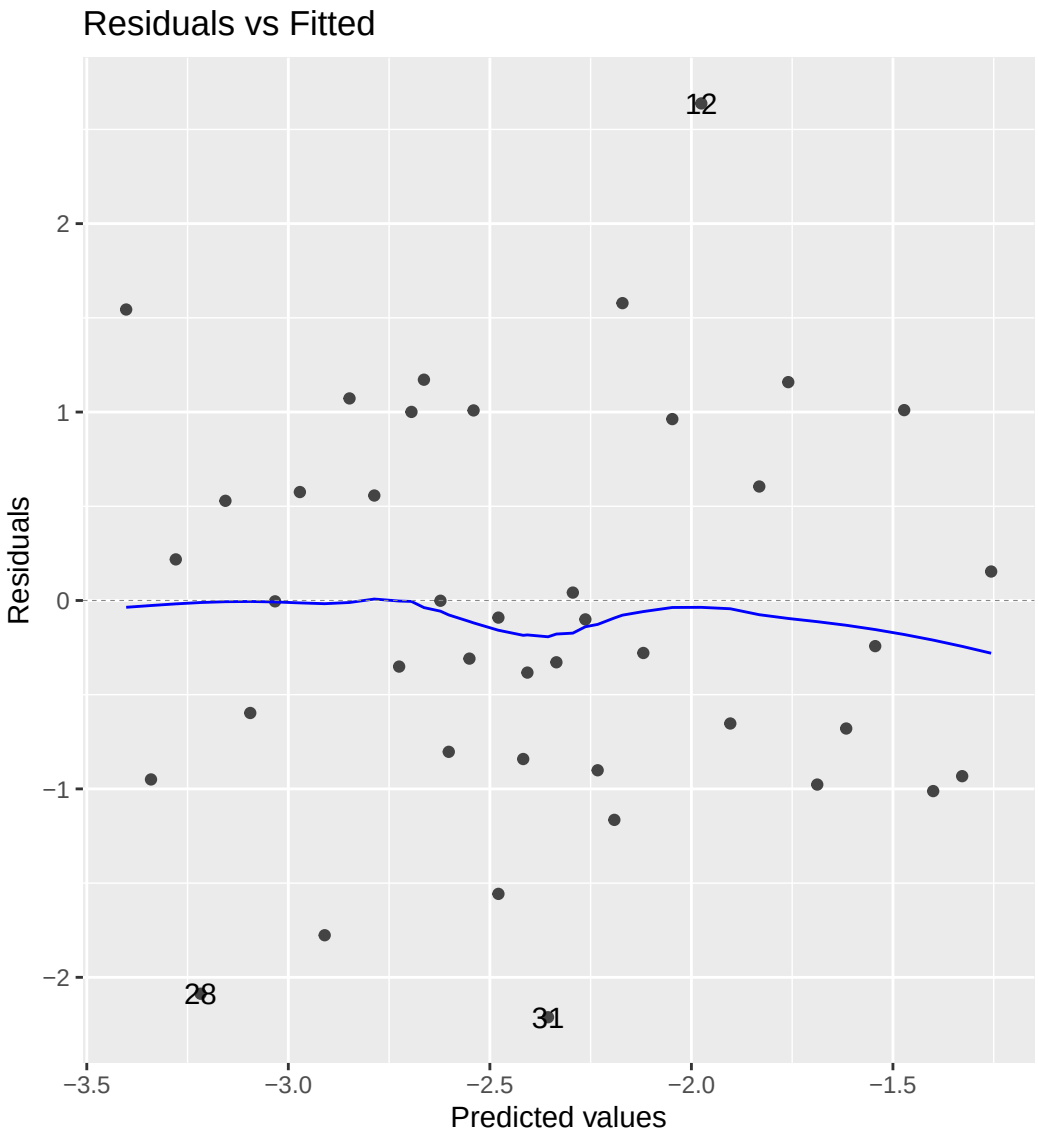
where

$$\begin{aligned}
\hat{\pi}_k &\stackrel{\text{def}}{=} \hat{\pi}(x_k) \\
&\stackrel{\text{def}}{=} \hat{P}(Y = 1|X = x_k) \\
&\stackrel{\text{def}}{=} \text{expit}(x_i'\hat{\beta}) \\
&\stackrel{\text{def}}{=} \text{expit}(\hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j x_{ij})
\end{aligned}$$

Let's take a look at the Pearson residuals for our CHD model from the WCGS data (graphed against the fitted values on the logit scale):

```
library(ggfortify)
```

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

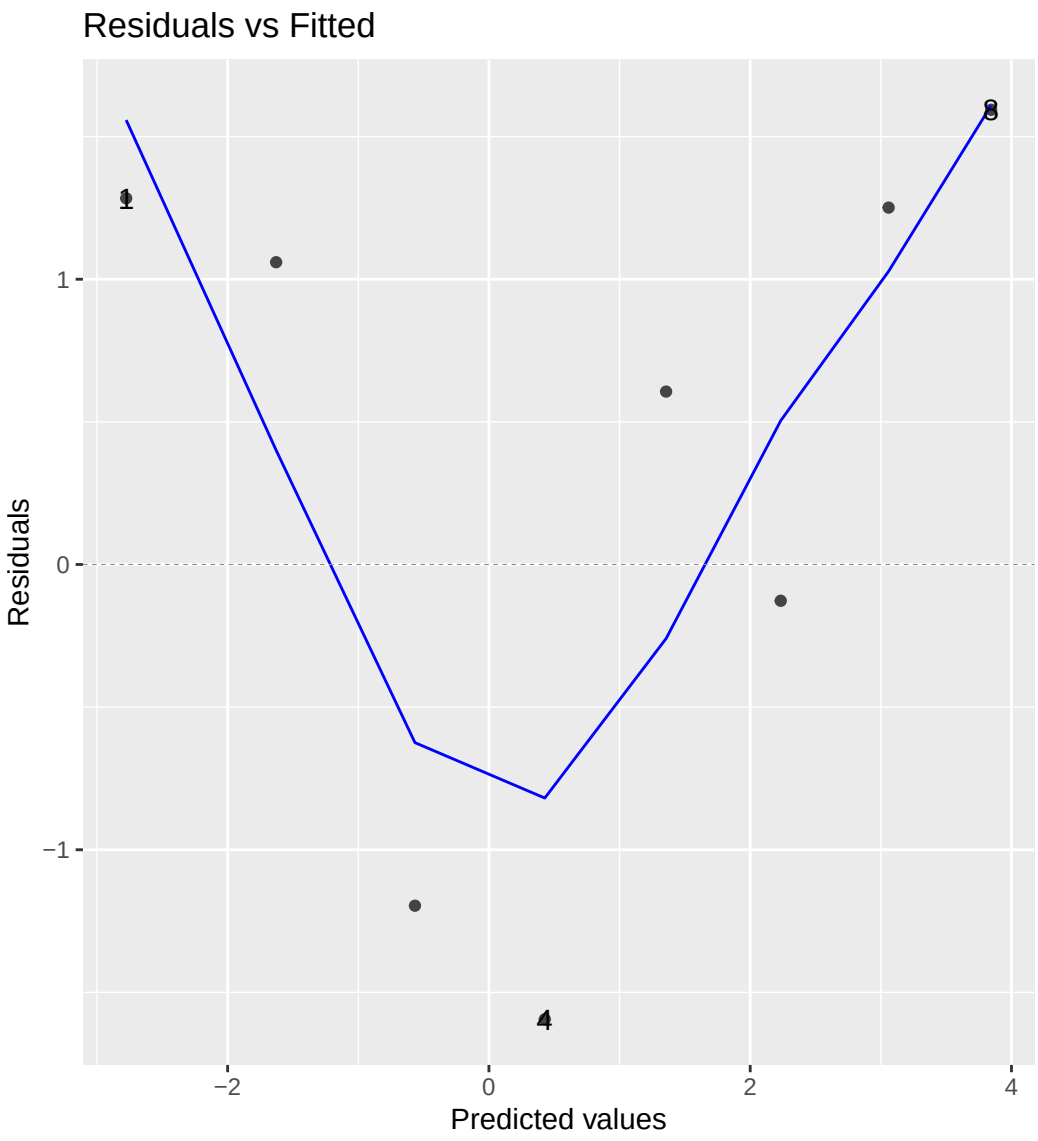


The fan-shape is gone, and these residuals don't show any obvious signs of model fit issues.

0.20.3.1 Pearson residuals plot for `beetles` data

If we create the same plot for the `beetles` model, we see some strong evidence of a lack of fit:

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

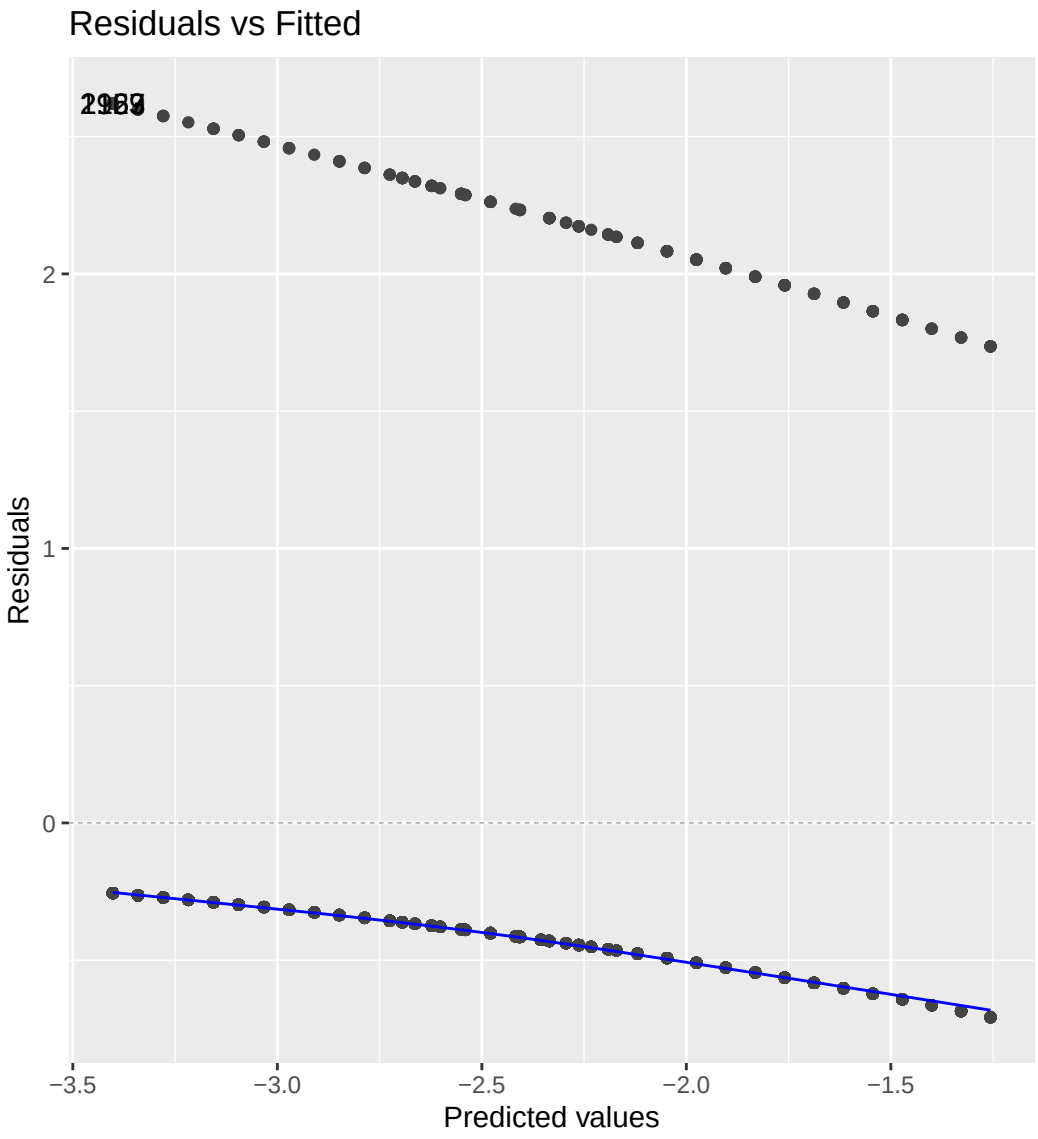


0.20.3.2 Pearson residuals with individual (ungrouped) data

What happens if we try to compute residuals without grouping the data by covariate pattern?

```
library(ggfortify)
```

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



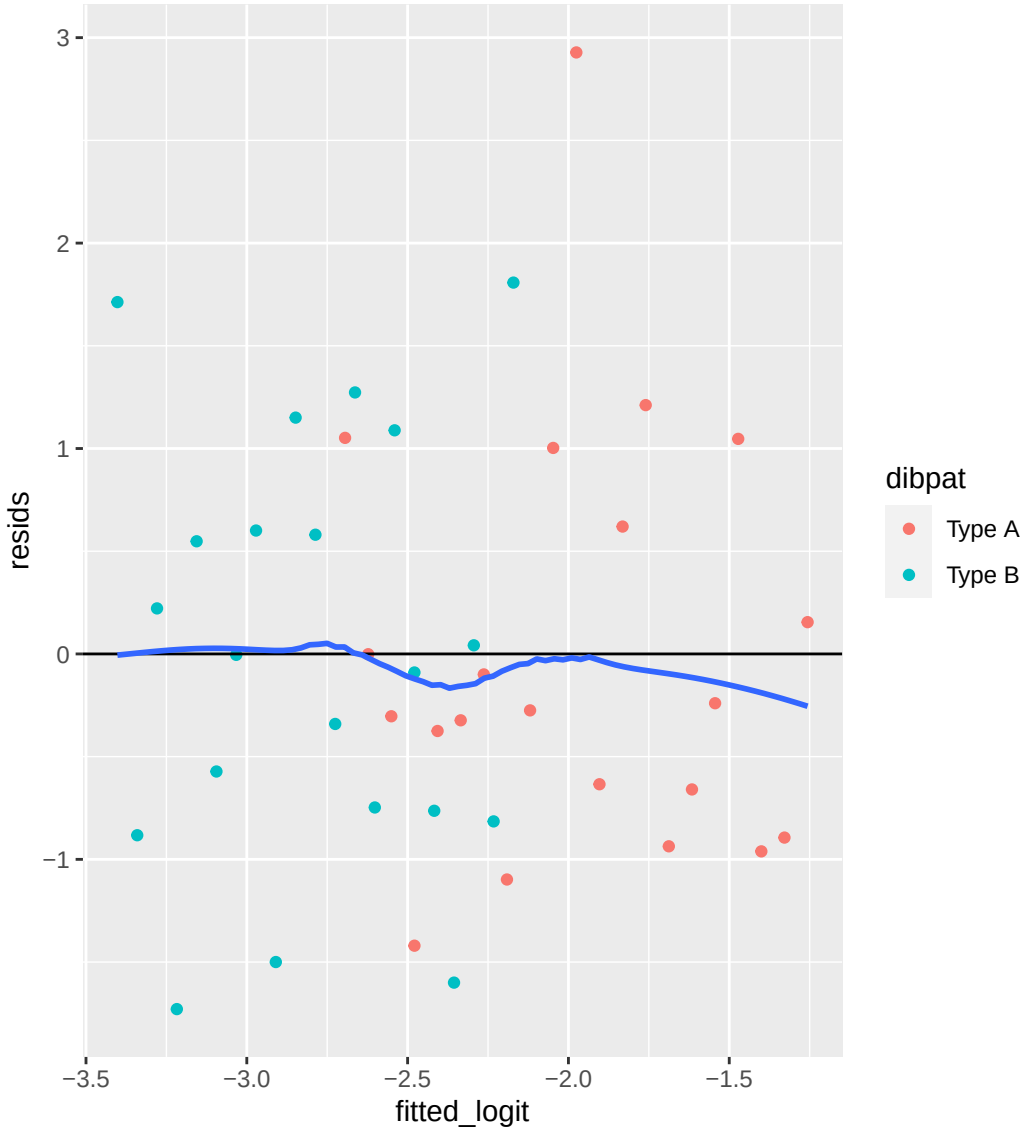
Meaningless.

0.20.3.3 Residuals plot by hand (*optional section*)

If you want to check your understanding of what these residual plots are, try building them yourself:

```
wcgs_grouped =
  wcgs_grouped |>
  mutate(
    fitted = chd_glm_strat_grouped |> fitted(),
    fitted_logit = fitted |> logit(),
    resids = chd_glm_strat_grouped |> resid(type = "pearson")
  )

wcgs_resid_plot1 =
  wcgs_grouped |>
  ggplot(
    mapping = aes(
      x = fitted_logit,
      y = resids

    )

  ) +
  geom_point(
    aes(col = dibpat)
  ) +
  geom_hline(yintercept = 0) +
  geom_smooth(se = FALSE,
              method.args = list(
                span=2/3,
                degree=1,
                family="symmetric",
                iterations=3,
                surface="direct"
                # span = 2/3,
                # iterations = 3
              ),
              method = stats::loess)
# plot.glm and autoplot use stats::lowess, which is hard to use with
# geom_smooth and hard to match exactly;
# see https://support.bioconductor.org/p/2323/
```

```
wcgs_resid_plot1 |> print()
```



0.20.4 Pearson chi-squared goodness of fit test

The Pearson chi-squared goodness of fit statistic is:

$$X^2 = \sum_{k=1}^{m} X_k^2$$

Under the null hypothesis that the model in question is correct (i.e., sufficiently complex), $X^2 \dot{\sim} \chi^2(N-p)$.

```r
X = chd_glm_strat_grouped |>
  resid(type = "pearson")

chisq_stat = sum(X^2)

pval = pchisq(
  chisq_stat,
  lower = FALSE,
  df = length(X) - length(coef(chd_glm_strat_grouped)))
```

For our CHD model, the p-value for this test is 0.2652; no significant evidence of a lack of fit at the 0.05 level.

0.20.4.1 Standardized Pearson residuals

Especially for small data sets, we might want to adjust our residuals for leverage (since outliers in X add extra variance to the residuals):

$$r_{P_k} = \frac{X_k}{\sqrt{1-h_k}}$$

where h_k is the leverage of X_k. The functions `autoplot()` and `plot.lm()` use these for some of their graphs.

0.20.5 Deviance residuals

For large sample sizes, the Pearson and deviance residuals will be approximately the same. For small sample sizes, the deviance residuals from covariate patterns with small sample sizes can be unreliable (high variance).

$$d_k = \text{sign}(y_k - n_k\hat{\pi}_k)\left\{\sqrt{2[\ell_{\text{full}}(x_k) - \ell(\hat{\beta}; x_k)]}\right\}$$

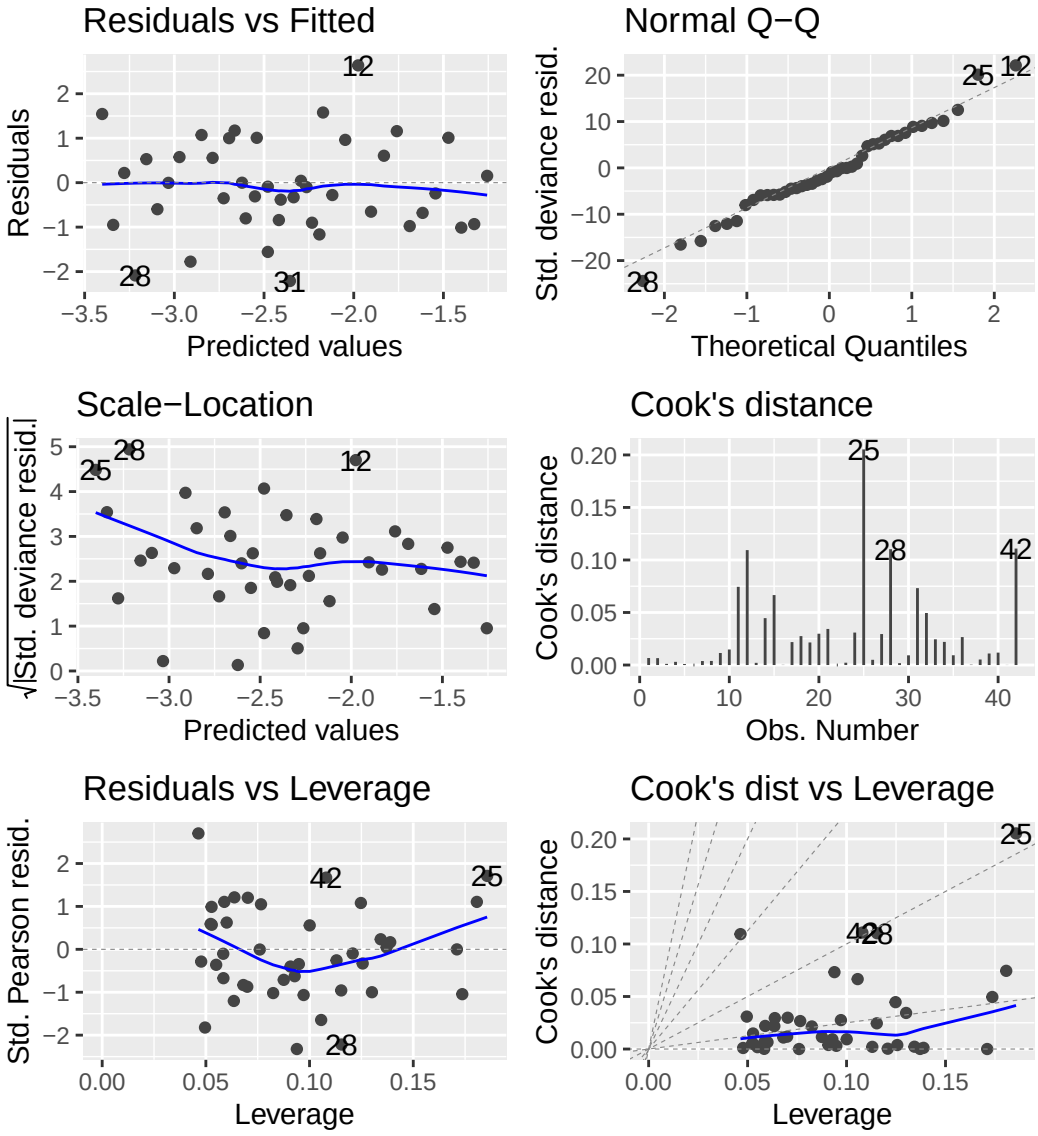
0.20.5.1 Standardized deviance residuals

$$r_{D_k} = \frac{d_k}{\sqrt{1-h_k}}$$

0.20.6 Diagnostic plots

Let's take a look at the full set of `autoplot()` diagnostics now for our `CHD` model:

```
chd_glm_strat_grouped |> autoplot(which = 1:6) |> print()
```

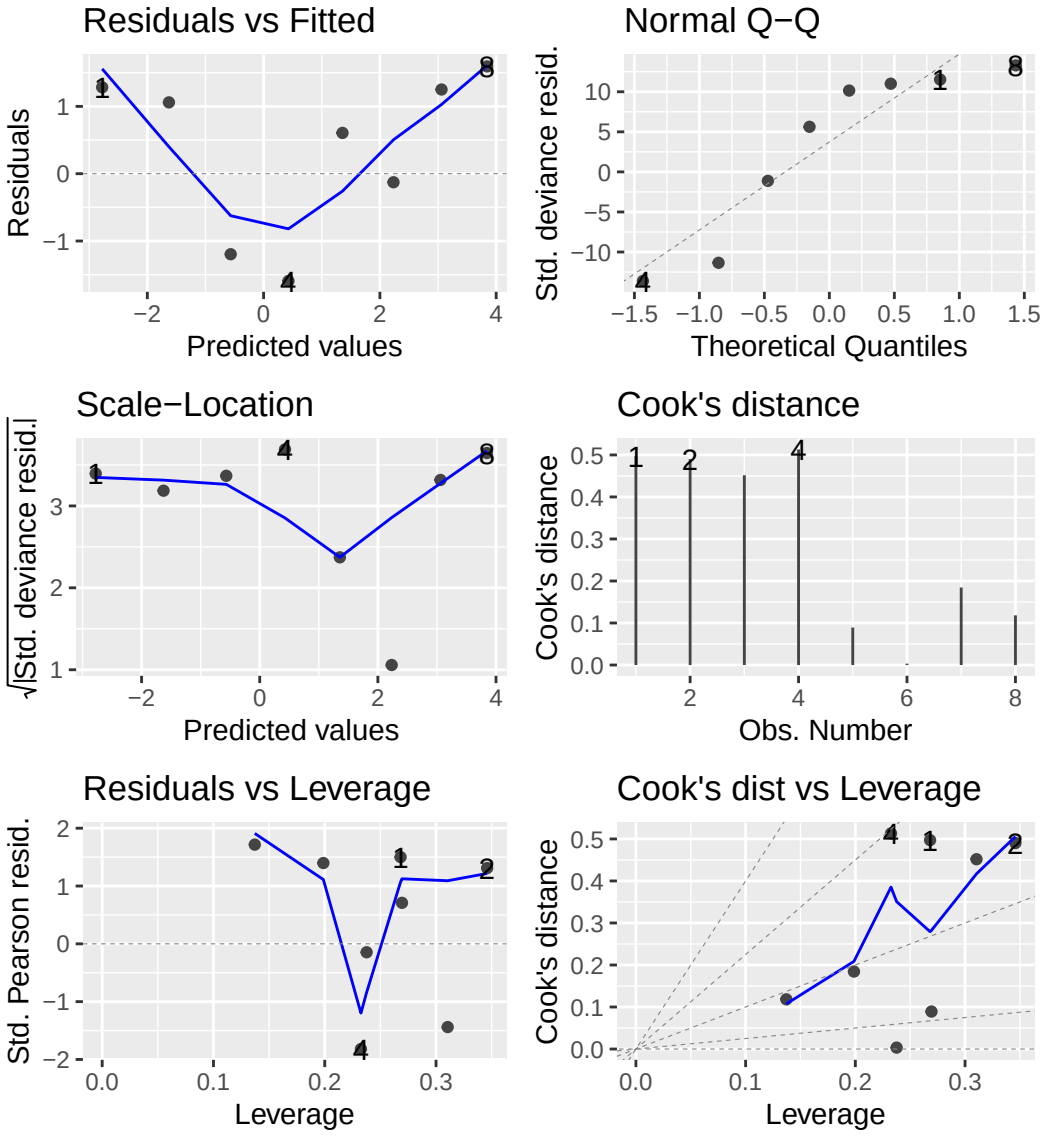


Things look pretty good here. The QQ plot is still usable; with large samples; the residuals should be approximately Gaussian.

0.20.6.1 Beetles

Let's look at the beetles model diagnostic plots for comparison:

```
beetles_glm_grouped |> autoplot(which = 1:6) |> print()
```



Hard to tell much from so little data, but there might be some issues here.

0.21 Odds Ratios vs Probability (Risk) Ratios

Definition 0.11 (Relative risk)**.** The **relative risk** comparing two probabilities π_1 and π_2, also known as the **risk ratio**, **relative risk ratio**, **probability ratio**, and **rate ratio**, is:

$$\rho(1,2) \stackrel{\text{def}}{=} \frac{\pi_1}{\pi_2}$$

0.21.0.1 Case 1: rare events

For rare events, odds ratios and probability (a.k.a. risk, a.k.a. prevalence) ratios will be close:

$\pi_1 = .01$ $\pi_2 = .02$

```
pi1 = .01
pi2 = .02
pi2/pi1
```

```
[1] 2
```

```
odds(pi2)/odds(pi1)
```

```
[1] 2.0204082
```

0.21.0.2 Case 2: frequent events

$\pi_1 = .4$ $\pi_2 = .5$

For more frequently-occurring outcomes, this won't be the case:

```
pi1 = .4
pi2 = .5
pi2/pi1
```

```
[1] 1.25
```

```
odds(pi2)/odds(pi1)
```

```
[1] 1.5
```

If you want risk ratios, you can sometimes get them by changing the link function:

```
data(anthers, package = "dobson")
anthers.sum<-aggregate(
  anthers[c("n","y")],
  by=anthers[c("storage")],FUN=sum)

anthers_glm_log = glm(
  formula = cbind(y,n-y)~storage,
  data=anthers.sum,
  family=binomial(link="log"))

anthers_glm_log |> parameters() |> print_md()
```

Parameter	Log-Risk	SE	95% CI	z	p
(Intercept)	-0.80	0.12	(-1.04, -0.58)	-6.81	< .001
storage	0.17	0.07	(0.02, 0.31)	2.31	0.021

Now $\exp\{\beta\}$ gives us risk ratios instead of odds ratios:

```
anthers_glm_log |> parameters(exponentiate = TRUE) |> print_md()
```

Parameter	Risk Ratio	SE	95% CI	z	p
(Intercept)	0.45	0.05	(0.35, 0.56)	-6.81	< .001
storage	1.18	0.09	(1.03, 1.36)	2.31	0.021

Let's compare this model with a logistic model:

```
anthers_glm_logit = glm(
  cbind(y,n-y)~storage,
  data=anthers.sum,
  family=binomial(link="logit"))

anthers_glm_logit |> parameters(exponentiate = TRUE) |> print_md()
```

Parameter	Odds Ratio	SE	95% CI	z	p
(Intercept)	0.76	0.20	(0.45, 1.27)	-1.05	0.296
storage	1.49	0.26	(1.06, 2.10)	2.29	0.022

[to add: fitted plots on each outcome scale]

When I try to use `link ="log"` in practice, I often get errors about not finding good starting values for the estimation procedure. This is likely because the model is producing fitted probabilities greater than 1.

When this happens, you can try to fit Poisson regression models instead (we will see those soon!). But then the outcome distribution isn't quite right, and you won't get warnings about fitted probabilities greater than 1. In my opinion, the Poisson model for binary outcomes is confusing and not very appealing.

0.22 Models for Count Outcomes

(Poisson Regression and Variations)

Acknowledgements

This content is adapted from:

- Dobson & Barnett 2018, "An Introduction to Generalized Linear Models", 4th edition, Chapter 9
- Vittinghoff et al 2012, "Regression Methods in Biostatistics", 2nd edition, Chapter 8

Configuring R

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(dplyr) # manipulate data
library(haven) # import Stata files
library(knitr) # format R output for markdown
```

```r
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(conflicted) # check for conflicting function definitions
```

Here are some R settings I use in this document:

```r
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
options('digits' = 4)
```

0.23 Introduction

0.23.1 Examples of count outcomes

- Cyclones per season
- Seconds of tooth-brushing per session (if rounded)
- Infections per person-year
- Visits to ER per person-month
- Car accidents per 1000 miles driven

> **Note**
>
> In many count outcomes, there is some sense of "exposure magnitude" or "duration of observation": person-year, time at risk, session, miles driven, etc.

0.23.2 Poisson distribution

$$P(Y = y) = \frac{\mu^y e^{-\mu}}{y!}$$

0.23.2.1 Properties

- $\mathbb{E}[Y] = \mu$
- $\text{Var}[Y] = \mu$

0.23.3 Accounting for exposure

If the exposures/observation durations, denoted $T = t$, are not all equal, we model

$$\mu = \lambda t$$

λ is interpreted as the "expected event rate per unit of exposure"; that is,

$$\lambda = \frac{\mathbb{E}[Y|T = t]}{t}$$

> **!** Important
>
> The exposure magnitude, T, is *similar* to a covariate in linear or logistic regression. However, there is an important difference: in count regression, **there is no intercept corresponding to** $\mathbb{E}[Y|T = 0]$. In other words, this model assumes that if there is no exposure, there can't be any events.

0.23.4 Adding covariates

With covariates, λ becomes a function of the covariates $\tilde{X} = (X_1, \ldots, X_n)$, with a $\log\{\}$ link function (and thus an $\exp\{\}$ inverse-link). That is:

$$\begin{aligned}
\mathbb{E}[Y|\tilde{X} = \tilde{x}, T = t] &= \mu(\tilde{x}, t) \\
\mu(\tilde{x}, t) &= \lambda(\tilde{x}) \cdot t \\
\lambda(\tilde{x}) &= \exp\{\eta(\tilde{x})\} \\
\eta(\tilde{x}) &= \tilde{x}'\tilde{\beta} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p
\end{aligned}$$

Therefore,

$$\begin{aligned}
\log\Big\{\{\mathbb{E}[Y|\tilde{X} = \tilde{x}, T = t]\}\Big\} &= \log\{\{\mu(\tilde{x})\}\} \\
&= \log\{\{\lambda(\tilde{x}) \cdot t\}\} \\
&= \log\{\lambda(\tilde{x})\} + \log\{t\} \\
&= \log\{\exp\{\eta(\tilde{x})\}\} + \log\{t\} \\
&= \eta(\tilde{x}) + \log\{t\} \\
&= \tilde{x}'\tilde{\beta} + \log\{t\} \\
&= (\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p) + \log\{t\}
\end{aligned}$$

In contrast with the Xs, T enters this expression with a $\log\{\}$ transformation and without a corresponding β coefficient.

> **Note**
>
> Terms that enter the linear component of a model without a coefficient, such as $\log\{t\}$ here, are called **offsets**.

0.23.5 Rate ratios

Differences on the log-rate scale become ratios on the rate scale.

> **Tip**
>
> $$\exp\{a-b\} = \frac{\exp\{a\}}{\exp\{b\}}$$

Therefore, according to this model, **differences of δ in covariate x_j correspond to rate ratios of exp $\{\beta_j \cdot \delta\}$**.

That is, letting $\tilde{X}_{-j}$ denote vector $\tilde{X}$ with element j removed:

$$\begin{aligned}
&\left\{ \begin{array}{c} \log\left\{\mathbb{E}[Y|X_j = a, \tilde{X}_{-j} = \tilde{x}_{-j}, T = t]\right\} \\ -\log\left\{\mathbb{E}[Y|X_j = b, \tilde{X}_{-j} = \tilde{x}_{-j}, T = t]\right\} \end{array} \right\} \\
&= \left\{ \begin{array}{c} \log\{t\} + \beta_0 + \beta_1 x_1 + ... + \beta_j(a) + ... + \beta_p x_p \\ -\log\{t\} + \beta_0 + \beta_1 x_1 + ... + \beta_j(b) + ... + \beta_p x_p \end{array} \right\} \\
&= \beta_j(a-b)
\end{aligned}$$

And accordingly,

$$\frac{\mathbb{E}[Y|X_j = a, \tilde{X}_{-j} = \tilde{x}_{-j}, T = t]}{\mathbb{E}[Y|X_j = b, \tilde{X}_{-j} = \tilde{x}_{-j}, T = t]} = \exp\{\beta_j(a-b)\}$$

0.24 Inference for count regression models

0.24.1 Confidence intervals for regression coefficients and rate ratios

As usual:

$$\beta \in \left[\hat{\beta} \pm z_{1-\frac{\alpha}{2}} \cdot \hat{\text{se}}\left(()\,\hat{\beta}\right)\right]$$

Rate ratios: exponentiate CI endpoints

$$\exp\{\beta\} \in \left[\exp\left\{\hat{\beta} \pm z_{1-\frac{\alpha}{2}} \cdot \widehat{\text{se}}(()\,\hat{\beta})\right\}\right]$$

0.24.2 Hypothesis tests for regression coefficients

$$t = \frac{\hat{\beta} - \beta_0}{\widehat{\text{se}}(()\,\hat{\beta})}$$

Compare t or $|t|$ to the tails of the standard Gaussian distribution, according to the null hypothesis.

0.24.3 Comparing nested models

log(likelihood ratio) tests, as usual.

0.25 Prediction

$$\begin{aligned}
\hat{y} &\stackrel{\text{def}}{=} \hat{\mathbb{E}}[Y|\tilde{X} = \tilde{x}, T = t] \\
&= \hat{\mu}(\tilde{x}, t) \\
&= \hat{\lambda}(\tilde{x}) \cdot t \\
&= \exp\{\hat{\eta}(\tilde{x})\} \cdot t \\
&= \exp\left\{\tilde{x}'\hat{\beta}\right\} \cdot t
\end{aligned}$$

0.26 Diagnostics

0.26.1 Residuals

0.26.1.1 Observation residuals

$$e \stackrel{\text{def}}{=} y - \hat{y}$$

0.26.1.2 Pearson residuals

$$r = \frac{e}{\widehat{\text{se}}(()\,e)} \approx \frac{e}{\sqrt{\hat{y}}}$$

0.26.1.3 Standardized Pearson residuals

$$r_p = \frac{r}{\sqrt{1-h}}$$

where h is the "leverage" (which we will continue to leave undefined).

0.26.1.4 Deviance residuals

$$d_k = \text{sign}(y - \hat{y}) \left\{ \sqrt{2[\ell_{\text{full}}(y) - \ell(\hat{\beta}; y)]} \right\}$$

> **Note**
>
> $$\text{sign}(x) \stackrel{\text{def}}{=} \frac{x}{|x|}$$
>
> In other words:
>
> - $\text{sign}(x) = -1$ if $x < 0$
> - $\text{sign}(x) = 0$ if $x = 0$
> - $\text{sign}(x) = 1$ if $x > 0$

0.27 Zero-inflation

0.27.1 Models for zero-inflated counts

We assume a latent (unobserved) binary variable, Z, which we model using logistic regression:

$$P(Z = 1|X = x) = \pi(x) = \text{expit}(\gamma_0 + \gamma_1 x_1 + ...)$$

According to this model, if $Z = 1$, then Y will always be zero, regardless of X and T:

$$P(Y = 0|Z = 1, X = x, T = t) = 1$$

Otherwise (if $Z = 0$), Y will have a Poisson distribution, conditional on X and T, as above.

Even though we never observe Z, we can estimate the parameters γ_0-γ_p, via maximum likelihood:

$$P(Y = y|X = x, T = t) = P(Y = y, Z = 1|...) + P(Y = y, Z = 0|...)$$

(by the Law of Total Probability)

where

$$P(Y = y, Z = z|...) = P(Y = y|Z = z, ...)P(Z = z|...)$$

0.27.1.1 Exercise

Expand $P(Y = 0|X = x, T = t)$, $P(Y = 1|X = x, T = t)$ and $P(Y = y|X = x, T = t)$ into expressions involving $P(Z = 1|X = x, T = t)$ and $P(Y = y|Z = 0, X = x, T = t)$.

0.27.1.2 Exercise

Derive the expected value and variance of Y, conditional on X and T, as functions of $P(Z = 1|X = x, T = t)$ and $\mathbb{E}[Y|Z = 0, X = x, T = t]$.

0.28 Over-dispersion

0.28.1 Negative binomial models

The Poisson distribution model **forces** the variance to equal the mean. In practice, many count distributions will have a variance substantially larger than the mean (or occasionally smaller).

When we encounter this, we can try to reduce the residual variance by adding more covariates. However, there are also alternatives to the Poisson model.

Most notably, the negative binomial model:

$$P(Y = y) = \frac{\mu^y}{y!} \cdot \frac{\Gamma(\rho + y)}{\Gamma(\rho) \cdot (\rho + \mu)^y} \cdot \left(1 + \frac{\mu}{\rho}\right)^{-\rho}$$

where ρ is an overdispersion parameter and $\Gamma(x) = (x - 1)!$ for integers x.

You don't need to memorize or understand this expression, but as $\rho \to \infty$, the second term converges to 1 and the third term converges to $\exp\{-\mu\}$, which brings us back to the Poisson distribution.

For this distribution, $\mathbb{E}[Y] = \mu$ and $\text{Var}(Y) = \mu + \frac{\mu^2}{\rho} > \mu$.

We can still model μ as a function of X and T as before, and we can combine this model with zero-inflation by using it in place of the Poisson distribution for $P(Y = y|Z = 0, X = x, T = t)$.

0.28.2 Quasipoisson

An alternative to Negative binomial is the "quasipoisson" distribution. I've never used it, but it seems to be a method-of-moments type approach rather than maximum likelihood. It models the variance as $\text{Var}(Y) = \mu\theta$, and estimates θ accordingly.

See `?quasipoisson` in R for more.

Time to Event Models

This section is adapted starting from the lecture notes by David Rocke from the 2021 edition of Epi 204:

http://dmrocke.ucdavis.edu/Class/EPI204-Spring-2021/EPI204-Spring-2021.html

0.29 Introduction to Survival Analysis

Configuring R

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(scales) # scales formatting
library(dplyr) # manipulate data
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(conflicted) # check for conflicting function definitions
```

Here are some R settings I use in this document:

```r
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
pander::panderOptions("table.split.table", Inf)
conflicts_prefer(dplyr::filter) # use the `filter()` function from dplyr() by default
options('digits' = 4)
```

0.30 Time-to-event outcome distributions

0.30.1 Time to Event Data

- Survival Analysis is a term for analyzing time-to-event data.
- This is used in clinical trials, where the event is often death or recurrence of disease.
- It is used in engineering reliability analysis, where the event is failure of a device or system.
- It is used in insurance, particularly life insurance, where the event is death.

0.30.2 Distributions of Time-to-Event Data

- The distribution of event times is asymmetric and can be long-tailed, and starts at 0 (that is, $P(T < 0) = 0$).
- The base distribution is not normal, but exponential.
- There are usually **censored** observations, which are ones in which the failure time is not observed.
- Often, these are **right-censored**, meaning that we know that the event occurred after some known time t, but we don't know the actual event time, as when a patient is still alive at the end of the study.
- Observations can also be **left-censored**, meaning we know the event has already happened at time t, or **interval-censored**, meaning that we only know that the event happened between times t_1 and t_2.
- Analysis is difficult if censoring is associated with treatment.

0.30.3 Right Censoring

- Patients are in a clinical trial for cancer, some on a new treatment and some on standard of care.
- Some patients in each group have died by the end of the study. We know the survival time (measured for example from time of diagnosis—each person on their own clock).
- Patients still alive at the end of the study are right censored.

- Patients who are lost to follow-up or withdraw from the study may be right-censored.

0.30.4 Left and Interval Censoring

- An individual tests positive for HIV.
- If the event is infection with HIV, then we only know that it has occurred before the testing time t, so this is left censored.
- If an individual has a negative HIV test at time t_1 and a positive HIV test at time t_2, then the infection event is interval censored.

0.30.5 Distribution functions for time-to-event variables

0.30.5.1 The Probability Density Function (PDF)

For a time-to-event variable T with a continuous distribution, the **probability density function** is defined as usual:

$$f(t) \stackrel{\text{def}}{=} p(t) \stackrel{\text{def}}{=} p(T = t)$$

Typically, this density is assumed to be 0 for all $t < 0$; that is, $f(t) = 0, \forall t < 0$. In other words, the range of T is typically $[0, \infty)$.

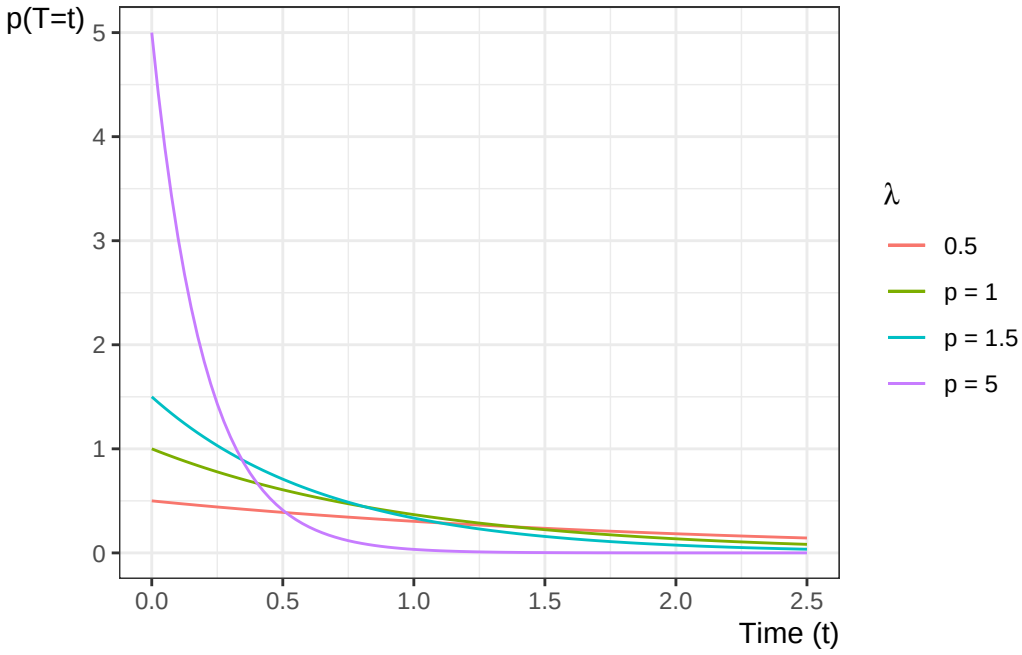
Example: exponential distribution

Recall from Epi 202: the pdf of the exponential distribution family of models is:

$$p(T = t) = \mathbb{1}_{t \geq 0} \cdot \lambda \mathrm{e}^{-\lambda t}$$

where $\lambda > 0$.

Here are some examples of exponential pdfs:



0.30.5.2 The Cumulative Distribution Function (CDF)

The **cumulative distribution function** is defined as:

$$\begin{aligned} F(t) &\stackrel{\text{def}}{=} \Pr(T \leq t) \\ &= \int_{u=0}^{t} f(u)du \end{aligned}$$

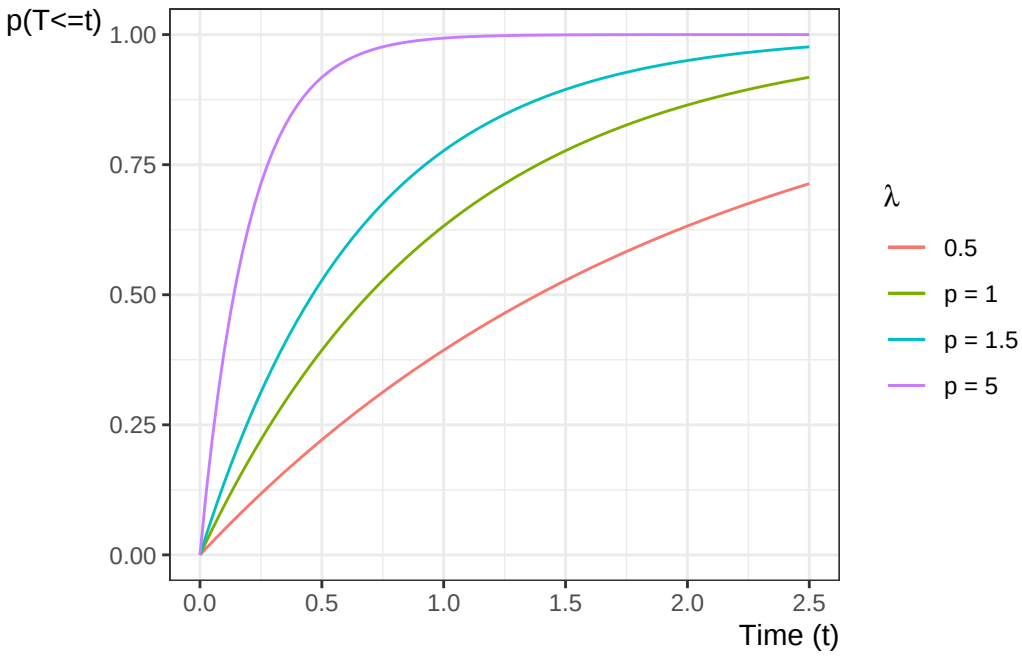
Example: exponential distribution

Recall from Epi 202: the cdf of the exponential distribution family of models is:

$$P(T \leq t) = \mathbb{1}_{t \geq 0} \cdot (1 - \mathrm{e}^{-\lambda t})$$

where $\lambda > 0$.

Here are some examples of exponential cdfs:



0.30.5.3 The Survival Function

For survival data, a more important quantity is the **survival function**:

$$\begin{aligned} S(t) &\stackrel{\text{def}}{=} \Pr(T > t) \\ &= \int_{u=t}^{\infty} p(u)du \\ &= 1 - F(t) \end{aligned}$$

The survival function $S(t)$ is the probability that the event time is later than t. If the event in a clinical trial is death, then this is the expected fraction of the original population at time 0 that is still alive at time t; that is, the fraction surviving to time t.

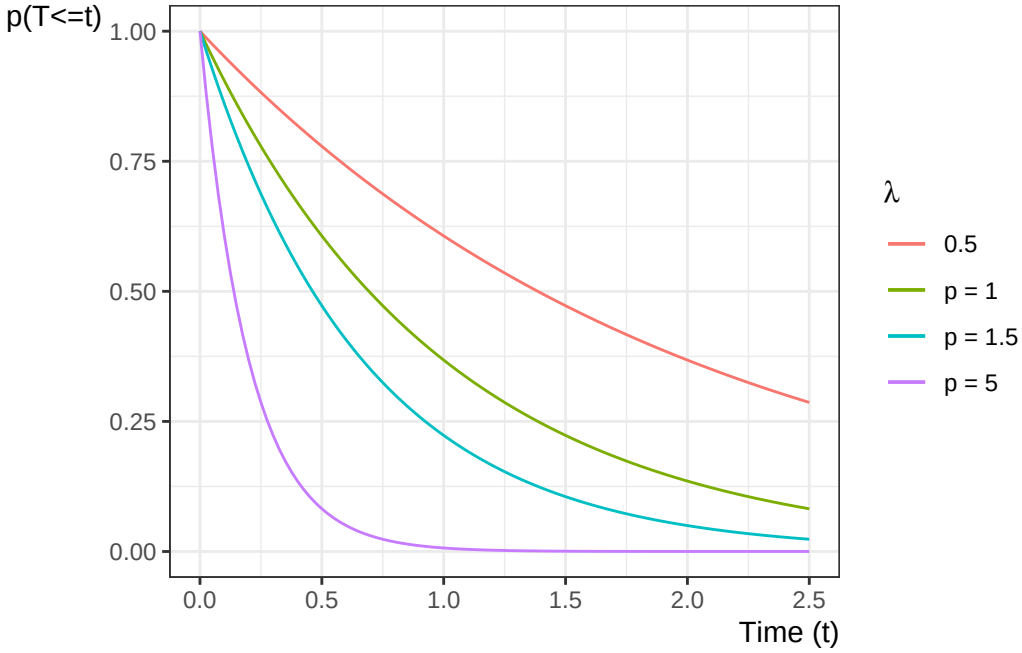
Example: exponential distribution

Since $S(t) = 1 - F(t)$, the survival function of the exponential distribution family of models is:

$$P(T > t) = \mathbb{1}_{t \geq 0} \cdot \mathrm{e}^{-\lambda t}$$

where $\lambda > 0$.

Here are some examples of exponential pdfs:



0.30.5.4 The Hazard Function

Another important quantity is the **hazard function**, which represents the probability that the event will occur at time t, given that it has not occurred yet:

$$h(t) \stackrel{\text{def}}{=} p(T = t | T \geq t)$$

> **Note**
>
> Recall from Epi 202: if $A \subseteq B$ then $p(A \cap B) = p(A)$.

Now, $\{T = t\} \subseteq \{T \geq t\}$. So:

$$\begin{aligned} p(T = t, T \geq t) &= p(T = t \cap T \geq t) \\ &= p(T = t) \end{aligned}$$

Hence:

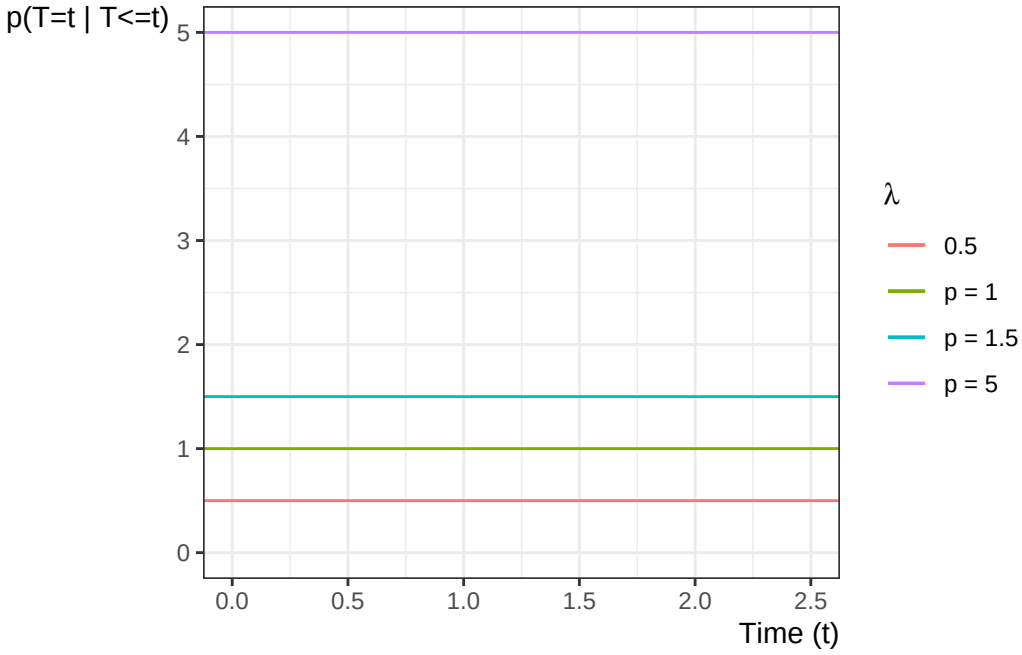
$$
\begin{aligned}
h(t) &= p(T = t | T \geq t) \\
&= \frac{p(T = t, T \geq t)}{p(T \geq t)} \\
&= \frac{p(T = t)}{p(T \geq t)} \\
&= \frac{f(t)}{S(t)}
\end{aligned}
$$

Example: exponential distribution

The hazard function of the exponential distribution family of models is:

$$P(T > t) = \mathbb{1}_{t \geq 0} \cdot \lambda$$

Here are some examples of exponential hazard functions:



0.30.5.5 The Cumulative Hazard Function

The **cumulative hazard function** $H(t)$ is defined as:

$$H(t) \stackrel{\text{def}}{=} \int_{u=0}^{t} h(u)du$$

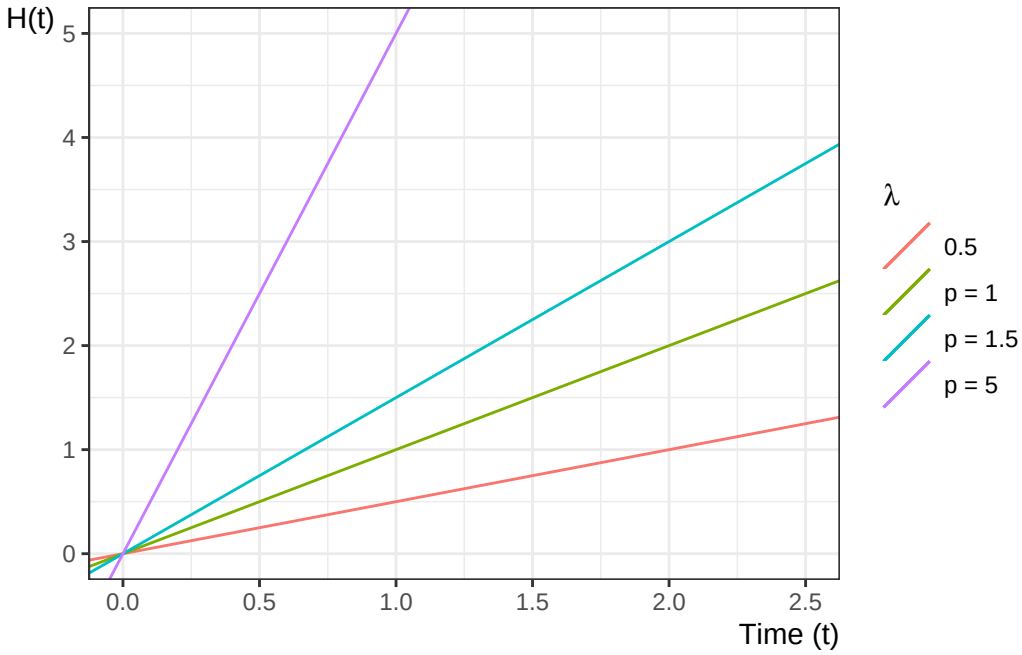
As we will see below, $H(t)$ is tractable to estimate, and we can then derive an estimate the hazard function using an approximate derivative of the estimated cumulative hazard.

Example: exponential distribution

The cumulative hazard function of the exponential distribution family of models is:

$$H(t) = \mathbb{1}_{t \geq 0} \cdot \lambda t$$

Here are some examples of exponential cumulative hazard functions:



0.30.5.6 Some Key Mathematical Relationships among Survival Concepts

$$\begin{aligned}
S(t) &= 1 - F(t) \\
&= \exp\{-H(t)\} \\
S'(t) &= -f(t) \\
H(t) &= -\log\{S(t)\} \\
H'(t) &= h(t) \\
h(t) &= \frac{f(t)}{S(t)} \\
f(t) &= h(t) \cdot S(t)
\end{aligned}$$

Some proofs (others left as exercises):

$$\begin{aligned}
S'(t) &= \frac{d}{dt}(1 - F(t)) \\
&= -F'(t) \\
&= -f(t)
\end{aligned}$$

$$\begin{aligned}
\frac{d}{dt}\log\{S(t)\} &= \frac{S'(t)}{S(t)} \\
&= -\frac{f(t)}{S(t)} \\
&= -h(t)
\end{aligned}$$

$$\begin{aligned}
H(t) &\stackrel{\text{def}}{=} \int_{u=0}^{t} h(u)du \\
&= \int_0^t -\frac{d}{du}\log\{S(u)\}\, du \\
&= [-\log\{S(u)\}]_{u=0}^{u=t} \\
&= [\log\{S(u)\}]_{u=t}^{u=0} \\
&= \log\{S(0)\} - \log\{S(t)\} \\
&= \log\{1\} - \log\{S(t)\} \\
&= 0 - \log\{S(t)\} \\
&= -\log\{S(t)\}
\end{aligned}$$

Equivalently:

$$S(t) = \exp\{-H(t)\}$$

0.30.5.7 Example: Time to death the US in 2004

Daily hazard rates for US Females in 2004

The first day is the most dangerous:

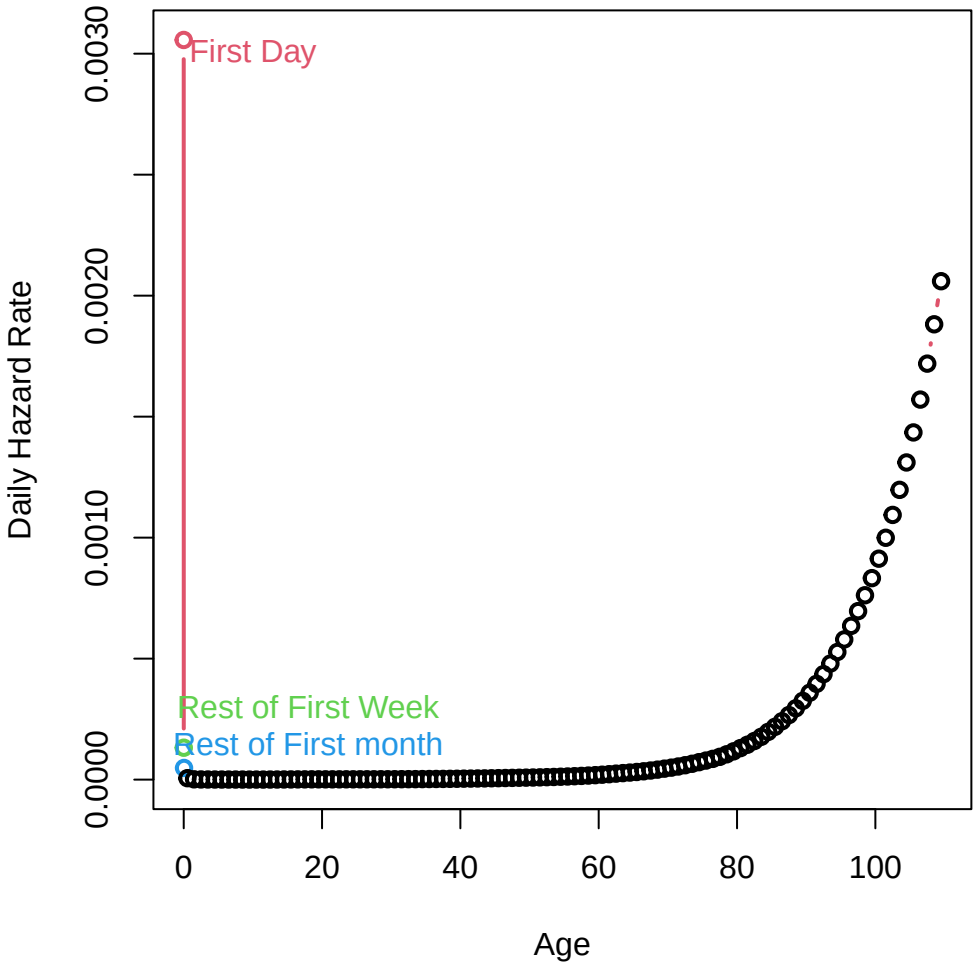


Figure 11: Daily Hazard Rates in 2004 for US Females

Daily hazard rates for US Males and Females in 2004

Exercise: hypothesize why these curves differ where they do?

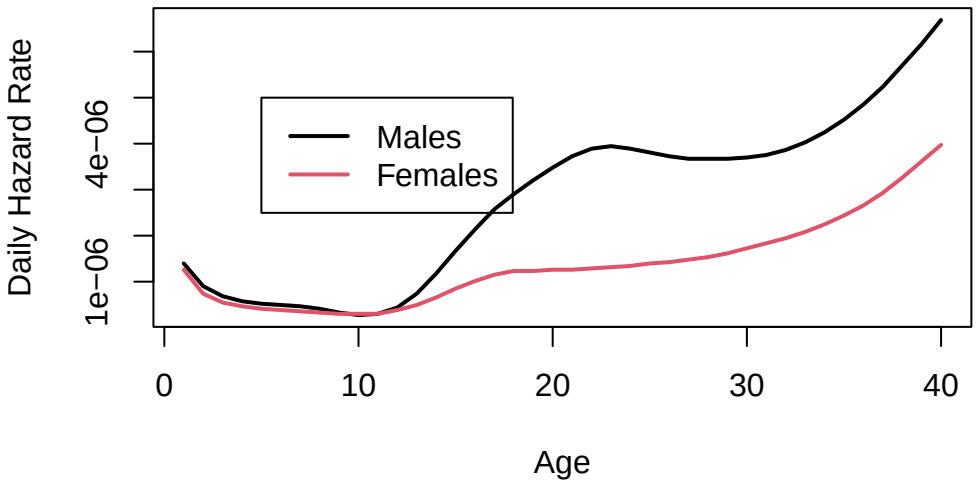


Figure 12: Daily Hazard Rates in 2004 for US Males and Females 1-40

Survival curve for US females

Exercise: compare and contrast this curve with the corresponding hazard curve.

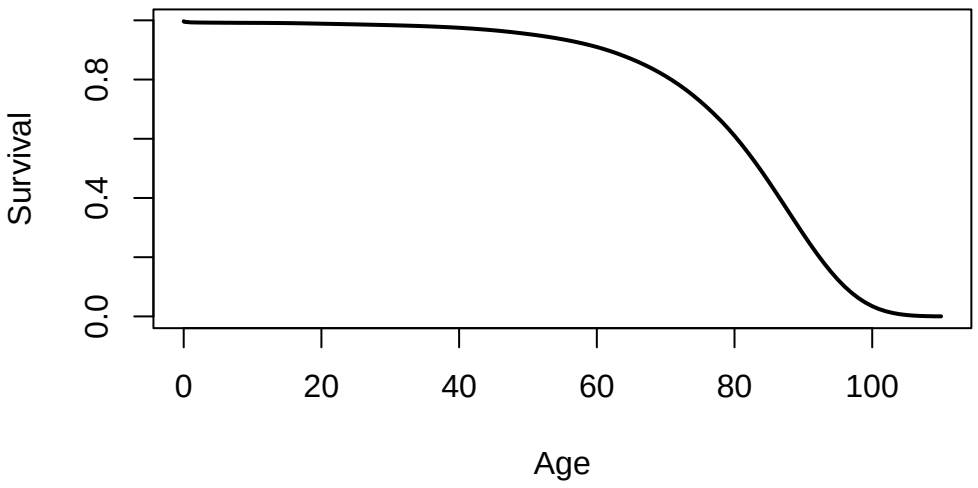


Figure 13: Survival Curve in 2004 for US Females

0.30.5.8 Likelihood with censoring *

> **Note**
>
> This subsection was not presented in class in 2023; it is not necessary to understand for the qualifying exam.

If an event time T is observed exactly as $T = t$, then the likelihood of that observation is just its probability density function:

$$\begin{aligned}
\mathcal{L}(t) &= p(T = t) \\
&\stackrel{\text{def}}{=} f_T(t) \\
&= h_T(t) S_T(t) \\
\ell(t) &\stackrel{\text{def}}{=} \log\{\mathcal{L}(t)\} \\
&= \log\{h_T(t) S_T(t)\} \\
&= \log\{h_T(t)\} + \log\{S_T(t)\} \\
&= \log\{h_T(t)\} - H_T(t)
\end{aligned}$$

If instead the event time T is censored and only known to be after time y, then the likelihood of that censored observation is instead the survival function evaluated at the censoring time:

$$\begin{aligned}\mathcal{L}(y) &= p_T(T > y) \\ &\stackrel{\text{def}}{=} S_T(y) \\ \ell(y) &\stackrel{\text{def}}{=} \log\{\mathcal{L}(y)\} \\ &= \log\{S(y)\} \\ &= -H(y)\end{aligned}$$

What's written above is incomplete. We also observed whether or not the observation was censored. Let C denote the time when censoring would occur (if the event did not occur first); let $f_C(y)$ and $S_C(y)$ be the corresponding density and survival functions for the censoring event.

Let Y denote the time when observation ended (either by censoring or by the event of interest occurring), and let D be an indicator variable for the event occurring at Y (so $D = 0$ represents a censored observation and $D = 1$ represents an uncensored observation). In other words, let $Y \stackrel{\text{def}}{=} \min(T, C)$ and $D \stackrel{\text{def}}{=} \mathbb{1}\{T <= C\}$.

Then the complete likelihood of the observed data (Y, D) is:

$$\begin{aligned}\mathcal{L}(y, d) &= p(Y = y, D = d) \\ &= [p(T = y, C > y)]^d \cdot [p(T > y, C = y)]^{1-d}\end{aligned}$$

Typically, survival analyses assume that C and T are mutually independent; this assumption is called "non-informative" censoring.

Then the joint likelihood $p(Y, D)$ factors into the product $p(Y), p(D)$, and the likelihood reduces to:

$$\begin{aligned}\mathcal{L}(y, d) &= [p(T = y, C > y)]^d \cdot [p(T > y, C = y)]^{1-d} \\ &= [p(T = y)p(C > y)]^d \cdot [p(T > y)p(C = y)]^{1-d} \\ &= [f_T(y)S_C(y)]^d \cdot [S(y)f_C(y)]^{1-d} \\ &= [f_T(y)^d S_C(y)^d] \cdot [S_T(y)^{1-d} f_C(y)^{1-d}] \\ &= (f_T(y)^d \cdot S_T(y)^{1-d}) \cdot (f_C(y)^{1-d} \cdot S_C(y)^d)\end{aligned}$$

The corresponding log-likelihood is:

$$\begin{aligned}
\ell(y,d) &= \log\{\mathcal{L}(y,d)\} \\
&= \log\left\{\left(f_T(y)^d \cdot S_T(y)^{1-d}\right) \cdot \left(f_C(y)^{1-d} \cdot S_C(y)^d\right)\right\} \\
&= \log\left\{f_T(y)^d \cdot S_T(y)^{1-d}\right\} + \log\left\{f_C(y)^{1-d} \cdot S_C(y)^d\right\}
\end{aligned}$$

Let

- θ_T represent the parameters of $p_T(t)$,
- θ_C represent the parameters of $p_C(c)$,
- $\theta = (\theta_T, \theta_C)$ be the combined vector of all parameters.

Then corresponding score function is:

$$\begin{aligned}
\ell'(y,d) &= \frac{d}{d\theta}\left[\log\left\{f_T(y)^d \cdot S_T(y)^{1-d}\right\} + \log\left\{f_C(y)^{1-d} \cdot S_C(y)^d\right\}\right] \\
&= \left(\frac{d}{d\theta}\log\left\{f_T(y)^d \cdot S_T(y)^{1-d}\right\}\right) + \left(\frac{d}{d\theta}\log\left\{f_C(y)^{1-d} \cdot S_C(y)^d\right\}\right)
\end{aligned}$$

As long as θ_C and θ_T don't share any parameters, then if censoring is non-informative, the partial derivative with respect to θ_T is:

$$\begin{aligned}
\ell'_{\theta_T}(y,d) &\stackrel{\text{def}}{=} \frac{d}{d\theta_T}\ell(y,d) \\
&= \left(\frac{d}{d\theta_T}\log\left\{f_T(y)^d \cdot S_T(y)^{1-d}\right\}\right) + \left(\frac{d}{d\theta_T}\log\left\{f_C(y)^{1-d} \cdot S_C(y)^d\right\}\right) \\
&= \left(\frac{d}{d\theta_T}\log\left\{f_T(y)^d \cdot S_T(y)^{1-d}\right\}\right) + 0 \\
&= \frac{d}{d\theta_T}\log\left\{f_T(y)^d \cdot S_T(y)^{1-d}\right\}
\end{aligned}$$

Thus, the MLE for θ_T won't depend on θ_C, and we can ignore the distribution of C when estimating the parameters of $f_T(t) = p(T = t)$.

Then:

$$\begin{aligned}
\mathcal{L}(y,d) &= f_T(y)^d \cdot S_T(y)^{1-d} \\
&= \left(h_T(y)^d S_T(y)^d\right) \cdot S_T(y)^{1-d} \\
&= h_T(y)^d \cdot S_T(y)^d \cdot S_T(y)^{1-d} \\
&= h_T(y)^d \cdot S_T(y) \\
&= S_T(y) \cdot h_T(y)^d
\end{aligned}$$

That is, if the event occurred at time y (i.e., if $d = 1$), then the likelihood of $(Y, D) = (y, d)$ is equal to the hazard function at y times the survival function at y. Otherwise, the likelihood is equal to just the survival function at y.

The corresponding log-likelihood is:

$$\begin{aligned}
\ell(y, d) &= \log\{\mathcal{L}(y, d)\} \\
&= \log\{S_T(y) \cdot h_T(y)^d\} \\
&= \log\{S_T(y)\} + \log\{h_T(y)^d\} \\
&= \log\{S_T(y)\} + d \cdot \log\{h_T(y)\} \\
&= -H_T(y) + d \cdot \log\{h_T(y)\}
\end{aligned}$$

In other words, the log-likelihood contribution from a single observation $(Y, D) = (y, d)$ is equal to the negative cumulative hazard at y, plus the log of the hazard at y if the event occurred at time y.

> **Note**
>
> End of extra section.

0.31 Parametric Models for Time-to-Event Outcomes

0.31.1 Exponential Distribution

- The exponential distribution is the base distribution for survival analysis.
- The distribution has a constant hazard λ
- The mean survival time is λ^{-1}

0.31.1.1 Mathematical details of exponential distribution

$$\begin{aligned}
f(t) &= \lambda e^{-\lambda t} \\
E(t) &= \lambda^{-1} \\
Var(t) &= \lambda^{-2} \\
F(t) &= 1 - e^{-\lambda x} \\
S(t) &= e^{-\lambda x} \\
\ln(S(t)) &= -\lambda x \\
h(t) &= -\frac{f(t)}{S(t)} = -\frac{\lambda e^{-\lambda t}}{e^{-\lambda t}} = \lambda
\end{aligned}$$

0.31.1.2 Estimation of λ

- Suppose we have m exponential survival times of $t_1, t_2, \ldots, t_m$ and k right-censored values at $u_1, u_2, \ldots, u_k$.
- A survival time of $t_i = 10$ means that subject i died at time 10. A right-censored time $u_i = 10$ means that at time 10, subject i was still alive and that we have no further follow-up.
- For the moment we will assume that the survival distribution is exponential and that all the subjects have the same parameter λ.

We have m exponential survival times of $t_1, t_2, \ldots, t_m$ and k right-censored values at $u_1, u_2, \ldots, u_k$. The log-likelihood of an observed survival time t_i is

$$\log\left\{\lambda \mathrm{e}^{-\lambda t_i}\right\} = \log\{\lambda\} - \lambda t_i$$

and the likelihood of a censored value is the probability of that outcome (survival greater than u_j) so the log-likelihood is

$$\log\left\{\lambda \mathrm{e}^{u_j}\right\} = -\lambda u_j.$$

Let $T = \sum t_i$ and $U = \sum u_j$. Then:

$$\begin{aligned}
\ell(\lambda) &= \sum_{i=1}^{m} (\ln \lambda - \lambda t_i) + \sum_{j=1}^{k} (-\lambda u_j) \\
&= m \ln \lambda - (T + U)\lambda \\
\ell'(\lambda) &= m\lambda^{-1} - (T + U) \\
\hat{\lambda} &= \frac{m}{T + U} \\
\ell'' &= -m/\lambda^2 \\
&< 0 \\
\hat{E}[T] &= \hat{\lambda}^{-1} \\
&= \frac{T + U}{m}
\end{aligned}$$

0.31.1.3 Fisher Information and Standard Error

$$\begin{aligned} E[-\ell''] &= m/\lambda^2 \\ \text{Var}\left(\widehat{\lambda}\right) &\approx \left(E[-\ell'']\right)^{-1} \\ &= \lambda^2/m \\ \text{SE}\left(\widehat{\lambda}\right) &= \sqrt{\text{Var}\left(\widehat{\lambda}\right)} \\ &\approx \lambda/\sqrt{m} \end{aligned}$$

$\widehat{\lambda}$ depends on the censoring times of the censored observations, but $\text{Var}\left(\widehat{\lambda}\right)$ only depends on the number of uncensored observations, m, and not on the number of censored observations (k).

0.31.1.4 Other Parametric Survival Distributions

- Any density on $[0, \infty)$ can be a survival distribution, but the most useful ones are all skew right.
- The commonest generalization of the exponential is the Weibull.
- Other common choices are the gamma, log-normal, log-logistic, Gompertz, inverse Gaussian, and Pareto.
- Most of what we do going forward is non-parametric or semi-parametric, but sometimes these parametric distributions provide a useful approach.

0.31.2 Weibull Distribution

$$\begin{aligned} p(t) &= \alpha\lambda x^{\alpha-1}\mathrm{e}^{-\lambda x^\alpha} \\ h(t) &= \alpha\lambda x^{\alpha-1} \\ S(t) &= \mathrm{e}^{-\lambda x^\alpha} \\ E(T) &= \Gamma(1+1/\alpha)\cdot\lambda^{-1/\alpha} \end{aligned}$$

When $\alpha = 1$ this is the exponential. When $\alpha > 1$ the hazard is increasing and when $\alpha < 1$ the hazard is decreasing. This provides more flexibility than the exponential.

We will see more of this distribution later.

0.32 Nonparametric Survival Analysis

0.32.1 Basic ideas

- Mostly, we work without a parametric model.
- The first task is to estimate a survival function from data listing survival times, and censoring times for censored data.
- For example one patient may have relapsed at 10 months. Another might have been followed for 32 months without a relapse having occurred (censored).
- The minimum information we need for each patient is a time and a censoring variable which is 1 if the event occurred at the indicated time and 0 if this is a censoring time.

0.33 Example: clinical trial for pediatric acute leukemia

0.33.1 Overview of study

This is from a clinical trial in 1963 for 6-MP treatment vs. placebo for Acute Leukemia in 42 children.

- Pairs of children:
  - matched by remission status at the time of treatment (`remstat`: `1` = partial, `2` = complete)
  - randomized to 6-MP (exit times in `t2`) or placebo (exit times in `t1`)
- Followed until relapse or end of study.
- All of the placebo group relapsed, but some of the 6-MP group were censored (which means they were still in remission); indicated by `relapse` variable (`0` = censored, `1` = relapse).
- 6-MP = 6-Mercaptopurine (Purinethol) is an anti-cancer ("antineoplastic" or "cytotoxic") chemotherapy drug used currently for Acute lymphoblastic leukemia (ALL). It is classified as an antimetabolite.

0.33.2 Study design

Clinical trial in 1963 for 6-MP treatment vs. placebo for Acute Leukemia in 42 children. Pairs of children matched by remission status at the time of treatment (1 = partial or 2 = complete) and randomized to 6-MP or placebo. Followed until relapse or end of study. All of the placebo group relapsed, but some of the 6-MP group were censored.

```
library(KMsurv)
data(drug6mp)
drug6mp |> tibble() |> print()
```

```
# A tibble: 21 x 5
    pair remstat    t1    t2 relapse
   <int>   <int> <int> <int>   <int>
 1     1       1     1    10       1
 2     2       2    22     7       1
 3     3       2     3    32       0
 4     4       2    12    23       1
 5     5       2     8    22       1
 6     6       1    17     6       1
 7     7       2     2    16       1
 8     8       2    11    34       0
 9     9       2     8    32       0
10    10       2    12    25       0
# i 11 more rows
```

0.33.3 Data documentation for `drug6mp`

```
library(printr) # inserts help-file output into markdown output
library(KMsurv)
?drug6mp
```

```
data from Section 1.2

Description:

     The 'drug6mp' data frame has 21 rows and 5 columns.

Format:
```

```
This data frame contains the following columns:

pair pair number

remstat Remission status at randomization (1=partial, 2=complete)

t1 Time to relapse for placebo patients, months

t2 Time to relapse for 6-MP patients, months

relapse Relapse indicator (0=censored, 1=relapse) for 6-MP
     patients
```

0.33.4 Descriptive Statistics

- The average time in each group is not useful. Some of the 6-MP patients have not relapsed at the time recorded, while all of the placebo patients have relapsed.
- The median time is not really useful either because so many of the 6-MP patients have not relapsed (12/21).
- Both are biased down in the 6-MP group. Remember that lower times are worse since they indicate sooner recurrence.
- We can compute the average hazard rate, which is the estimate of the exponential parameter: number of relapses divided by the sum of the times.
- For the placebo, that is just the reciprocal of the mean time = $1/8.667 = 0.115$.
- For the 6-MP group this is $9/359 = 0.025$
- The estimated average hazard in the placebo group is 4.6 times as large (if the hazard is constant over time).

0.34 The Kaplan-Meier Product Limit Estimator

- The estimated survival function for the placebo patients is easy to compute. For any time t in months, $S(t)$ is the fraction of patients with times greater than t.
- For the 6-MP patients, we cannot ignore the censored data because we know that the time to relapse is greater than the censoring time.
- For any time t in months, we know that 6-MP patients with times greater than t have not relapsed, and those with relapse time less than t have relapsed, but we don't know if patients with censored time less than t have relapsed or not.
- The procedure we usually use is the Kaplan-Meier product-limit estimator of the survival function.

- The Kaplan-Meier estimator is a step function (like the empirical cdf), which changes value only at the event times, not at the censoring times.
- At each event time t, we compute the at-risk group size Y, which is all those observations whose event time or censoring time is at least t.
- If d of the observations have an event time (not a censoring time) of t, then the group of survivors immediately following time t is reduced by the fraction

$$\frac{Y-d}{Y} = 1 - \frac{d}{Y}$$

If the event times are t_i with events per time of d_i $(1 \leq i \leq k)$, then

$$\hat{S}(t) = \prod_{t_i<t} [1 - d_i/Y_i]$$

where Y_i is the set of observations whose time (event or censored) is $\geq t_i$, the group at risk at time t_i.

If there are no censored data, and there are n data points, then just after (say) the third event time

$$\begin{aligned}
\hat{S}(t) &= \prod_{t_i<t} [1 - d_i/Y_i] \\
&= [\frac{n-d_1}{n}][\frac{n-d_1-d_2}{n-d_1}][\frac{n-d_1-d_2-d_3}{n-d_1-d_2}] \\
&= \frac{n-d_1-d_2-d_3}{n} \\
&= 1 - \frac{d_1+d_2+d_3}{n} \\
&= 1 - \hat{F}(t)
\end{aligned}$$

where $\hat{F}(t)$ is the usual empirical CDF estimate.

0.34.1 Kaplan-Meier curve for `drug6mp` data

Here is the Kaplan-Meier estimated survival curve for the patients who received 6-MP in the `drug6mp` dataset (we will see code to produce figures like this one shortly):

0.34.2 Kaplan-Meier calculations

Let's compute these estimates and build the chart by hand:

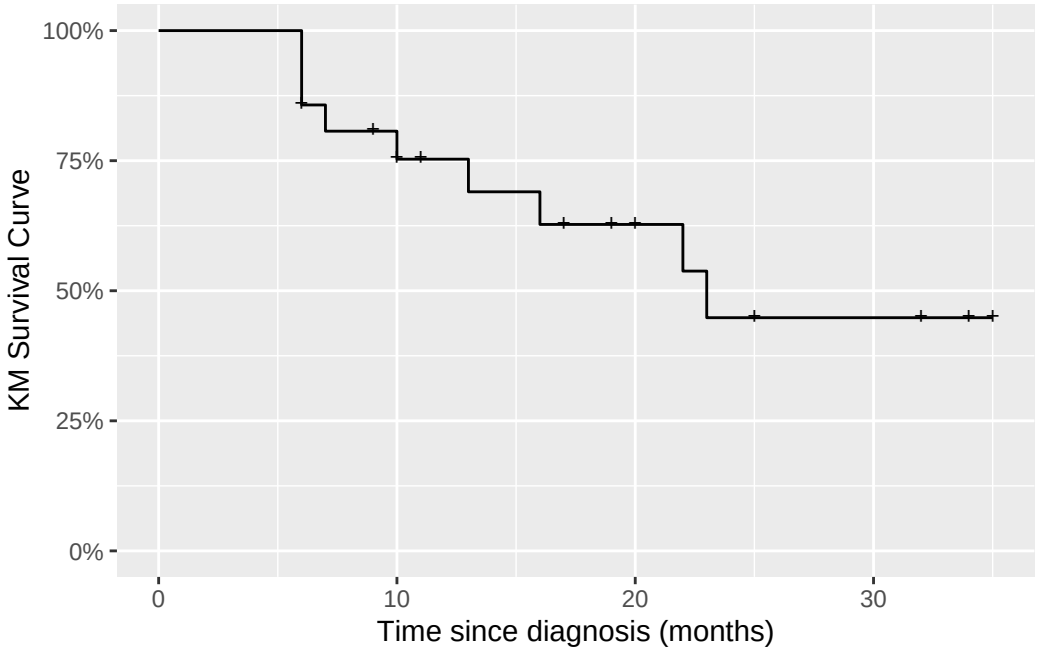


Figure 14: Kaplan-Meier Survival Curve for 6-MP Patients

```r
library(KMsurv)
library(dplyr)
data(drug6mp)

drug6mp.v2 =
  drug6mp |>
  as_tibble() |>
  mutate(
    remstat = remstat |>
      case_match(
        1 ~ "partial",
        2 ~ "complete"
      ),
    # renaming to "outcome" while relabeling is just a style choice:
    outcome = relapse |>
      case_match(
        0 ~ "censored",
        1 ~ "relapsed"
      )
  )
```

```
km.6mp =
  drug6mp.v2 |>
  summarize(
    .by = t2,
    Relapses = sum(outcome == "relapsed"),
    Censored = sum(outcome == "censored")) |>
  # here we add a start time row, so the graph starts at time 0:
  bind_rows(
    tibble(
      t2 = 0,
      Relapses = 0,
      Censored = 0)
  ) |>
  # sort in time order:
  arrange(t2) |>
  mutate(
    Exiting = Relapses + Censored,
    `Study Size` = sum(Exiting),
    Exited = cumsum(Exiting) |> dplyr::lag(default = 0),
    `At Risk` = `Study Size` - Exited,
    Hazard = Relapses / `At Risk`,
    `KM Factor` = 1 - Hazard,
    `Cumulative Hazard` = cumsum(`Hazard`),
    `KM Survival Curve` = cumprod(`KM Factor`)
  )

library(pander)
pander(km.6mp)
```

t2	Relapses	Censored	Exiting	Study Size	Exited	At Risk	Hazard	KM Factor	Cumulative Hazard	KM Survival Curve
0	0	0	0	21	0	21	0	1	0	1
6	3	1	4	21	0	21	0.1429	0.8571	0.1429	0.8571
7	1	0	1	21	4	17	0.05882	0.9412	0.2017	0.8067
9	0	1	1	21	5	16	0	1	0.2017	0.8067
10	1	1	2	21	6	15	0.06667	0.9333	0.2683	0.7529
11	0	1	1	21	8	13	0	1	0.2683	0.7529
13	1	0	1	21	9	12	0.08333	0.9167	0.3517	0.6902
16	1	0	1	21	10	11	0.09091	0.9091	0.4426	0.6275
17	0	1	1	21	11	10	0	1	0.4426	0.6275

t2	Relapses	Censored	Exiting	Study Size	Exited	At Risk	Hazard	KM Factor	Cumulative Hazard	KM Survival Curve
19	0	1	1	21	12	9	0	1	0.4426	0.6275
20	0	1	1	21	13	8	0	1	0.4426	0.6275
22	1	0	1	21	14	7	0.1429	0.8571	0.5854	0.5378
23	1	0	1	21	15	6	0.1667	0.8333	0.7521	0.4482
25	0	1	1	21	16	5	0	1	0.7521	0.4482
32	0	2	2	21	17	4	0	1	0.7521	0.4482
34	0	1	1	21	19	2	0	1	0.7521	0.4482
35	0	1	1	21	20	1	0	1	0.7521	0.4482

0.34.2.1 Summary

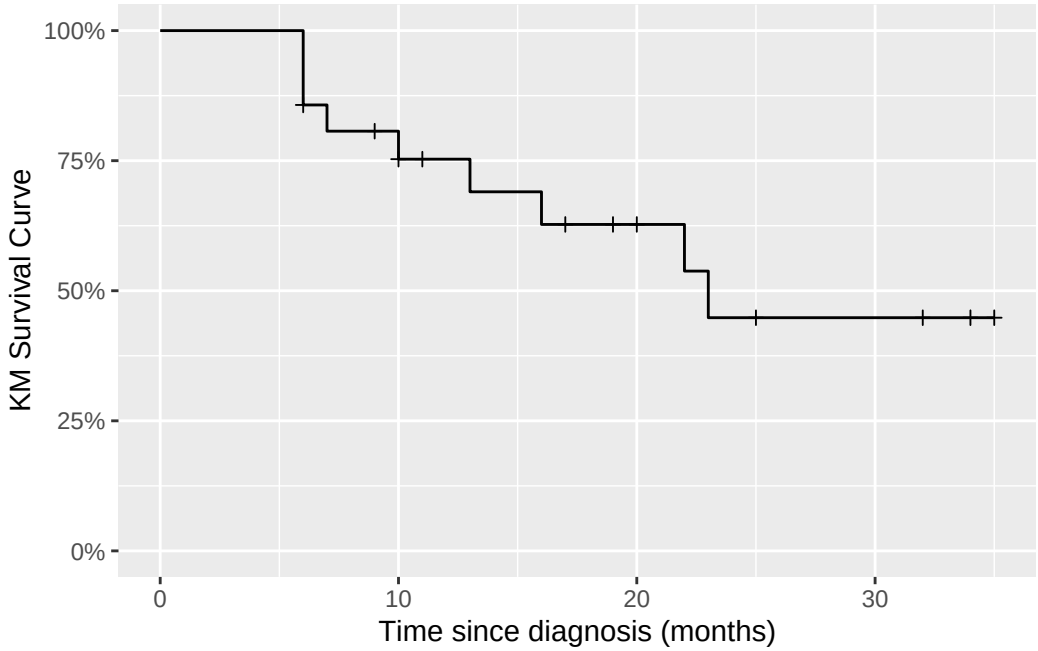
For the 6-MP patients at time 6 months, there are 21 patients at risk. At $t = 6$ there are 3 relapses and 1 censored observations.

The Kaplan-Meier factor is $(21 - 3)/21 = 0.857$. The number at risk for the next time ($t = 7$) is $21 - 3 - 1 = 17$.

At time 7 months, there are 17 patients at risk. At $t = 7$ there is 1 relapse and 0 censored observations. The Kaplan-Meier factor is $(17 - 1)/17 = 0.941$. The Kaplan Meier estimate is $0.857 \times 0.941 = 0.807$. The number at risk for the next time ($t = 9$) is $17 - 1 = 16$.

Now, let's graph this estimated survival curve using `ggplot()`:

```
library(ggplot2)
conflicts_prefer(dplyr::filter)
km.6mp |>
  ggplot(aes(x = t2, y = `KM Survival Curve`)) +
  geom_step() +
  geom_point(data = km.6mp |> filter(Censored > 0), shape = 3) +
  expand_limits(y = c(0,1), x = 0) +
  xlab('Time since diagnosis (months)') +
  ylab("KM Survival Curve") +
  scale_y_continuous(labels = scales::percent)
```



0.35 Using the `survival` package in R

We don't have to do these calculations by hand every time; the `survival` package and several others have functions available to automate many of these tasks (full list: https://cran.r-project.org/web/views/Survival.html).

0.35.1 The `Surv` function

To use the `survival` package, the first step is telling R how to combine the exit time and exit reason (censoring versus event) columns. The `Surv()` function accomplishes this task.

0.35.1.1 Example: `Surv()` with `drug6mp` data

```
library(survival)
drug6mp.v3 =
  drug6mp.v2 |>
  mutate(
    surv2 = Surv(
      time = t2,
```

```
7         event = (outcome == "relapsed")))
8
9  print(drug6mp.v3)
```

```
# A tibble: 21 x 7
    pair remstat     t1    t2 relapse outcome   surv2
   <int> <chr>    <int> <int>   <int> <chr>     <Surv>
 1     1 partial      1    10       1 relapsed    10
 2     2 complete    22     7       1 relapsed     7
 3     3 complete     3    32       0 censored    32+
 4     4 complete    12    23       1 relapsed    23
 5     5 complete     8    22       1 relapsed    22
 6     6 partial     17     6       1 relapsed     6
 7     7 complete     2    16       1 relapsed    16
 8     8 complete    11    34       0 censored    34+
 9     9 complete     8    32       0 censored    32+
10    10 complete    12    25       0 censored    25+
# i 11 more rows
```

The output of `Surv()` is a vector of objects with class `Surv`. When we print this vector:

- observations where the event was observed are printed as the event time (for example, `surv2 = 10` on line 1)
- observations where the event was right-censored are printed as the censoring time with a plus sign (`+`; for example, `surv2 = 32+` on line 3).

0.35.2 The `survfit` function

Once we have constructed our `Surv` variable, we can calculate the Kaplan-Meier estimate of the survival curve using the `survfit()` function.

> **Note**
>
> The documentation for `?survfit` isn't too helpful; the `survfit.formula` documentation is better.

0.35.2.1 Example: `survfit()` with `drug6mp` data

Here we use `survfit()` to create a `survfit` object, which contains the Kaplan-Meier estimate:

```
drug6mp.km_model = survfit(
  formula = surv2 ~ 1,
  data = drug6mp.v3)
```

`print.survfit()` just gives some summary statistics:

```
print(drug6mp.km_model)
```

```
Call: survfit(formula = surv2 ~ 1, data = drug6mp.v3)

      n events median 0.95LCL 0.95UCL
[1,] 21      9     23      16      NA
```

`summary.survfit()` shows us the underlying Kaplan-Meier table:

```
summary(drug6mp.km_model)
```

```
Call: survfit(formula = surv2 ~ 1, data = drug6mp.v3)

 time n.risk n.event survival std.err lower 95% CI upper 95% CI
    6     21       3    0.857  0.0764        0.720        1.000
    7     17       1    0.807  0.0869        0.653        0.996
   10     15       1    0.753  0.0963        0.586        0.968
   13     12       1    0.690  0.1068        0.510        0.935
   16     11       1    0.627  0.1141        0.439        0.896
   22      7       1    0.538  0.1282        0.337        0.858
   23      6       1    0.448  0.1346        0.249        0.807
```

`summary.survfit()` shows us the underlying Kaplan-Meier table:

```
summary(drug6mp.km_model)
```

```
Call: survfit(formula = surv2 ~ 1, data = drug6mp.v3)

 time n.risk n.event survival std.err lower 95% CI upper 95% CI
    6     21       3    0.857  0.0764        0.720        1.000
    7     17       1    0.807  0.0869        0.653        0.996
   10     15       1    0.753  0.0963        0.586        0.968
   13     12       1    0.690  0.1068        0.510        0.935
```

```
   16     11       1    0.627  0.1141        0.439        0.896
   22      7       1    0.538  0.1282        0.337        0.858
   23      6       1    0.448  0.1346        0.249        0.807
```

We can specify which time points we want using the `times` argument:

```
summary(
  drug6mp.km_model,
  times = c(0, drug6mp.v3$t2))
```

```
Call: survfit(formula = surv2 ~ 1, data = drug6mp.v3)

 time n.risk n.event survival std.err lower 95% CI upper 95% CI
    0     21       0    1.000  0.0000        1.000        1.000
    6     21       3    0.857  0.0764        0.720        1.000
    6     21       0    0.857  0.0764        0.720        1.000
    6     21       0    0.857  0.0764        0.720        1.000
    6     21       0    0.857  0.0764        0.720        1.000
    7     17       1    0.807  0.0869        0.653        0.996
    9     16       0    0.807  0.0869        0.653        0.996
   10     15       1    0.753  0.0963        0.586        0.968
   10     15       0    0.753  0.0963        0.586        0.968
   11     13       0    0.753  0.0963        0.586        0.968
   13     12       1    0.690  0.1068        0.510        0.935
   16     11       1    0.627  0.1141        0.439        0.896
   17     10       0    0.627  0.1141        0.439        0.896
   19      9       0    0.627  0.1141        0.439        0.896
   20      8       0    0.627  0.1141        0.439        0.896
   22      7       1    0.538  0.1282        0.337        0.858
   23      6       1    0.448  0.1346        0.249        0.807
   25      5       0    0.448  0.1346        0.249        0.807
   32      4       0    0.448  0.1346        0.249        0.807
   32      4       0    0.448  0.1346        0.249        0.807
   34      2       0    0.448  0.1346        0.249        0.807
   35      1       0    0.448  0.1346        0.249        0.807
```

```
?summary.survfit
```

```
Summary of a Survival Curve
```

```
Description:

     Returns a list containing the survival curve, confidence limits
     for the curve, and other information.

Usage:

     ## S3 method for class 'survfit'
     summary(object, times, censored=FALSE, scale=1,
       extend=FALSE, rmean=getOption('survfit.rmean'), ...)

Arguments:

  object: the result of a call to the 'survfit' function.

   times: vector of times; the returned matrix will contain 1 row for
          each time.  The vector will be sorted into increasing order;
          missing values are not allowed.  If 'censored=T', the default
          'times' vector contains all the unique times in 'fit',
          otherwise the default 'times' vector uses only the event
          (death) times.

censored: logical value: should the censoring times be included in the
          output?  This is ignored if the 'times' argument is present.

   scale: numeric value to rescale the survival time, e.g., if the
          input data to 'survfit' were in days, 'scale = 365.25' would
          scale the output to years.

  extend: logical value: if TRUE, prints information for all specified
          'times', even if there are no subjects left at the end of the
          specified 'times'.  This is only used if the 'times' argument
          is present.

   rmean: Show restricted mean: see 'print.survfit' for details

     ...: for future methods
```

0.35.3 Plotting estimated survival functions

We can plot `survfit` objects with `plot()`, `autoplot()`, or `ggsurvplot()`:

```
library(ggfortify)
autoplot(drug6mp.km_model)
```

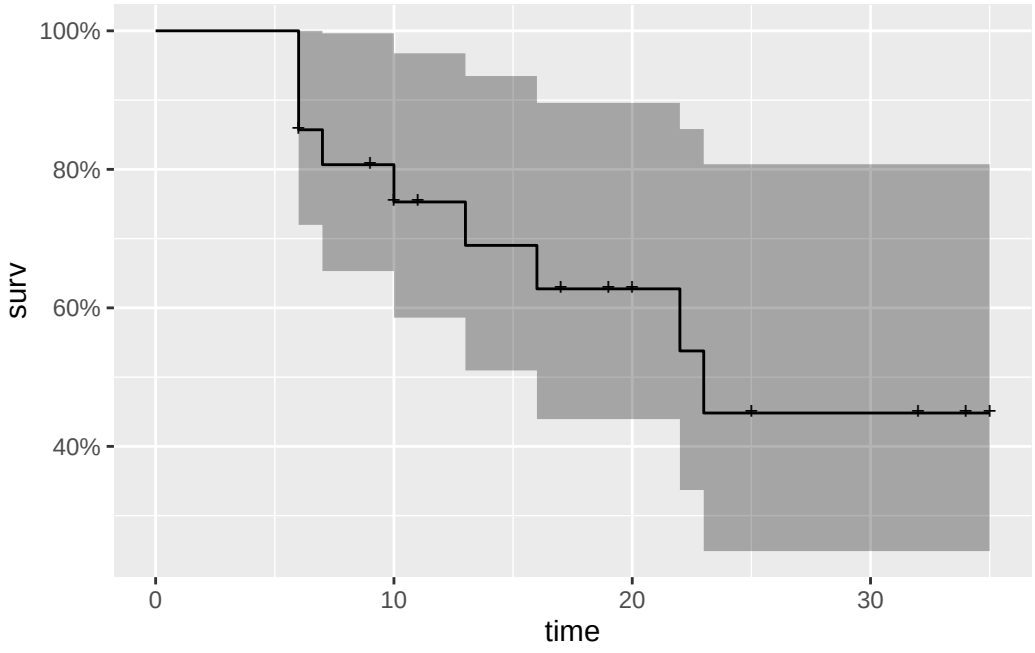


Figure 15: Kaplan-Meier Survival Curve for 6-MP Patients

```
# not shown:
# plot(drug6mp.km_model)

# library(survminer)
# ggsurvplot(drug6mp.km_model)
```

0.35.3.1 quantiles of survival curve

We can extract quantiles with `quantile()`:

```
drug6mp.km_model |>
  quantile(p = c(.25, .5)) |>
  as_tibble() |>
  mutate(p = c(.25, .5)) |>
  relocate(p, .before = everything())
```

p	quantile	lower	upper
0.25	13	6	NA
0.50	23	16	NA

0.35.4 Two-sample tests

0.35.4.1 The `survdiff` function

```
?survdiff
```

```
Test Survival Curve Differences

Description:

     Tests if there is a difference between two or more survival curves
     using the G-rho family of tests, or for a single curve against a
     known alternative.

Usage:

     survdiff(formula, data, subset, na.action, rho=0, timefix=TRUE)
```

0.35.4.2 Example: `survdiff()` with `drug6mp` data

Now we are going to compare the placebo and 6-MP data. We need to reshape the data to make it usable with the standard `survival` workflow:

```
library(survival)

drug6mp.v4 =
  drug6mp.v3 |>
  select(pair, remstat, t1, t2, outcome) |>
  # here we are going to change the data from a wide format to long:
  pivot_longer(
    cols = c(t1, t2),
    names_to = "treatment",
    values_to = "exit_time") |>
  mutate(
    treatment = treatment |>
```

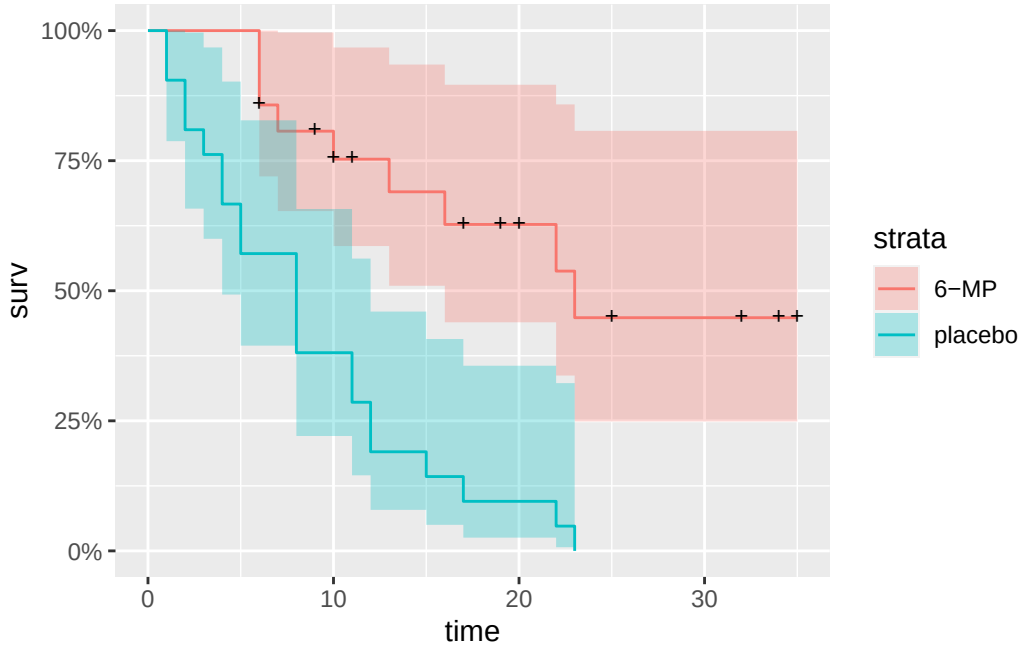
```
      case_match(
        "t1" ~ "placebo",
        "t2" ~ "6-MP"
      ),
    outcome = if_else(
      treatment == "placebo",
      "relapsed",
      outcome
    ),
    surv = Surv(
      time = exit_time,
      event = (outcome == "relapsed"))
  )
```

Using this long data format, we can fit a Kaplan-Meier curve for each treatment group simultaneously:

```
drug6mp.km_model2 =
   survfit(
  formula = surv ~ treatment,
  data = drug6mp.v4)
```

We can plot the curves in the same graph:

```
drug6mp.km_model2 |> autoplot()
```



We can also perform something like a t-test, where the null hypothesis is that the curves are the same:

```
survdiff(
  formula = surv ~ treatment,
  data = drug6mp.v4)
```

```
Call:
survdiff(formula = surv ~ treatment, data = drug6mp.v4)

                  N Observed Expected (O-E)^2/E (O-E)^2/V
treatment=6-MP   21        9     19.3      5.46      16.8
treatment=placebo 21       21     10.7      9.77      16.8

 Chisq= 16.8  on 1 degrees of freedom, p= 4e-05
```

By default, `survdiff()` ignores any pairing, but we can use `strata()` to perform something similar to a paired t-test:

```
survdiff(
  formula = surv ~ treatment + strata(pair),
  data = drug6mp.v4)
```

```
Call:
survdiff(formula = surv ~ treatment + strata(pair), data = drug6mp.v4)

                   N Observed Expected (O-E)^2/E (O-E)^2/V
treatment=6-MP    21        9     16.5      3.41      10.7
treatment=placebo 21       21     13.5      4.17      10.7

 Chisq= 10.7  on 1 degrees of freedom, p= 0.001
```

Interestingly, accounting for pairing reduces the significant of the difference.

0.36 Example: Bone Marrow Transplant Data

(Copelan et al., 1991)

* Treatment
  - **allogeneic** (from a donor) **bone marrow transplant therapy**

* Inclusion criteria
  - **acute myeloid leukemia (AML)**
  - **acute lymphoblastic leukemia (ALL).**

* Possible intermediate events
  - **graft vs. host disease (GVHD)**: an immunological rejection response to the transplant
  - **platelet recovery**: a return of platelet count to normal levels.

One or the other, both in either order, or neither may occur.

End point events

- relapse of the disease
- death

Any or all of these events may be censored.

0.36.1 KMsurv::bmt data in R

```
library(printr) # inserts help-file output into markdown output
library(KMsurv)
?bmt
```

```
data from Section 1.3

Description:

     The 'bmt' data frame has 137 rows and 22 columns.

Format:

     This data frame contains the following columns:

     group Disease Group 1-ALL, 2-AML Low Risk, 3-AML High Risk

     t1 Time To Death Or On Study Time

     t2 Disease Free Survival Time (Time To Relapse, Death Or End Of
          Study)

     d1 Death Indicator 1-Dead 0-Alive

     d2 Relapse Indicator 1-Relapsed, 0-Disease Free

     d3 Disease Free Survival Indicator 1-Dead Or Relapsed, 0-Alive
          Disease Free)

     ta Time To Acute Graft-Versus-Host Disease

     da Acute GVHD Indicator 1-Developed Acute GVHD 0-Never Developed
          Acute GVHD)

     tc Time To Chronic Graft-Versus-Host Disease

     dc Chronic GVHD Indicator 1-Developed Chronic GVHD 0-Never
          Developed Chronic GVHD

     tp Time To Chronic Graft-Versus-Host Disease
```

```
     dp Platelet Recovery Indicator 1-Platelets Returned To Normal,
          0-Platelets Never Returned to Normal

     z1 Patient Age In Years

     z2 Donor Age In Years

     z3 Patient Sex: 1-Male, 0-Female

     z4 Donor Sex: 1-Male, 0-Female

     z5 Patient CMV Status: 1-CMV Positive, 0-CMV Negative

     z6 Donor CMV Status: 1-CMV Positive, 0-CMV Negative

     z7 Waiting Time to Transplant In Days

     z8 FAB: 1-FAB Grade 4 Or 5 and AML, 0-Otherwise

     z9 Hospital: 1-The Ohio State University, 2-Alferd , 3-St.
          Vincent, 4-Hahnemann

     z10 MTX Used as a Graft-Versus-Host- Prophylactic: 1-Yes 0-No

Source:

     Klein and Moeschberger (1997) _Survival Analysis Techniques for
     Censored and truncated data_, Springer.

Examples:

     data(bmt)
```

0.36.2 Analysis plan

- We concentrate for now on disease-free survival (`t2` and `d3`) for the three risk groups, ALL, AML Low Risk, and AML High Risk.
- We will construct the Kaplan-Meier survival curves, compare them, and test for differences.
- We will construct the cumulative hazard curves and compare them.
- We will estimate the hazard functions, interpret, and compare them.

0.36.3 Survival Function Estimate and Variance

$$\hat{S}(t) = \prod_{t_i<t} \left[1 - \frac{d_i}{Y_i}\right]$$

where Y_i is the group at risk at time t_i.

The estimated variance of $\hat{S}(t)$ is (Greenwood's formula)

$$\hat{\text{Var}}\left(\hat{S}(t)\right) = \hat{S}(t)^2 \sum_{t_i<t} \frac{d_i}{Y_i(Y_i - d_i)}$$

which we can use for confidence intervals for a survival function or a difference of survival functions.

Kaplan-Meier survival curves

```
library(KMsurv)
library(survival)
data(bmt)

bmt =
  bmt |>
  as_tibble() |>
  mutate(
    group =
      group |>
      factor(
        labels = c("ALL","Low Risk AML","High Risk AML")),
    surv = Surv(t2,d3))

km_model1 = survfit(
  formula = surv ~ group,
  data = bmt)
```

```
library(ggfortify)
autoplot(
  km_model1,
  conf.int = TRUE,
  ylab = "Pr(disease-free survival)",
  xlab = "Time since transplant (days)") +
  theme_bw() +
  theme(legend.position="bottom")
```

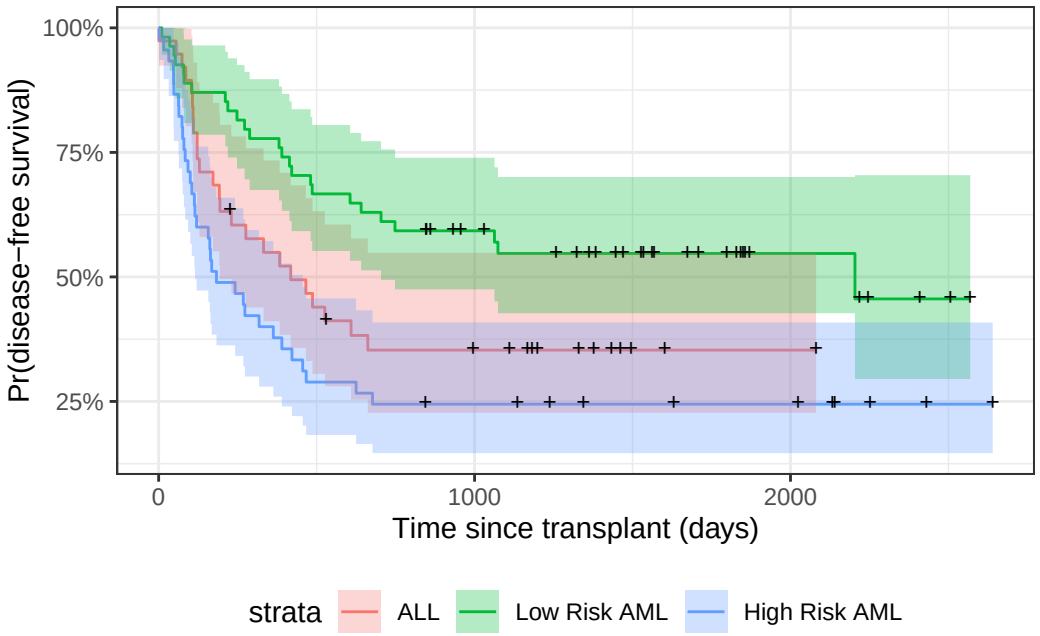


Figure 16: Disease-Free Survival by Disease Group

0.36.3.1 Understanding Greenwood's formula (optional)

To see where Greenwood's formula comes from, let $x_i = Y_i - d_i$. We approximate the solution treating each time as independent, with Y_i fixed and ignore randomness in times of failure and we treat x_i as independent binomials $\text{Bin}(Y_i, p_i)$. Letting $S(t)$ be the "true" survival function

$$\hat{S}(t) = \prod_{t_i<t} x_i/Y_i$$
$$S(t) = \prod_{t_i<t} p_i$$

$$\begin{aligned}\frac{\hat{S}(t)}{S(t)} &= \prod_{t_i<t} \frac{x_i}{p_i Y_i} = \prod_{t_i<t} \frac{\hat{p}_i}{p_i} \\ &= \prod_{t_i<t} \left(1 + \frac{\hat{p}_i - p_i}{p_i}\right) \\ &\approx 1 + \sum_{t_i<t} \frac{\hat{p}_i - p_i}{p_i}\end{aligned}$$

$$
\begin{aligned}
\operatorname{Var}\left(\frac{\hat{S}(t)}{S(t)}\right) &\approx \operatorname{Var}\left(1+\sum_{t_i<t}\frac{\hat{p}_i-p_i}{p_i}\right) \\
&= \sum_{t_i<t}\frac{1}{p_i^2}\frac{p_i(1-p_i)}{Y_i} \\
&= \sum_{t_i<t}\frac{(1-p_i)}{p_i Y_i} \approx \sum_{t_i<t}\frac{(1-x_i/Y_i)}{x_i} \\
&= \sum_{t_i<t}\frac{Y_i-x_i}{x_i Y_i} = \sum_{t_i<t}\frac{d_i}{Y_i(Y_i-d_i)} \\
\operatorname{Var}\left(\hat{S}(t)\right) &\approx \hat{S}(t)^2 \sum_{t_i<t}\frac{d_i}{Y_i(Y_i-d_i)}
\end{aligned}
$$

0.36.4 Test for differences among the disease groups

Here we compute a chi-square test for assocation between disease group (`group`) and disease-free survival:

```
survdiff(surv ~ group, data = bmt)
```

```
Call:
survdiff(formula = surv ~ group, data = bmt)

                    N Observed Expected (O-E)^2/E (O-E)^2/V
group=ALL          38       24     21.9     0.211     0.289
group=Low Risk AML 54       25     40.0     5.604    11.012
group=High Risk AML 45      34     21.2     7.756    10.529

 Chisq= 13.8  on 2 degrees of freedom, p= 0.001
```

0.36.5 Cumulative Hazard

$$
\begin{aligned}
h(t) &\stackrel{\text{def}}{=} P(T=t|T\geq t) \\
&= \frac{p(T=t)}{P(T\geq t)} \\
&= -\frac{d}{dt}\log\{S(t)\}
\end{aligned}
$$

The **cumulative hazard** (or **integrated hazard**) function is

$$H(t) \stackrel{\text{def}}{=} \int_0^t h(t)dt$$

Since $h(t) = -\frac{d}{dt}\log\{S(t)\}$ as shown above, we have:

$$H(t) = -\log\{S\}(t)$$

So we can estimate $H(t)$ as:

$$\begin{aligned}\hat{H}(t) &= -\log\left\{\hat{S}(t)\right\} \\ &= -\log\left\{\prod_{t_i<t}\left[1-\frac{d_i}{Y_i}\right]\right\} \\ &= -\sum_{t_i<t}\log\left\{1-\frac{d_i}{Y_i}\right\}\end{aligned}$$

This is the **Kaplan-Meier (product-limit) estimate of cumulative hazard**.

0.36.5.1 Example: Cumulative Hazard Curves for Bone-Marrow Transplant (`bmt`) data

```
autoplot(
  fun = "cumhaz",
  km_model1,
  conf.int = FALSE,
  ylab = "Cumulative hazard (disease-free survival)",
  xlab = "Time since transplant (days)") +
  theme_bw() +
  theme(legend.position="bottom")
```

0.37 Nelson-Aalen Estimates of Cumulative Hazard and Survival

The point hazard at time t_i can be estimated by d_i/Y_i, which leads to the **Nelson-Aalen estimator of the cumulative hazard**:

$$\hat{H}_{NA}(t) \stackrel{\text{def}}{=} \sum_{t_i<t}\frac{d_i}{Y_i}$$

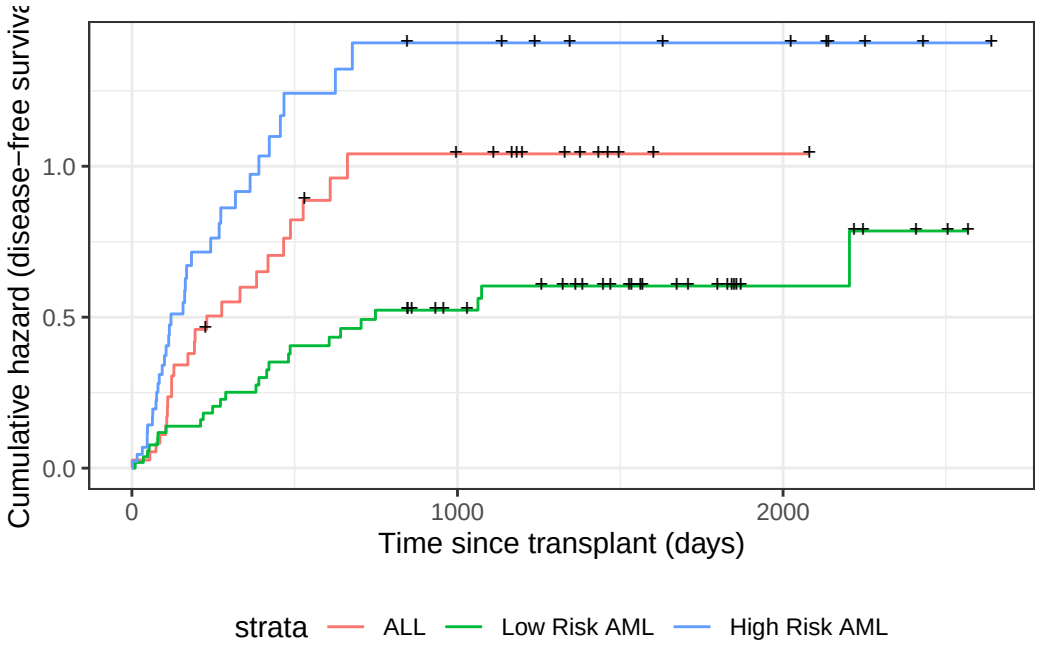


Figure 17: Disease-Free Cumulative Hazard by Disease Group

The variance of this estimator is approximately:

$$\begin{aligned}\hat{\text{Var}}\left(\hat{H}_{NA}(t)\right) &= \sum_{t_i<t} \frac{(d_i/Y_i)(1-d_i/Y_i)}{Y_i} \\ &\approx \sum_{t_i<t} \frac{d_i}{Y_i^2}\end{aligned}$$

Since $S(t) = \exp\{-H(t)\}$, the Nelson-Aalen cumulative hazard estimate can be converted into an alternate estimate of the survival function:

$$\begin{aligned}\hat{S}_{NA}(t) &= \exp\left\{-\hat{H}_{NA}(t)\right\} \\ &= \exp\left\{-\sum_{t_i<t} \frac{d_i}{Y_i}\right\} \\ &= \prod_{t_i<t} \exp\left\{-\frac{d_i}{Y_i}\right\}\end{aligned}$$

Compare these with the corresponding Kaplan-Meier estimates:

$$\hat{H}_{KM}(t) = -\sum_{t_i<t} \log\left\{1 - \frac{d_i}{Y_i}\right\}$$

$$\hat{S}_{KM}(t) = \prod_{t_i<t}\left[1 - \frac{d_i}{Y_i}\right]$$

The product limit estimate and the Nelson-Aalen estimate often do not differ by much. The latter is considered more accurate in small samples and also directly estimates the cumulative hazard. The `"fleming-harrington"` method for `survfit()` reduces to Nelson-Aalen when the data are unweighted. We can also estimate the cumulative hazard as the negative log of the KM survival function estimate.

0.37.1 Application to `bmt` dataset

```
na_fit = survfit(
  formula = surv ~ group,
  type = "fleming-harrington",
  data = bmt)

km_fit = survfit(
  formula = surv ~ group,
  type = "kaplan-meier",
  data = bmt)

km_and_na =
  bind_rows(
    .id = "model",
    "Kaplan-Meier" = km_fit |> fortify(surv.connect = TRUE),
    "Nelson-Aalen" = na_fit |> fortify(surv.connect = TRUE)
  ) |>
  as_tibble()
```

```
km_and_na |>
  ggplot(aes(x = time, y = surv, col = model)) +
  geom_step() +
  facet_grid(. ~ strata) +
  theme_bw() +
  ylab("S(t) = P(T>=t)") +
  xlab("Survival time (t, days)") +
  theme(legend.position = "bottom")
```

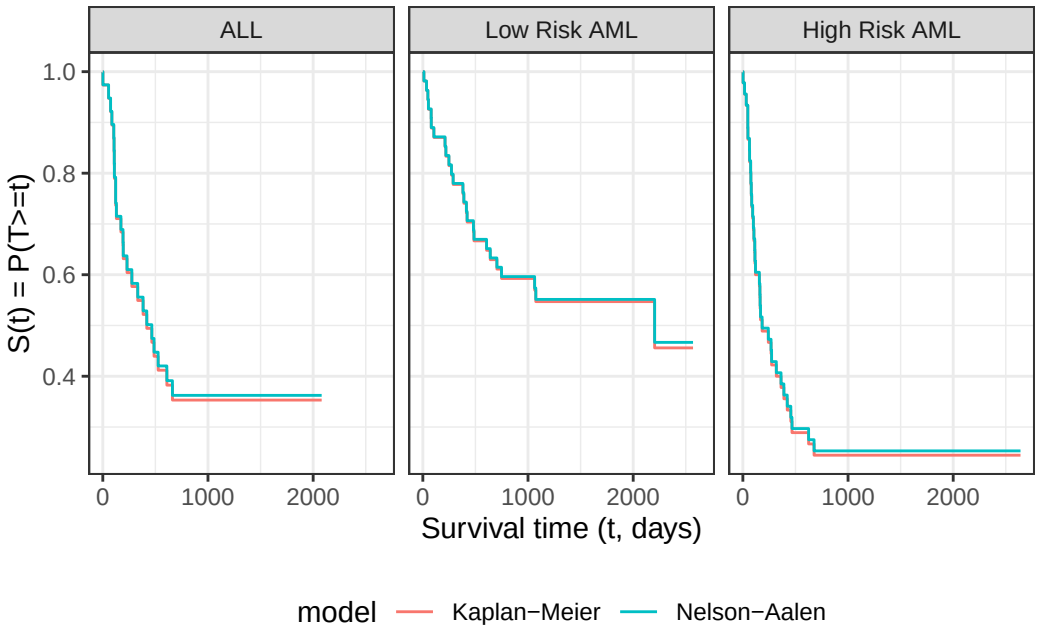


Figure 18: Kaplan-Meier and Nelson-Aalen Survival Function Estimates, stratified by disease group

The Kaplan-Meier and Nelson-Aalen survival estimates are very similar for this dataset.

0.38 Proportional Hazards Models

Configuring R

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(dplyr) # manipulate data
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(conflicted) # check for conflicting function definitions
conflicts_prefer(dplyr::filter)
```

Here are some R settings I use in this document:

```
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
options('digits' = 4)
```

0.39 The proportional hazards model

0.39.1 Background on the Proportional Hazards Model

The exponential distribution has constant hazard:

$$\begin{aligned} f(t) &= \lambda e^{-\lambda t} \\ S(t) &= e^{-\lambda t} \\ h(t) &= \lambda \end{aligned}$$

Let's make two generalizations. First, we let the hazard depend on some covariates $x_1, x_2, \ldots, x_p$; we will indicate this dependence by extending our notation for hazard:

$$h(t|x) \stackrel{\text{def}}{=} p(T = t | T \geq t, X = x)$$

Second, we let the base hazard depend on t, but not on the covariates (for now). We can do this using either parametric or semi-parametric approaches.

0.39.2 Cox's Proportional Hazards Model

The generalization is that the hazard function is

$$\begin{aligned} h(t|x) &= h_0(t)\theta(x) \\ \theta(x) &= \exp\{\eta(x)\} \\ \eta(x) &= x'\beta \\ &\stackrel{\text{def}}{=} \beta_1 x_1 + \cdots + \beta_p x_p \end{aligned}$$

The relationship between $h(t|x)$ and $\eta(x)$ has a log link (that is, $\log\{h(t|x)\} = \log\{h_0(t)\} + \eta(x)$), as in a generalized linear model.

This model is **semi-parametric**, because the linear predictor depends on estimated parameters but the base hazard function is unspecified. There is no constant term in $\eta(x)$, because it is absorbed in the base hazard.

Alternatively, we could define $\beta_0(t) = \log\{h_0(t)\}$, and then $\eta(x,t) = \beta_0(t) + \beta_1 x_1 + \cdots + \beta_p x_p$.

For two different individuals with covariate patterns x_1 and x_2, the ratio of the hazard functions (a.k.a. **hazard ratio**, a.k.a. **relative hazard**) is:

$$\begin{aligned}\frac{h(t|x_1)}{h(t|x_2)} &= \frac{h_0(t)\theta(x_1)}{h_0(t)\theta(x_2)} \\ &= \frac{\theta(x_1)}{\theta(x_2)}\end{aligned}$$

Under the proportional hazards model, this ratio (a.k.a. proportion) does not depend on t. This property is a structural limitation of the model; it is called the **proportional hazards assumption**.

Definition 0.12 (proportional hazards)**.** A conditional probability distribution $p(T|X)$ has **proportional hazards** if the hazard ratio $h(t|x_1)/h(t|x_2)$ does not depend on t. Mathematically, it can be written as:

$$\frac{h(t|x_1)}{h(t|x_2)} = \theta(x_1, x_2)$$

As we saw above, Cox's proportional hazards model has this property, with $\theta(x_1, x_2) = \frac{\theta(x_1)}{\theta(x_2)}$.

> **Note**
>
> We are using two similar notations, $\theta(x_1, x_2)$ and $\theta(x)$. We can link these notations if we define $\theta(x) \stackrel{\text{def}}{=} \theta(x, 0)$ and $\theta(0) = 1$.

It also has additional notable properties:

$$\begin{aligned}\frac{h(t|x_1)}{h(t|x_2)} &= \frac{\theta(x_1)}{\theta(x_2)} \\ &= \frac{\exp\{\eta(x_1)\}}{\exp\{\eta(x_2)\}} \\ &= \exp\{\eta(x_1) - \eta(x_2)\} \\ &= \exp\{x_1'\beta - x_2'\beta\} \\ &= \exp\{(x_1 - x_2)'\beta\}\end{aligned}$$

Hence on the log scale, we have:

$$\begin{aligned}\log\left\{\frac{h(t|x_1)}{h(t|x_2)}\right\} &= \eta(x_1) - \eta(x_2) \\ &= x_1'\beta - x_2'\beta \\ &= (x_1 - x_2)'\beta\end{aligned}$$

If only one covariate x_j is changing, then:

$$\begin{aligned}\log\left\{\frac{h(t|x_1)}{h(t|x_2)}\right\} &= (x_{1j} - x_{2j}) \cdot \beta_j \\ &\propto (x_{1j} - x_{2j})\end{aligned}$$

That is, under Cox's model $h(t|x) = h_0(t)\exp\{x'\beta\}$, the log of the hazard ratio is proportional to the difference in x_j, with the proportionality coefficient equal to β_j.

Further,

$$\log\{h(t|x)\} = \log\{h_0(t)\} + x'\beta$$

That is, the covariate effects are additive on the log-hazard scale.

See also:

https://en.wikipedia.org/wiki/Proportional_hazards_model#Why_it_is_called_%22proportional%22

0.39.3 Additional properties of the proportional hazards model

If $h(t|x) = h_0(t)\theta(x)$, then:

0.39.3.1 Cumulative hazards are also proportional to $H_0(t)$

$$\begin{aligned} H(t|x) &\stackrel{\text{def}}{=} \int_{u=0}^{t} h(u)du \\ &= \int_{u=0}^{t} h_0(u)\theta(x)du \\ &= \theta(x)\int_{u=0}^{t} h_0(u)du \\ &= \theta(x)H_0(t) \end{aligned}$$

where $H_0(t) \stackrel{\text{def}}{=} H(t|0) = \int_{u=0}^{t} h_0(u)du$.

0.39.3.2 Survival functions are exponential multiples of $S_0(t)$

$$\begin{aligned} S(t|x) &= \exp\{-H(t|x)\} \\ &= \exp\{-\theta(x)\cdot H_0(t)\} \\ &= (\exp\{-H_0(t)\})^{\theta(x)} \\ &= (S_0(t))^{\theta(x)} \end{aligned}$$

where $S_0(t) \stackrel{\text{def}}{=} P(T \geq t|X = 0)$ is the survival function for an individual whose covariates are all equal to their default values.

0.39.4 Testing the proportional hazards assumption

The Nelson-Aalen estimate of the cumulative hazard is usually used for estimates of the hazard and often the cumulative hazard.

If the hazards of the three groups are proportional, that means that the ratio of the hazards is constant over t. We can test this using the ratios of the estimated cumulative hazards, which also would be proportional, as shown above.

```
library(KMsurv)
library(survival)
data(bmt)

bmt =
  bmt |>
  as_tibble() |>
  mutate(
```

```
    group =
      group |>
      factor(
        labels = c("ALL","Low Risk AML","High Risk AML")))

nafit = survfit(
  formula = Surv(t2,d3) ~ group,
  type = "fleming-harrington",
  data = bmt)

bmt_curves = tibble(timevec = 1:1000)
sf1 <- with(nafit[1], stepfun(time,c(1,surv)))
sf2 <- with(nafit[2], stepfun(time,c(1,surv)))
sf3 <- with(nafit[3], stepfun(time,c(1,surv)))

bmt_curves =
  bmt_curves |>
  mutate(
    cumhaz1 = -log(sf1(timevec)),
    cumhaz2 = -log(sf2(timevec)),
    cumhaz3 = -log(sf3(timevec)))
```

```
library(ggplot2)
bmt_rel_hazard_plot =
  bmt_curves |>
  ggplot(
    aes(
      x = timevec,
      y = cumhaz1/cumhaz2)
  ) +
  geom_line(aes(col = "ALL/Low Risk AML")) +
  ylab("Hazard Ratio") +
  xlab("Time") +
  ylim(0,6) +
  geom_line(aes(y = cumhaz3/cumhaz1, col = "High Risk AML/ALL")) +
  geom_line(aes(y = cumhaz3/cumhaz2, col = "High Risk AML/Low Risk AML")) +
  theme_bw() +
  labs(colour = "Comparison") +
  theme(legend.position="bottom")

print(bmt_rel_hazard_plot)
```

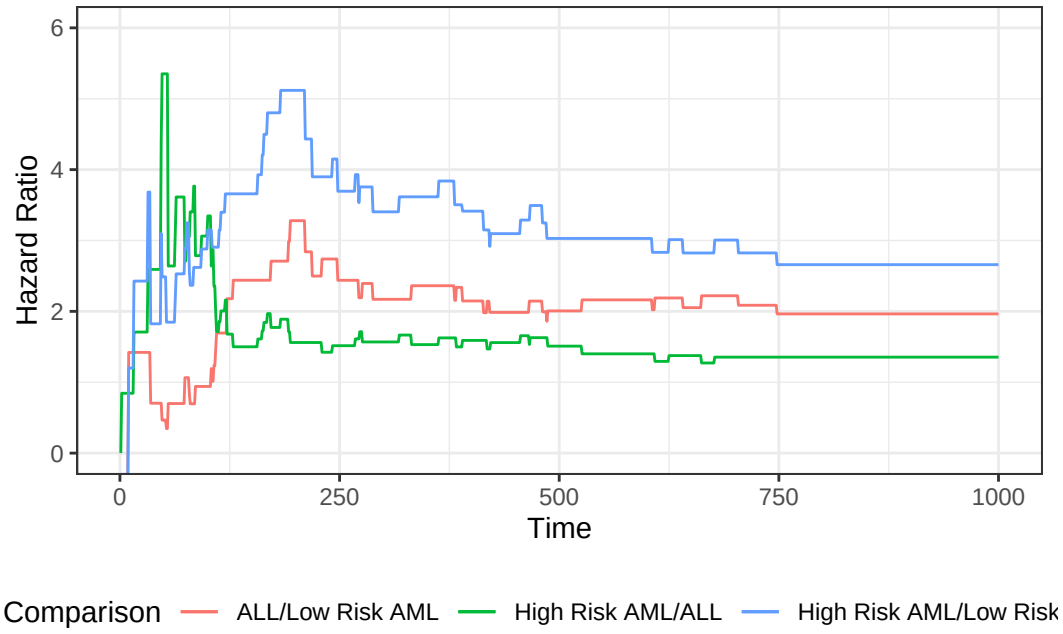


Figure 19: Hazard Ratios by Disease Group

We can zoom in on 30-300 days to take a closer look:

```
bmt_rel_hazard_plot + xlim(c(30,300))
```

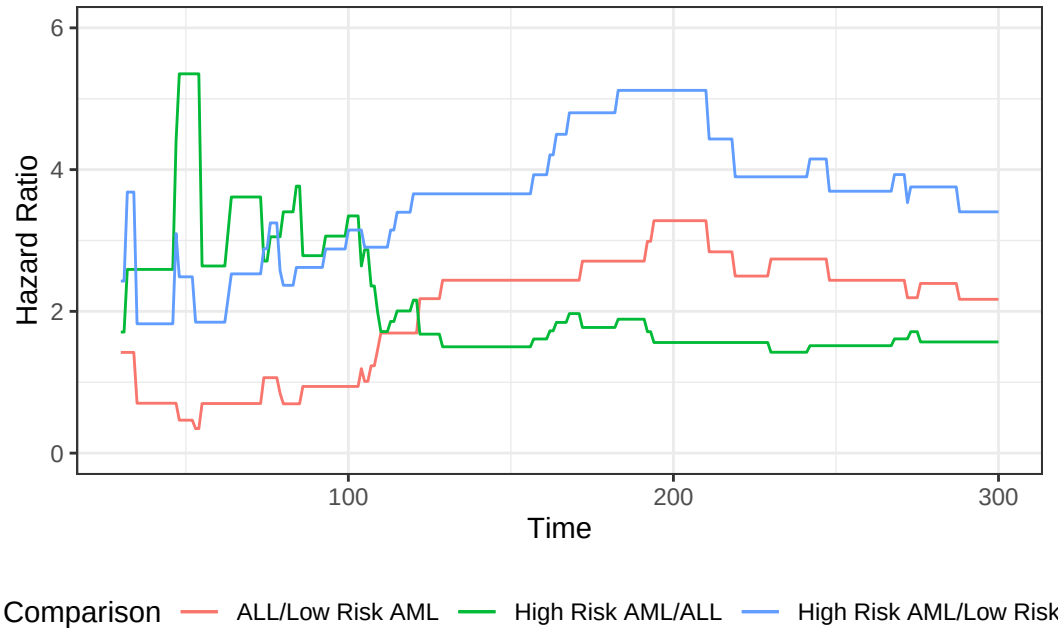


Figure 20: Hazard Ratios by Disease Group (30-300 Days)

0.39.5 Smoothed hazard functions

The Nelson-Aalen estimate of the cumulative hazard is usually used for estimates of the hazard. Since the hazard is the derivative of the cumulative hazard, we need a smooth estimate of the cumulative hazard, which is provided by smoothing the step-function cumulative hazard.

The R package **muhaz** handles this for us. What we are looking for is whether the hazard function is more or less the same shape, increasing, decreasing, constant, etc. Are the hazards "proportional"?

```
plot(
  survfit(Surv(t2,d3)~group,data=bmt),
  col=1:3,
  lwd=2,
  fun="cumhaz",
  mark.time = TRUE)
legend("bottomright",c("ALL","Low Risk AML","High Risk AML"),col=1:3,lwd=2)
```

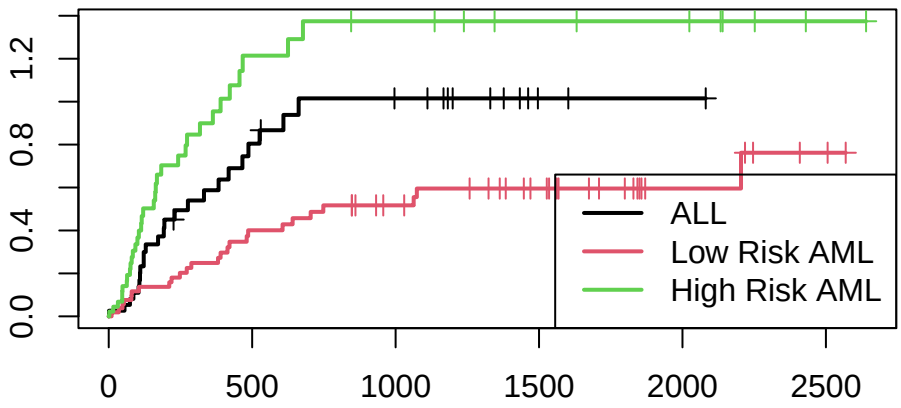


Figure 21: Disease-Free Cumulative Hazard by Disease Group

```
library(muhaz)

muhaz(bmt$t2,bmt$d3,bmt$group=="High Risk AML") |> plot(lwd=2,col=3)
muhaz(bmt$t2,bmt$d3,bmt$group=="ALL") |> lines(lwd=2,col=1)
muhaz(bmt$t2,bmt$d3,bmt$group=="Low Risk AML") |> lines(lwd=2,col=2)
legend("topright",c("ALL","Low Risk AML","High Risk AML"),col=1:3,lwd=2)
```

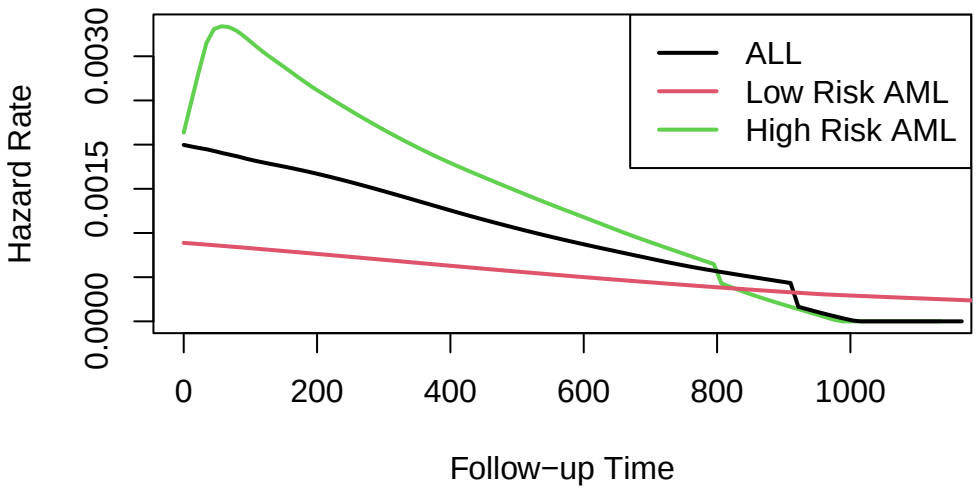


Figure 22: Smoothed Hazard Rate Estimates by Disease Group

Group 3 was plotted first because it has the highest hazard.

We will see that except for an initial blip in the high risk AML group, the hazards look roughly proportional . They are all strongly decreasing.

0.39.6 Fitting the Proportional Hazards Model

How do we fit a proportional hazards regression model? We need to estimate the coefficients of the covariates, and we need to estimate the base hazard $h_0(t)$. For the covariates, supposing for simplicity that there are no tied event times, let the event times for the whole data set be $t_1, t_2, \ldots, t_D$. Let the risk set at time t_i be $R(t_i)$ and

$$\begin{aligned} \eta(x) &= \beta_1 x_1 + \cdots + \beta_p x_p \\ \theta(x) &= e^{\eta(x)} \\ h(t|X = x) &= h_0(t)e^{\eta(x)} = \theta(x)h_0(t) \end{aligned}$$

Conditional on a single failure at time t, the probability that the event is due to subject $f \in R(t)$ is approximately

$$\begin{aligned}\Pr(f \text{ fails}|1 \text{ failure at } t) &= \frac{h_0(t)e^{\eta(x_f)}}{\sum_{k\in R(t)} h_0(t)e^{\eta(x_f)}} \\ &= \frac{\theta(x_f)}{\sum_{k\in R(t)} \theta(x_k)}\end{aligned}$$

The logic behind this has several steps. We first fix (ex post) the failure times and note that in this discrete context, the probability p_j that a subject j in the risk set fails at time t is just the hazard of that subject at that time.

If all of the p_j are small, the chance that exactly one subject fails is

$$\sum_{k\in R(t)} p_k \left[\prod_{m\in R(t), m\neq k} (1-p_m)\right] \approx \sum_{k\in R(t)} p_k$$

If subject i is the one who experiences the event of interest at time t_i, then the **partial likelihood** is

$$\mathcal{L}^*(\beta|T) = \prod_i \frac{\theta(x_i)}{\sum_{k\in R(t_i)} \theta(x_k)}$$

and we can numerically maximize this with respect to the coefficients β that specify $\eta(x) = x'\beta$. When there are tied event times adjustments need to be made, but the likelihood is still similar. Note that we don't need to know the base hazard to solve for the coefficients.

Once we have coefficient estimates $\hat{\beta} = (\hat{\beta}_1, \ldots, \hat{\beta}_p)$, this also defines $\hat{\eta}(x)$ and $\hat{\theta}(x)$ and then the estimated base cumulative hazard function is

$$\hat{H}(t) = \sum_{t_i<t} \frac{d_i}{\sum_{k\in R(t_i)} \theta(x_k)}$$

which reduces to the Nelson-Aalen estimate when there are no covariates. There are numerous other estimates that have been proposed as well.

0.40 Cox Model for the `bmt` data

0.40.1 Fit the model

```
bmt.cox <- coxph(Surv(t2, d3) ~ group, data = bmt)
summary(bmt.cox)
```

```
Call:
coxph(formula = Surv(t2, d3) ~ group, data = bmt)

  n= 137, number of events= 83

                     coef exp(coef) se(coef)     z Pr(>|z|)
groupLow Risk AML  -0.574     0.563    0.287 -2.00    0.046 *
groupHigh Risk AML  0.383     1.467    0.267  1.43    0.152
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                   exp(coef) exp(-coef) lower .95 upper .95
groupLow Risk AML      0.563      1.776     0.321     0.989
groupHigh Risk AML     1.467      0.682     0.869     2.478

Concordance= 0.625  (se = 0.03 )
Likelihood ratio test= 13.4  on 2 df,   p=0.001
Wald test            = 13  on 2 df,   p=0.001
Score (logrank) test = 13.8  on 2 df,   p=0.001
```

The table provides hypothesis tests comparing groups 2 and 3 to group 1. Group 3 has the highest hazard, so the most significant comparison is not directly shown.

The coefficient 0.3834 is on the log-hazard-ratio scale, as in log-risk-ratio. The next column gives the hazard ratio 1.4673, and a hypothesis (Wald) test.

The (not shown) group 3 vs. group 2 log hazard ratio is $0.3834 + 0.5742 = 0.9576$. The hazard ratio is then $\exp(0.9576)$ or 2.605.

Inference on all coefficients and combinations can be constructed using `coef(bmt.cox)` and `vcov(bmt.cox)` as with logistic and poisson regression.

Concordance is agreement of first failure between pairs of subjects and higher predicted risk between those subjects, omitting non-informative pairs.

The Rsquare value is Cox and Snell's pseudo R-squared and is not very useful.

`summary()` prints three tests for whether the model with the group covariate is better than the one without

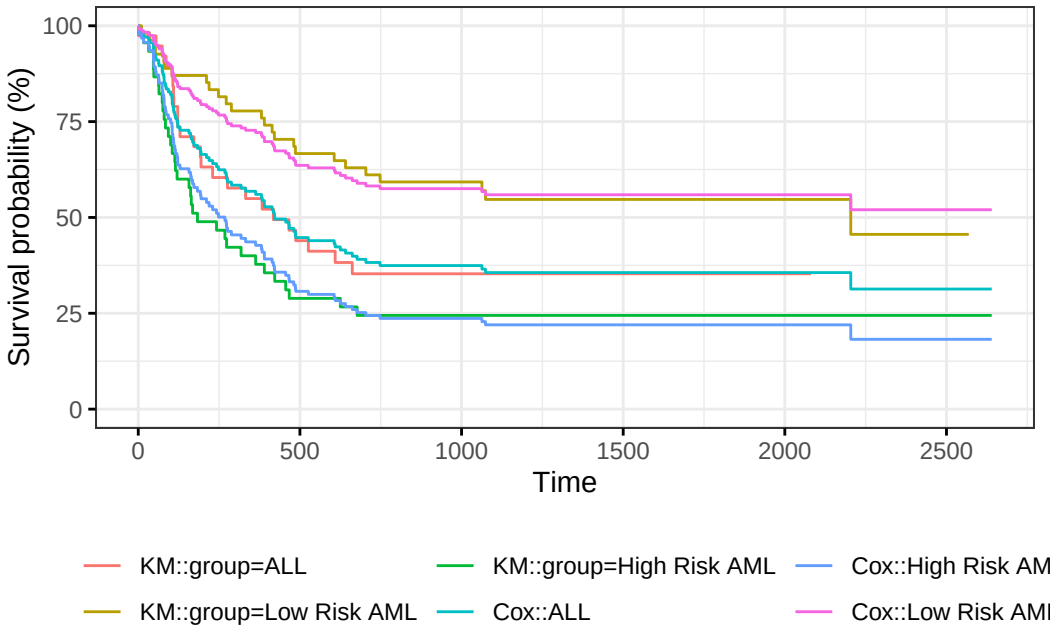
- `Likelihood ratio test` (chi-squared)
- `Wald test` (also chi-squared), obtained by adding the squares of the z-scores
- `Score` = log-rank test, as with comparison of survival functions.

The likelihood ratio test is probably best in smaller samples, followed by the Wald test.

0.40.2 Survival Curves from the Cox Model

We can take a look at the resulting group-specific curves:

```
#| fig-cap: "Survival Functions for Three Groups by KM and Cox Model"

km_fit = survfit(Surv(t2, d3) ~ group, data = as.data.frame(bmt))

cox_fit = survfit(
  bmt.cox,
  newdata =
    data.frame(
      group = unique(bmt$group),
      row.names = unique(bmt$group)))

library(survminer)

list(KM = km_fit, Cox = cox_fit) |>
  survminer::ggsurvplot(
    # facet.by = "group",
    legend = "bottom",
    legend.title = "",
    combine = TRUE,
    fun = 'pct',
    size = .5,
    ggtheme = theme_bw(),
    conf.int = FALSE,
    censor = FALSE) |>
  suppressWarnings() # ggsurvplot() throws some warnings that aren't too worrying
```



When we use `survfit()` with a Cox model, we have to specify the covariate levels we are interested in; the argument `newdata` should include a `data.frame` with the same named columns as the predictors in the Cox model and one or more levels of each.

Otherwise (that is, if the `newdata` argument is missing), a curve is produced for a single "pseudo" subject with covariate values equal to the means component of the fit.

The resulting curve(s) almost never make sense, but the default remains due to an unwarranted attachment to the option shown by some users and by other packages.

Two particularly egregious examples are factor variables and interactions. Suppose one were studying interspecies transmission of a virus, and the data set has a factor variable with levels ("pig", "chicken") and about equal numbers of observations for each. The "mean" covariate level will be 0.5 – is this a flying pig?

0.40.3 Examining `survfit`

```
survfit(Surv(t2, d3)~group,data=bmt)
```

```
Call: survfit(formula = Surv(t2, d3) ~ group, data = bmt)

                    n events median 0.95LCL 0.95UCL
```

```
group=ALL            38     24    418     194      NA
group=Low Risk AML   54     25   2204     704      NA
group=High Risk AML  45     34    183     115     456
```

```
survfit(Surv(t2, d3)~group,data=bmt) |> summary()
```

```
Call: survfit(formula = Surv(t2, d3) ~ group, data = bmt)

                group=ALL
 time n.risk n.event survival std.err lower 95% CI upper 95% CI
    1     38       1    0.974  0.0260        0.924        1.000
   55     37       1    0.947  0.0362        0.879        1.000
   74     36       1    0.921  0.0437        0.839        1.000
   86     35       1    0.895  0.0498        0.802        0.998
  104     34       1    0.868  0.0548        0.767        0.983
  107     33       1    0.842  0.0592        0.734        0.966
  109     32       1    0.816  0.0629        0.701        0.949
  110     31       1    0.789  0.0661        0.670        0.930
  122     30       2    0.737  0.0714        0.609        0.891
  129     28       1    0.711  0.0736        0.580        0.870
  172     27       1    0.684  0.0754        0.551        0.849
  192     26       1    0.658  0.0770        0.523        0.827
  194     25       1    0.632  0.0783        0.495        0.805
  230     23       1    0.604  0.0795        0.467        0.782
  276     22       1    0.577  0.0805        0.439        0.758
  332     21       1    0.549  0.0812        0.411        0.734
  383     20       1    0.522  0.0817        0.384        0.709
  418     19       1    0.494  0.0819        0.357        0.684
  466     18       1    0.467  0.0818        0.331        0.658
  487     17       1    0.439  0.0815        0.305        0.632
  526     16       1    0.412  0.0809        0.280        0.605
  609     14       1    0.382  0.0803        0.254        0.577
  662     13       1    0.353  0.0793        0.227        0.548

                group=Low Risk AML
 time n.risk n.event survival std.err lower 95% CI upper 95% CI
   10     54       1    0.981  0.0183        0.946        1.000
   35     53       1    0.963  0.0257        0.914        1.000
   48     52       1    0.944  0.0312        0.885        1.000
   53     51       1    0.926  0.0356        0.859        0.998
   79     50       1    0.907  0.0394        0.833        0.988
```

```
  80     49       1    0.889  0.0428        0.809        0.977
 105     48       1    0.870  0.0457        0.785        0.965
 211     47       1    0.852  0.0483        0.762        0.952
 219     46       1    0.833  0.0507        0.740        0.939
 248     45       1    0.815  0.0529        0.718        0.925
 272     44       1    0.796  0.0548        0.696        0.911
 288     43       1    0.778  0.0566        0.674        0.897
 381     42       1    0.759  0.0582        0.653        0.882
 390     41       1    0.741  0.0596        0.633        0.867
 414     40       1    0.722  0.0610        0.612        0.852
 421     39       1    0.704  0.0621        0.592        0.837
 481     38       1    0.685  0.0632        0.572        0.821
 486     37       1    0.667  0.0642        0.552        0.805
 606     36       1    0.648  0.0650        0.533        0.789
 641     35       1    0.630  0.0657        0.513        0.773
 704     34       1    0.611  0.0663        0.494        0.756
 748     33       1    0.593  0.0669        0.475        0.739
1063     26       1    0.570  0.0681        0.451        0.720
1074     25       1    0.547  0.0691        0.427        0.701
2204      6       1    0.456  0.1012        0.295        0.704

                group=High Risk AML
time n.risk n.event survival std.err lower 95% CI upper 95% CI
   2     45       1    0.978  0.0220        0.936        1.000
  16     44       1    0.956  0.0307        0.897        1.000
  32     43       1    0.933  0.0372        0.863        1.000
  47     42       2    0.889  0.0468        0.802        0.986
  48     40       1    0.867  0.0507        0.773        0.972
  63     39       1    0.844  0.0540        0.745        0.957
  64     38       1    0.822  0.0570        0.718        0.942
  74     37       1    0.800  0.0596        0.691        0.926
  76     36       1    0.778  0.0620        0.665        0.909
  80     35       1    0.756  0.0641        0.640        0.892
  84     34       1    0.733  0.0659        0.615        0.875
  93     33       1    0.711  0.0676        0.590        0.857
 100     32       1    0.689  0.0690        0.566        0.838
 105     31       1    0.667  0.0703        0.542        0.820
 113     30       1    0.644  0.0714        0.519        0.801
 115     29       1    0.622  0.0723        0.496        0.781
 120     28       1    0.600  0.0730        0.473        0.762
 157     27       1    0.578  0.0736        0.450        0.742
 162     26       1    0.556  0.0741        0.428        0.721
 164     25       1    0.533  0.0744        0.406        0.701
```

```
  168     24       1    0.511  0.0745        0.384        0.680
  183     23       1    0.489  0.0745        0.363        0.659
  242     22       1    0.467  0.0744        0.341        0.638
  268     21       1    0.444  0.0741        0.321        0.616
  273     20       1    0.422  0.0736        0.300        0.594
  318     19       1    0.400  0.0730        0.280        0.572
  363     18       1    0.378  0.0723        0.260        0.550
  390     17       1    0.356  0.0714        0.240        0.527
  422     16       1    0.333  0.0703        0.221        0.504
  456     15       1    0.311  0.0690        0.201        0.481
  467     14       1    0.289  0.0676        0.183        0.457
  625     13       1    0.267  0.0659        0.164        0.433
  677     12       1    0.244  0.0641        0.146        0.409
```

```
survfit(bmt.cox)
```

```
Call: survfit(formula = bmt.cox)

       n events median 0.95LCL 0.95UCL
[1,] 137     83    422     268      NA
```

```
survfit(bmt.cox, newdata = tibble(group = unique(bmt$group)))
```

```
Call: survfit(formula = bmt.cox, newdata = tibble(group = unique(bmt$group)))

    n events median 0.95LCL 0.95UCL
1 137     83    422     268      NA
2 137     83     NA     625      NA
3 137     83    268     162     467
```

```
bmt.cox |>
  survfit(newdata = tibble(group = unique(bmt$group))) |>
  summary()
```

```
Call: survfit(formula = bmt.cox, newdata = tibble(group = unique(bmt$group)))

 time n.risk n.event survival1 survival2 survival3
    1    137       1     0.993     0.996     0.989
    2    136       1     0.985     0.992     0.978
```

10	135	1	0.978	0.987	0.968
16	134	1	0.970	0.983	0.957
32	133	1	0.963	0.979	0.946
35	132	1	0.955	0.975	0.935
47	131	2	0.941	0.966	0.914
48	129	2	0.926	0.957	0.893
53	127	1	0.918	0.953	0.882
55	126	1	0.911	0.949	0.872
63	125	1	0.903	0.944	0.861
64	124	1	0.896	0.940	0.851
74	123	2	0.881	0.931	0.830
76	121	1	0.873	0.926	0.819
79	120	1	0.865	0.922	0.809
80	119	2	0.850	0.913	0.788
84	117	1	0.843	0.908	0.778
86	116	1	0.835	0.903	0.768
93	115	1	0.827	0.899	0.757
100	114	1	0.820	0.894	0.747
104	113	1	0.812	0.889	0.737
105	112	2	0.797	0.880	0.717
107	110	1	0.789	0.875	0.707
109	109	1	0.782	0.870	0.697
110	108	1	0.774	0.866	0.687
113	107	1	0.766	0.861	0.677
115	106	1	0.759	0.856	0.667
120	105	1	0.751	0.851	0.657
122	104	2	0.735	0.841	0.637
129	102	1	0.727	0.836	0.627
157	101	1	0.720	0.831	0.617
162	100	1	0.712	0.826	0.607
164	99	1	0.704	0.821	0.598
168	98	1	0.696	0.815	0.588
172	97	1	0.688	0.810	0.578
183	96	1	0.680	0.805	0.568
192	95	1	0.672	0.800	0.558
194	94	1	0.664	0.794	0.549
211	93	1	0.656	0.789	0.539
219	92	1	0.648	0.783	0.530
230	90	1	0.640	0.778	0.520
242	89	1	0.632	0.773	0.511
248	88	1	0.624	0.767	0.501
268	87	1	0.616	0.761	0.492
272	86	1	0.608	0.756	0.482

273	85	1	0.600	0.750	0.473
276	84	1	0.592	0.745	0.464
288	83	1	0.584	0.739	0.454
318	82	1	0.576	0.733	0.445
332	81	1	0.568	0.727	0.436
363	80	1	0.560	0.722	0.427
381	79	1	0.552	0.716	0.418
383	78	1	0.544	0.710	0.409
390	77	2	0.528	0.698	0.392
414	75	1	0.520	0.692	0.383
418	74	1	0.512	0.686	0.374
421	73	1	0.504	0.680	0.366
422	72	1	0.496	0.674	0.357
456	71	1	0.488	0.667	0.349
466	70	1	0.480	0.661	0.340
467	69	1	0.472	0.655	0.332
481	68	1	0.464	0.649	0.324
486	67	1	0.455	0.642	0.315
487	66	1	0.447	0.636	0.307
526	65	1	0.439	0.629	0.299
606	63	1	0.431	0.623	0.291
609	62	1	0.423	0.616	0.283
625	61	1	0.415	0.609	0.275
641	60	1	0.407	0.603	0.267
662	59	1	0.399	0.596	0.260
677	58	1	0.391	0.589	0.252
704	57	1	0.383	0.582	0.244
748	56	1	0.374	0.575	0.237
1063	47	1	0.365	0.567	0.228
1074	46	1	0.356	0.559	0.220
2204	9	1	0.313	0.520	0.182

0.41 Adjustment for Ties (optional)

0.41.1

At each time t_i at which more than one of the subjects has an event, let d_i be the number of events at that time, D_i the set of subjects with events at that time, and let s_i be a covariate vector for an artificial subject obtained by adding up the covariate values for the subjects with an event at time t_i. Let

$$\bar{\eta}_i = \beta_1 s_{i1} + \cdots + \beta_p s_{ip}$$

and $\bar{\theta}_i = \exp\{\bar{\eta}_i\}$.

Let s_i be a covariate vector for an artificial subject obtained by adding up the covariate values for the subjects with an event at time t_i. Note that

$$\begin{aligned}
\bar{\eta}_i &= \sum_{j \in D_i} \beta_1 x_{j1} + \cdots + \beta_p x_{jp} \\
&= \beta_1 s_{i1} + \cdots + \beta_p s_{ip} \\
\bar{\theta}_i &= \exp\{\bar{\eta}_i\} \\
&= \prod_{j \in D_i} \theta_i
\end{aligned}$$

0.41.1.1 Breslow's method for ties

Breslow's method estimates the partial likelihood as

$$\begin{aligned}
L(\beta|T) &= \prod_i \frac{\bar{\theta}_i}{[\sum_{k \in R(t_i)} \theta_k]^{d_i}} \\
&= \prod_i \prod_{j \in D_i} \frac{\theta_j}{\sum_{k \in R(t_i)} \theta_k}
\end{aligned}$$

This method is equivalent to treating each event as distinct and using the non-ties formula. It works best when the number of ties is small. It is the default in many statistical packages, including PROC PHREG in SAS.

0.41.1.2 Efron's method for ties

The other common method is Efron's, which is the default in R.

$$L(\beta|T) = \prod_i \frac{\bar{\theta}_i}{\prod_{j=1}^{d_i} [\sum_{k \in R(t_i)} \theta_k - \frac{j-1}{d_i} \sum_{k \in D_i} \theta_k]}$$

This is closer to the exact discrete partial likelihood when there are many ties.

The third option in R (and an option also in SAS as `discrete`) is the "exact" method, which is the same one used for matched logistic regression.

0.41.1.3 Example: Breslow's method

Suppose as an example we have a time t where there are 20 individuals at risk and three failures. Let the three individuals have risk parameters $\theta_1, \theta_2, \theta_3$ and let the sum of the risk parameters of the remaining 17 individuals be θ_R. Then the factor in the partial likelihood at time t using Breslow's method is

$$\left(\frac{\theta_1}{\theta_R + \theta_1 + \theta_2 + \theta_3}\right)\left(\frac{\theta_2}{\theta_R + \theta_1 + \theta_2 + \theta_3}\right)\left(\frac{\theta_3}{\theta_R + \theta_1 + \theta_2 + \theta_3}\right)$$

If on the other hand, they had died in the order 1,2, 3, then the contribution to the partial likelihood would be:

$$\left(\frac{\theta_1}{\theta_R + \theta_1 + \theta_2 + \theta_3}\right)\left(\frac{\theta_2}{\theta_R + \theta_2 + \theta_3}\right)\left(\frac{\theta_3}{\theta_R + \theta_3}\right)$$

as the risk set got smaller with each failure. The exact method roughly averages the results for the six possible orderings of the failures.

0.41.1.4 Example: Efron's method

But we don't know the order they failed in, so instead of reducing the denominator by one risk coefficient each time, we reduce it by the same fraction. This is Efron's method.

$$\left(\frac{\theta_1}{\theta_R + \theta_1 + \theta_2 + \theta_3}\right)\left(\frac{\theta_2}{\theta_R + 2(\theta_1 + \theta_2 + \theta_3)/3}\right)\left(\frac{\theta_3}{\theta_R + (\theta_1 + \theta_2 + \theta_3)/3}\right)$$

0.42 Building Cox Proportional Hazards models

Configuring R

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggfortify) # help with graphics
library(ggeasy) # help with graphics
library(survival) # survival analysis
library(survminer) # survival analysis graphics
library(dplyr) # manipulate data
```

```r
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(fs) # filesystem path manipulations
library(KMsurv) # datasets from Klein and Moeschberger
library(parameters) # format model output tables for markdown
library(conflicted) # check for conflicting function definitions
conflicts_prefer(dplyr::filter)
conflicts_prefer(ggplot2::autoplot)
```

Here are some R settings I use in this document:

```r
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
options('digits' = 4)
legend_text_size = 9
```

0.43 Building Cox Proportional Hazards models

0.43.1 `hodg` Lymphoma Data Set from `KMsurv`

0.43.1.1 Participants

43 bone marrow transplant patients at Ohio State University (Avalos 1993)

0.43.1.2 Variables

- `dtype`: Disease type (Hodgkin's or non-Hodgkins lymphoma)
- `gtype`: Bone marrow graft type:
- allogeneic: from HLA-matched sibling
- autologous: from self (prior to chemo)
- `time`: time to study exit
- `delta`: study exit reason (death/relapse vs censored)
- `wtime`: waiting time to transplant (in months)
- `score`: Karnofsky score:
- 80–100: Able to carry on normal activity and to work; no special care needed.

- 50–70: Unable to work; able to live at home and care for most personal needs; varying amount of assistance needed.
- 10–60: Unable to care for self; requires equivalent of institutional or hospital care; disease may be progressing rapidly.

0.43.1.3 Data

```
data(hodg, package = "KMsurv")
hodg2 = hodg |>
  as_tibble() |>
  mutate(
    # We add factor labels to the categorical variables:
    gtype = gtype |>
      case_match(
        1 ~ "Allogenic",
        2 ~ "Autologous"),
    dtype = dtype |>
      case_match(
        1 ~ "Non-Hodgkins",
        2 ~ "Hodgkins") |>
      factor() |>
      relevel(ref = "Non-Hodgkins"),
    delta = delta |>
      case_match(
        1 ~ "dead",
        0 ~ "alive"),
    surv = Surv(
      time = time,
      event = delta == "dead")
  )
hodg2 |> print()
```

```
# A tibble: 43 x 7
   gtype     dtype         time delta score wtime   surv
   <chr>     <fct>        <int> <chr> <int> <int> <Surv>
 1 Allogenic Non-Hodgkins    28 dead     90    24    28
 2 Allogenic Non-Hodgkins    32 dead     30     7    32
 3 Allogenic Non-Hodgkins    49 dead     40     8    49
 4 Allogenic Non-Hodgkins    84 dead     60    10    84
 5 Allogenic Non-Hodgkins   357 dead     70    42   357
 6 Allogenic Non-Hodgkins   933 alive    90     9   933+
```

```
 7 Allogenic Non-Hodgkins  1078 alive   100    16  1078+
 8 Allogenic Non-Hodgkins  1183 alive    90    16  1183+
 9 Allogenic Non-Hodgkins  1560 alive    80    20  1560+
10 Allogenic Non-Hodgkins  2114 alive    80    27  2114+
# i 33 more rows
```

0.43.2 Proportional hazards model

```
hodg.cox1 = coxph(
  formula = surv ~ gtype * dtype + score + wtime,
  data = hodg2)

summary(hodg.cox1)
```

```
Call:
coxph(formula = surv ~ gtype * dtype + score + wtime, data = hodg2)

  n= 43, number of events= 26

                                  coef exp(coef) se(coef)     z Pr(>|z|)
gtypeAutologous                 0.6394    1.8953   0.5937  1.08   0.2815
dtypeHodgkins                   2.7603   15.8050   0.9474  2.91   0.0036 **
score                          -0.0495    0.9517   0.0124 -3.98  6.8e-05 ***
wtime                          -0.0166    0.9836   0.0102 -1.62   0.1046
gtypeAutologous:dtypeHodgkins -2.3709    0.0934   1.0355 -2.29   0.0220 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                              exp(coef) exp(-coef) lower .95 upper .95
gtypeAutologous                  1.8953     0.5276    0.5920     6.068
dtypeHodgkins                   15.8050     0.0633    2.4682   101.207
score                            0.9517     1.0507    0.9288     0.975
wtime                            0.9836     1.0167    0.9641     1.003
gtypeAutologous:dtypeHodgkins    0.0934    10.7074    0.0123     0.711

Concordance= 0.776  (se = 0.059 )
Likelihood ratio test= 32.1  on 5 df,   p=6e-06
Wald test            = 27.2  on 5 df,   p=5e-05
Score (logrank) test = 37.7  on 5 df,   p=4e-07
```

0.44 Diagnostic graphs for proportional hazards assumption

0.44.1 Analysis plan

- **survival function** for the four combinations of disease type and graft type.
- **observed (nonparametric) vs. expected (semiparametric) survival** functions.
- **complementary log-log survival** for the four groups.

0.44.2 Kaplan-Meier survival functions

```
km_model = survfit(
  formula = surv ~ dtype + gtype,
  data = hodg2)

km_model |>
  autoplot(conf.int = FALSE) +
  theme_bw() +
  theme(
    legend.position="bottom",
    legend.title = element_blank(),
    legend.text = element_text(size = legend_text_size)
  ) +
  guides(col=guide_legend(ncol=2)) +
  ylab('Survival probability, S(t)') +
  xlab("Time since transplant (days)")
```

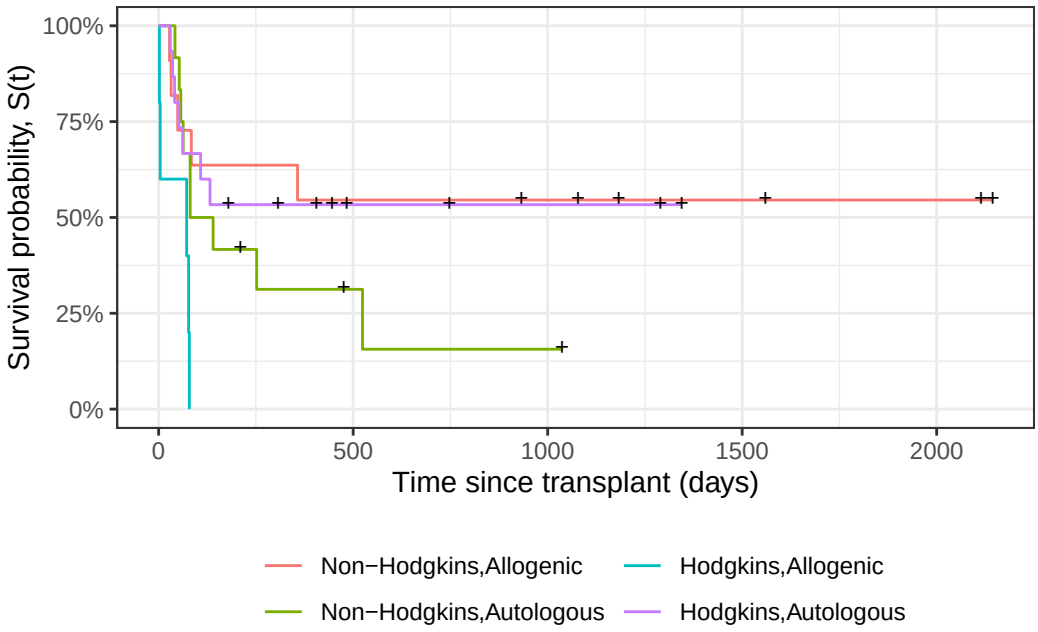


Figure 23: Kaplan-Meier Survival Curves for HOD/NHL and Allo/Auto Grafts

0.44.3 Observed and expected survival curves

```
# we need to create a tibble of covariate patterns;
# we will set score and wtime to mean values for disease and graft types:
means = hodg2 |>
  summarize(
    .by = c(dtype, gtype),
    score = mean(score),
    wtime = mean(wtime)) |>
  arrange(dtype, gtype) |>
  mutate(strata = paste(dtype, gtype, sep = ",")) |>
  as.data.frame()

# survfit.coxph() will use the rownames of its `newdata`
# argument to label its output:
rownames(means) = means$strata

cox_model =
  hodg.cox1 |>
```

```
  survfit(
    data = hodg2, # ggsurvplot() will need this
    newdata = means)
```

```
# I couldn't find a good function to reformat `cox_model` for ggplot,
# so I made my own:
stack_surv_ph = function(cox_model)
{
  cox_model$surv |>
    as_tibble() |>
    mutate(time = cox_model$time) |>
    pivot_longer(
      cols = -time,
      names_to = "strata",
      values_to = "surv") |>
    mutate(
      cumhaz = -log(surv),
      model = "Cox PH")
}

km_and_cph =
  km_model |>
  fortify(surv.connect = TRUE) |>
  mutate(
    strata = trimws(strata),
    model = "Kaplan-Meier",
    cumhaz = -log(surv)) |>
  bind_rows(stack_surv_ph(cox_model))
```

```
km_and_cph |>
  ggplot(aes(x = time, y = surv, col = model)) +
  geom_step() +
  facet_wrap(~strata) +
  theme_bw() +
  ylab("S(t) = P(T>=t)") +
  xlab("Survival time (t, days)") +
  theme(legend.position = "bottom")
```

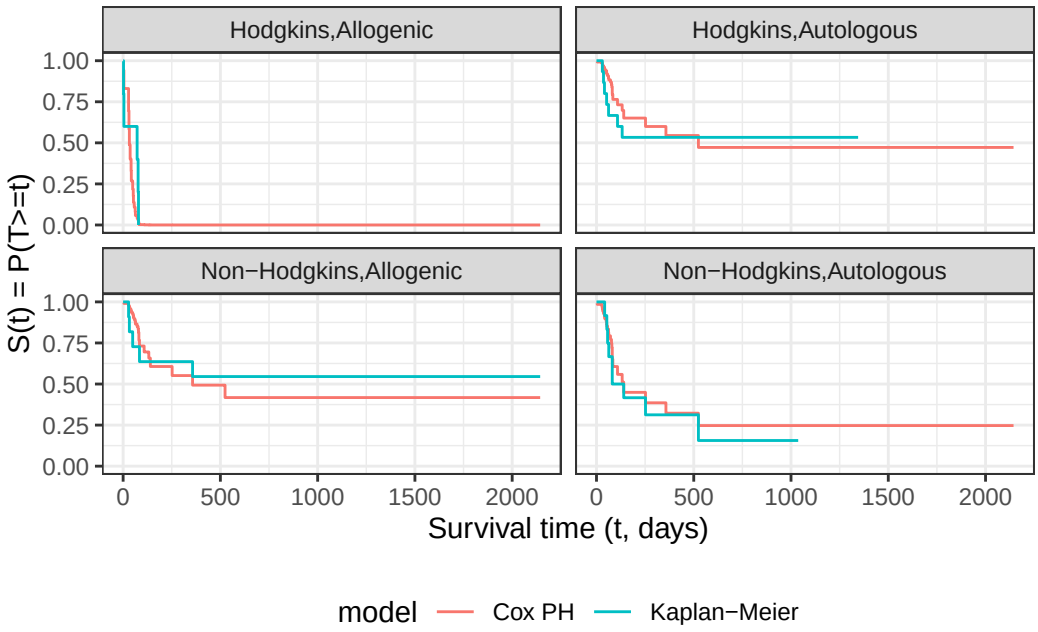


Figure 24: Observed and expected survival curves for `bmt` data

0.44.4 Cumulative hazard (log-scale) curves

Also known as "complementary log-log (clog-log) survival curves".

```
na_model = survfit(
  formula = surv ~ dtype + gtype,
  data = hodg2,
  type = "fleming")

na_model |> survminer::ggsurvplot(
  legend = "bottom",
  legend.title = "",
  ylab = "log(Cumulative Hazard)",
  xlab = "Time since transplant (days, log-scale)",
  fun = 'cloglog',
  size = .5,
  ggtheme = theme_bw(),
  conf.int = FALSE,
  censor = TRUE) +
  guides(
```

```
    col =
      guide_legend(
        ncol = 2,
        label.theme =
          element_text(
            size = legend_text_size)))
```

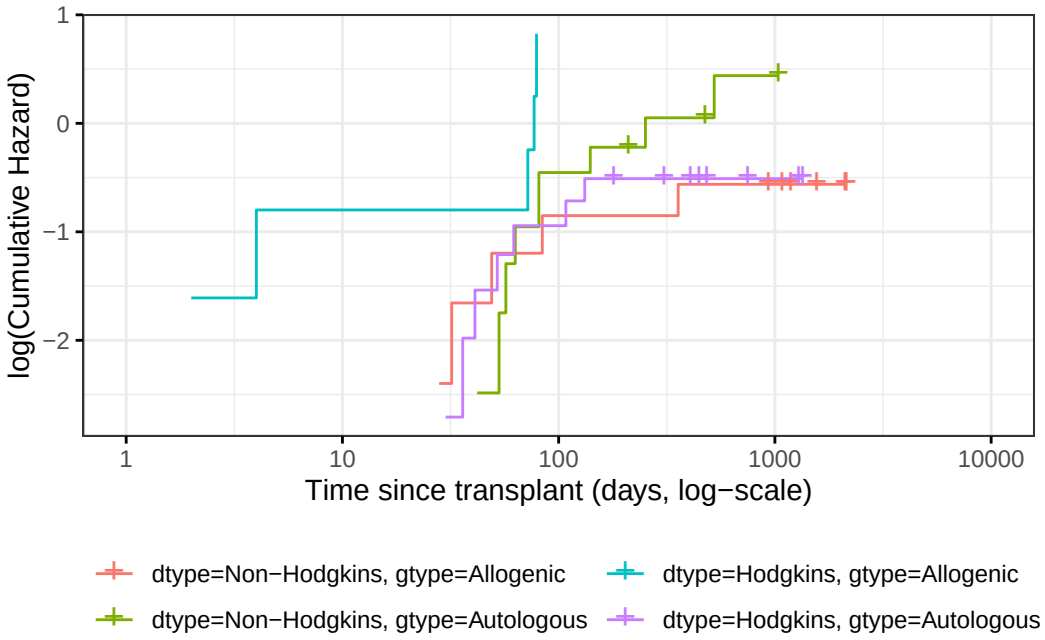


Figure 25: Complementary log-log survival curves - Nelson-Aalen estimates

Let's compare these empirical (i.e., non-parametric) curves with the fitted curves from our `coxph()` model:

```
cox_model |>
  survminer::ggsurvplot(
    facet_by = "",
    legend = "bottom",
    legend.title = "",
    ylab = "log(Cumulative Hazard)",
    xlab = "Time since transplant (days, log-scale)",
    fun = 'cloglog',
    size = .5,
```

```
    ggtheme = theme_bw(),
    censor = FALSE, # doesn't make sense for cox model
    conf.int = FALSE) +
  guides(
    col =
      guide_legend(
        ncol = 2,
        label.theme =
          element_text(
            size = legend_text_size)))
```

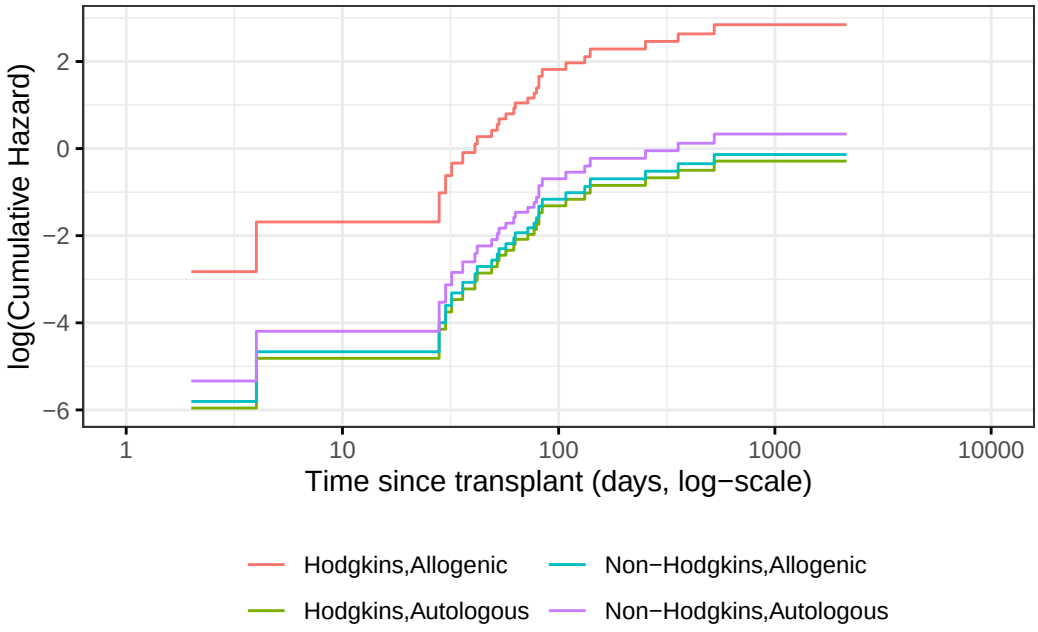


Figure 26: Complementary log-log survival curves - PH estimates

Now let's overlay these cumulative hazard curves:

```
na_and_cph =
  na_model |>
  fortify(fun = "cumhaz") |>
  # `fortify.survfit()` doesn't name cumhaz correctly:
  rename(cumhaz = surv) |>
  mutate(
```

```
    surv = exp(-cumhaz),
    strata = trimws(strata)) |>
  mutate(model = "Nelson-Aalen") |>
  bind_rows(stack_surv_ph(cox_model))

na_and_cph |>
  ggplot(
    aes(
      x = time,
      y = cumhaz,
      col = model)) +
  geom_step() +
  facet_wrap(~strata) +
  theme_bw() +
  scale_y_continuous(
    trans = "log10",
    name = "Cumulative hazard H(t) (log-scale)") +
  scale_x_continuous(
    trans = "log10",
    name = "Survival time (t, days, log-scale)") +
  theme(legend.position = "bottom")
```

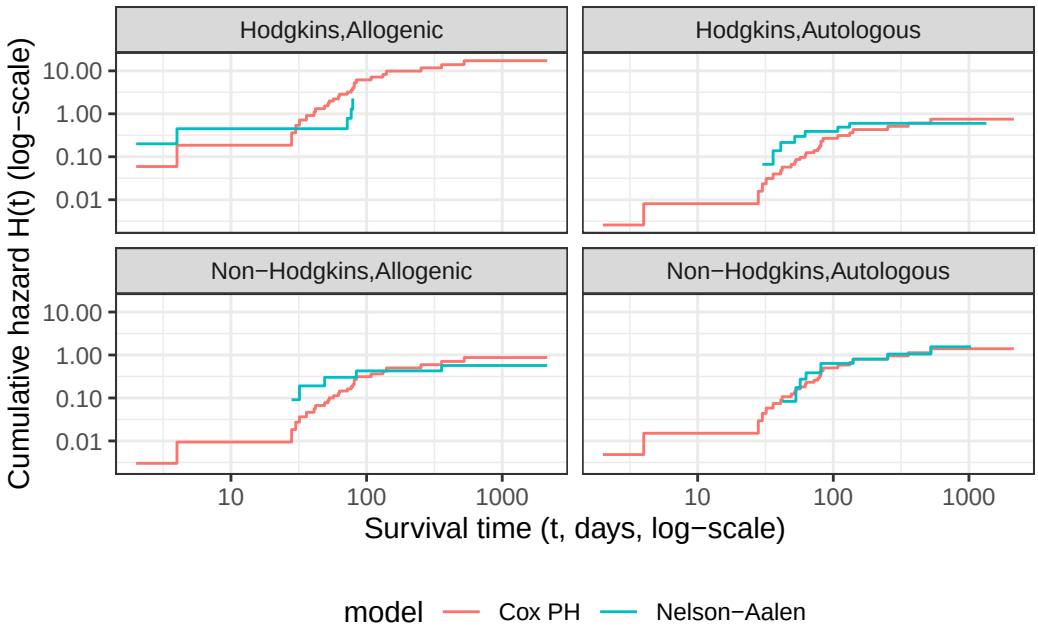


Figure 27: Observed and expected cumulative hazard curves for `bmt` data (cloglog format)

0.45 Predictions and Residuals

0.45.1 Review: Predictions in Linear Regression

- In linear regression, we have a linear predictor for each data point i

$$\eta_i = \beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi}$$
$$\hat{y}_i = \hat{\eta}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \cdots + \hat{\beta}_p x_{pi}$$
$$y_i \sim N(\eta_i, \sigma^2)$$

- $\hat{y}_i$ estimates the conditional mean of y_i given the covariate values $\tilde{x}_i$. This together with the prediction error says that we are predicting the distribution of values of y.

0.45.2 Review: Residuals in Linear Regression

- The usual residual is $r_i = y_i - \hat{y}_i$, the difference between the actual value of y and a prediction of its mean.

- The residuals are also the quantities the sum of whose squares is being minimized by the least squares/MLE estimation.

0.45.3 Predictions and Residuals in survival models

- In survival analysis, the equivalent of y_i is the event time t_i, which is unknown for the censored observations.
- The expected event time can be tricky to calculate:

$$\hat{\mathrm{E}}[T|X = x] = \int_{t=0}^{\infty} \hat{S}(t)dt$$

0.45.4 Wide prediction intervals

The nature of time-to-event data results in very wide prediction intervals:

- Suppose a cancer patient is predicted to have a mean lifetime of 5 years after diagnosis and suppose the distribution is exponential.
- If we want a 95% interval for survival, the lower end is at the 0.025 percentage point of the exponential which is `qexp(.025, rate = 1/5)` = 0.1266 years, or 1/40 of the mean lifetime.
- The upper end is at the 0.975 point which is `qexp(.975, rate = 1/5)` = 18.4444 years, or 3.7 times the mean lifetime.
- Saying that the survival time is somewhere between 6 weeks and 18 years does not seem very useful, but it may be the best we can do.
- For survival analysis, something is like a residual if it is small when the model is accurate or if the accumulation of them is in some way minimized by the estimation algorithm, but there is no exact equivalence to linear regression residuals.
- And if there is, they are mostly quite large!

0.45.5 Types of Residuals in Time-to-Event Models

- It is often hard to make a decision from graph appearances, though the process can reveal much.
- Some diagnostic tests are based on residuals as with other regression methods:
- **Schoenfeld residuals** (via `cox.zph`) for proportionality.
- **Cox-Snell residuals** for goodness of fit.
- **martingale residuals** for non-linearity.
- **dfbeta** for influence.

0.45.6 Schoenfeld residuals

- There is a Schoenfeld residual for each subject i with an event (not censored) and for each predictor x_k.
- At the event time t for that subject, there is a risk set R, and each subject j in the risk set has a risk coefficient θ_j and also a value x_{jk} of the predictor.
- The Schoenfeld residual is the difference between x_{ik} and the risk-weighted average of all the x_{jk} over the risk set.

$$r_{ik}^S = x_{ik} - \frac{\sum_{k \in R} x_{jk}\theta_k}{\sum_{k \in R} \theta_k}$$

This residual measures how typical the individual subject is with respect to the covariate at the time of the event. Since subjects should fail more or less uniformly according to risk, the Schoenfeld residuals should be approximately level over time, not increasing or decreasing.

We can test this with the correlation with time on some scale, which could be the time itself, the log time, or the rank in the set of failure times.

The default is to use the KM curve as a transform, which is similar to the rank but deals better with censoring.

The `cox.zph()` function implements a score test proposed by Grambsch and Therneau[1]:

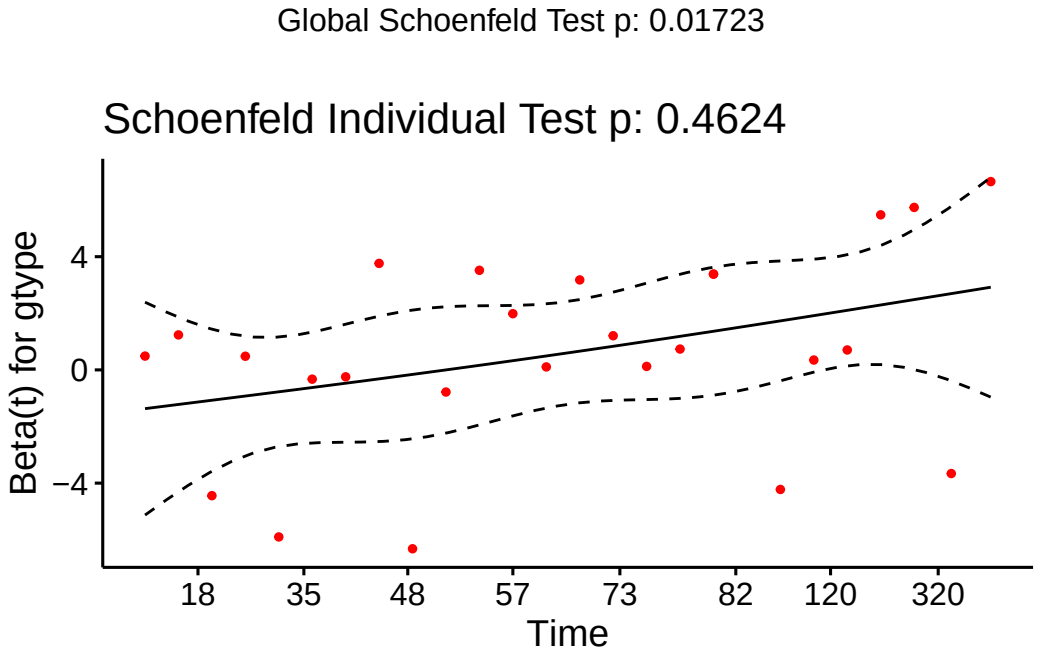
```
hodg.zph = cox.zph(hodg.cox1)
print(hodg.zph)
```

```
              chisq df     p
gtype        0.5400  1 0.462
dtype        1.8012  1 0.180
score        3.8805  1 0.049
wtime        0.0173  1 0.895
gtype:dtype  4.0474  1 0.044
GLOBAL      13.7573  5 0.017
```

[1] P. Grambsch and T. Therneau (1994), Proportional hazards tests and diagnostics based on weighted residuals. Biometrika, 81, 515-26.

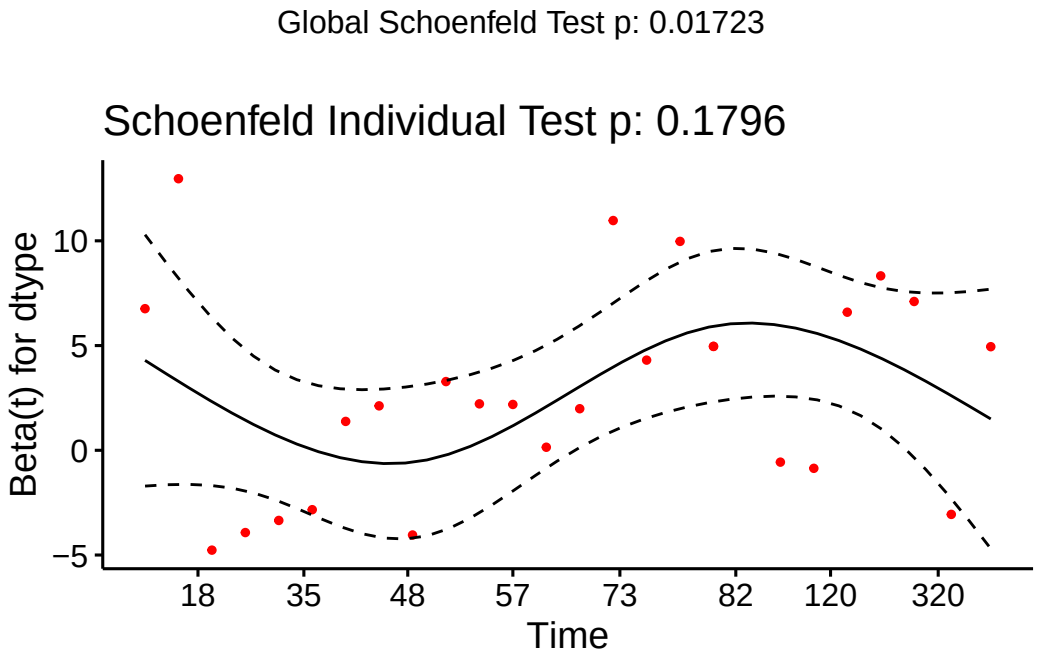
0.45.6.1 gtype

```
ggcoxzph(hodg.zph, var = "gtype")
```



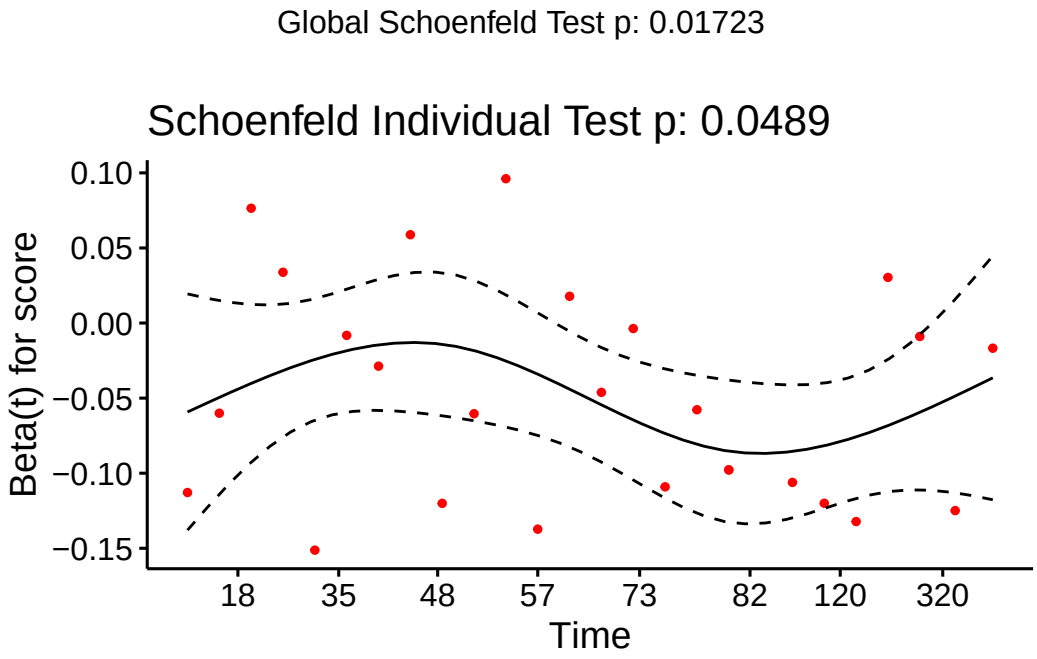
0.45.6.2 dtype

```
ggcoxzph(hodg.zph, var = "dtype")
```



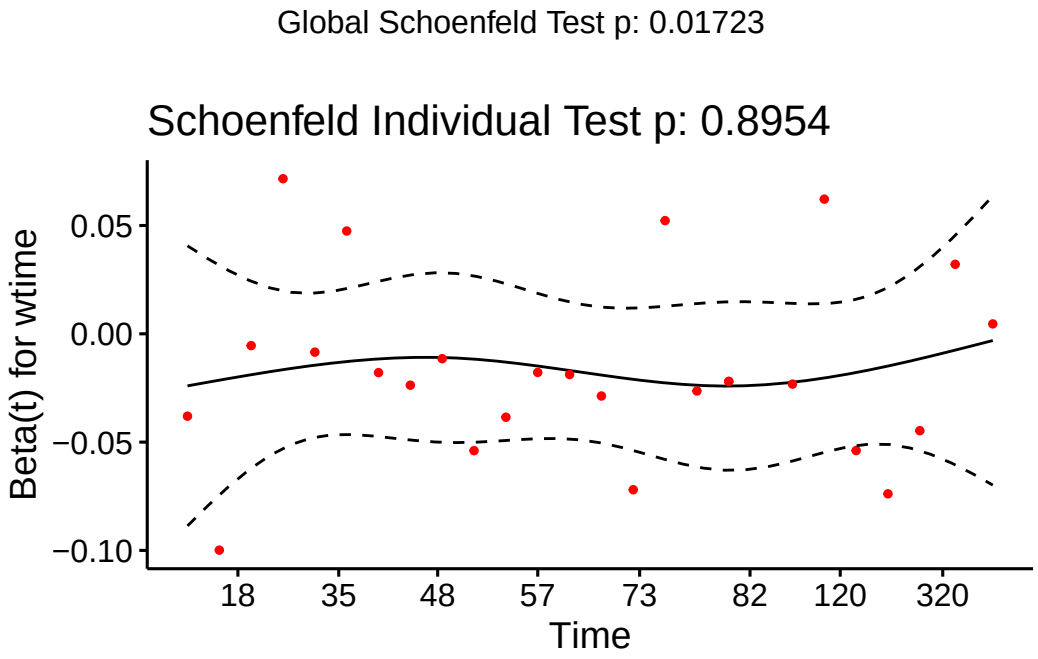
0.45.6.3 `score`

```
ggcoxzph(hodg.zph, var = "score")
```



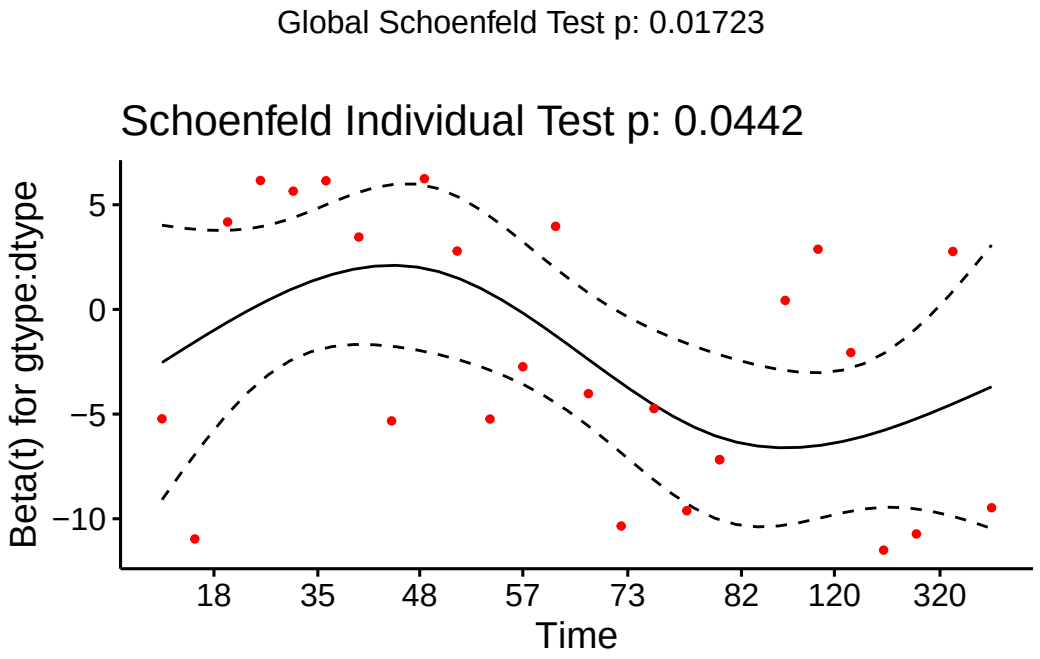
0.45.6.4 `wtime`

```
ggcoxzph(hodg.zph, var = "wtime")
```



0.45.6.5 `gtype:dtype`

```
ggcoxzph(hodg.zph, var = "gtype:dtype")
```



0.45.6.6 Conclusions

- From the correlation test, the Karnofsky score and the interaction with graft type disease type induce modest but statistically significant non-proportionality.
- The sample size here is relatively small (26 events in 43 subjects). If the sample size is large, very small amounts of non-proportionality can induce a significant result.
- As time goes on, autologous grafts are over-represented at their own event times, but those from HOD patients become less represented.
- Both the statistical tests and the plots are useful.

0.46 Goodness of Fit using the Cox-Snell Residuals

(references: Klein & Moeschberger textbook, §11.2, and Dobson & Barnett textbook, §10.6)

Suppose that an individual has a survival time T which has survival function $S(t)$, meaning that $\Pr(T > t) = S(t)$. Then $S(T)$ has a uniform distribution on $(0, 1)$.

$$\begin{aligned}\Pr(S(T_i) \leq u) &= \Pr(T_i > S_i^{-1}(u)) \\ &= S_i(S_i^{-1}(u)) \\ &= u\end{aligned}$$

Also, if U has a uniform distribution on $(0, 1)$, then what is the distribution of $-\ln(U)$?

$$\begin{aligned}\Pr(-\ln(U) < x) &= \Pr(U > \exp\{-x\}) \\ &= 1 - e^{-x}\end{aligned}$$

which is the CDF of an exponential distribution with parameter $\lambda = 1$.

So,

$$r_i^{CS} \stackrel{\text{def}}{=} -\ln[\hat{S}(t_i|x_i)] = \hat{H}(t_i|\tilde{x}_i)$$

should have an exponential distribution with constant hazard $\lambda = 1$ if the estimate $\hat{S}_i$ is accurate, which means that these values should look like a censored sample from this exponential distribution. These values are called **generalized residuals** or **Cox-Snell residuals**.

```
hodg2 = hodg2 |>
  mutate(cs = predict(hodg.cox1, type = "expected"))

surv.csr = survfit(
  data = hodg2,
  formula = Surv(time = cs, event = delta == "dead") ~ 1,
  type = "fleming-harrington")

autoplot(surv.csr, fun = "cumhaz") +
  geom_abline(aes(intercept = 0, slope = 1), col = "red") +
  theme_bw()
```

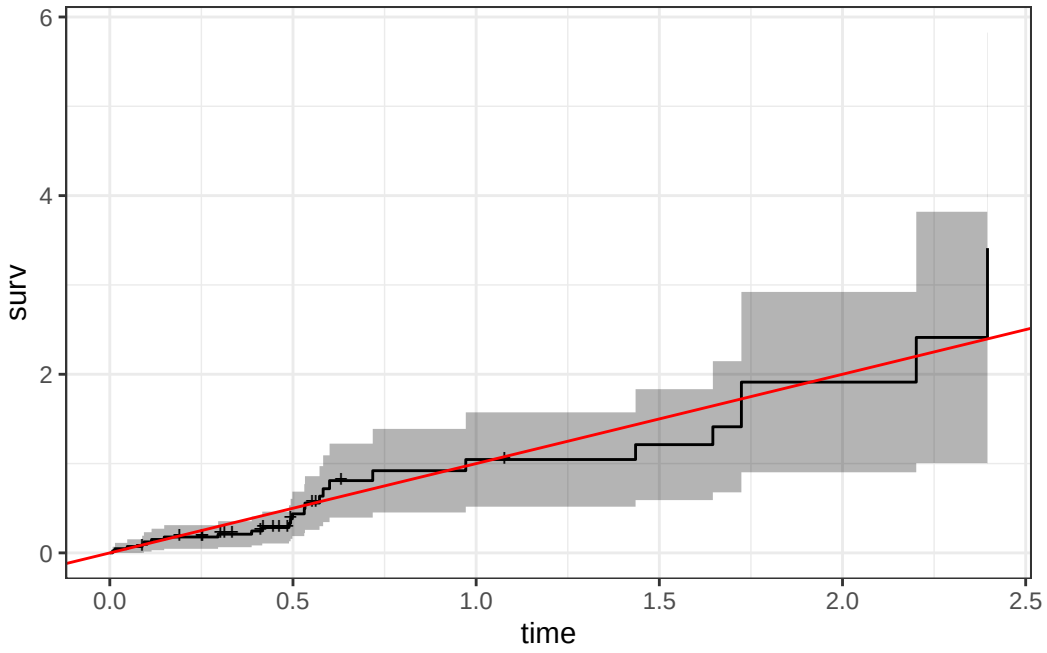


Figure 28: Cumulative Hazard of Cox-Snell Residuals

The line with slope 1 and intercept 0 fits the curve relatively well, so we don't see lack of fit using this procedure.

0.47 Martingale Residuals

The **martingale residuals** are a slight modification of the Cox-Snell residuals. If the censoring indicator is δ_i, then

$$r_i^M = \delta_i - r_i^{CS}$$

These residuals can be interpreted as an estimate of the excess number of events seen in the data but not predicted by the model. We will use these to examine the functional forms of continuous covariates.

0.47.1 Using Martingale Residuals

Martingale residuals can be used to examine the functional form of a numeric variable.

- We fit the model without that variable and compute the martingale residuals.
- We then plot these martingale residuals against the values of the variable.
- We can see curvature, or a possible suggestion that the variable can be discretized.

Let's use this to examine the `score` and `wtime` variables in the `wtime` data set.

Karnofsky score

```
hodg2 = hodg2 |>
  mutate(
    mres =
      hodg.cox1 |>
      update(. ~ . - score) |>
      residuals(type="martingale"))

hodg2 |>
  ggplot(aes(x = score, y = mres)) +
  geom_point() +
  geom_smooth(method = "loess", aes(col = "loess")) +
  geom_smooth(method = 'lm', aes(col = "lm")) +
  theme_classic() +
  xlab("Karnofsky Score") +
  ylab("Martingale Residuals") +
  guides(col=guide_legend(title = ""))
```

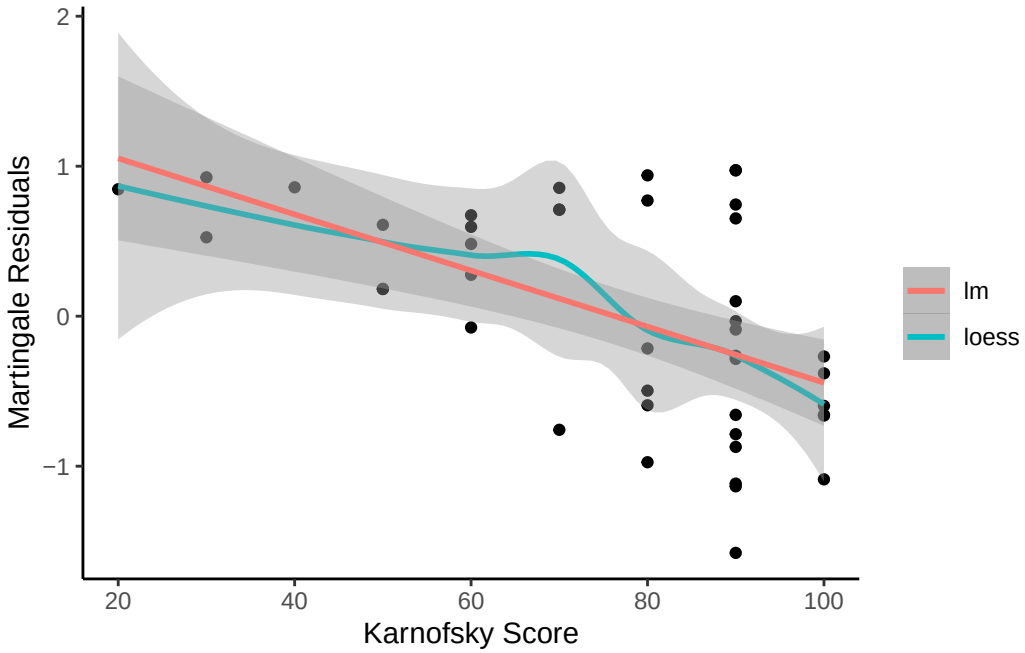


Figure 29: Martingale Residuals vs. Karnofsky Score

The line is almost straight. It could be some modest transformation of the Karnofsky score would help, but it might not make much difference.

Waiting time

```
hodg2$mres =
  hodg.cox1 |>
  update(. ~ . - wtime) |>
  residuals(type="martingale")

hodg2 |>
  ggplot(aes(x = wtime, y = mres)) +
  geom_point() +
  geom_smooth(method = "loess", aes(col = "loess")) +
  geom_smooth(method = 'lm', aes(col = "lm")) +
  theme_classic() +
  xlab("Waiting Time") +
  ylab("Martingale Residuals") +
  guides(col=guide_legend(title = ""))
```

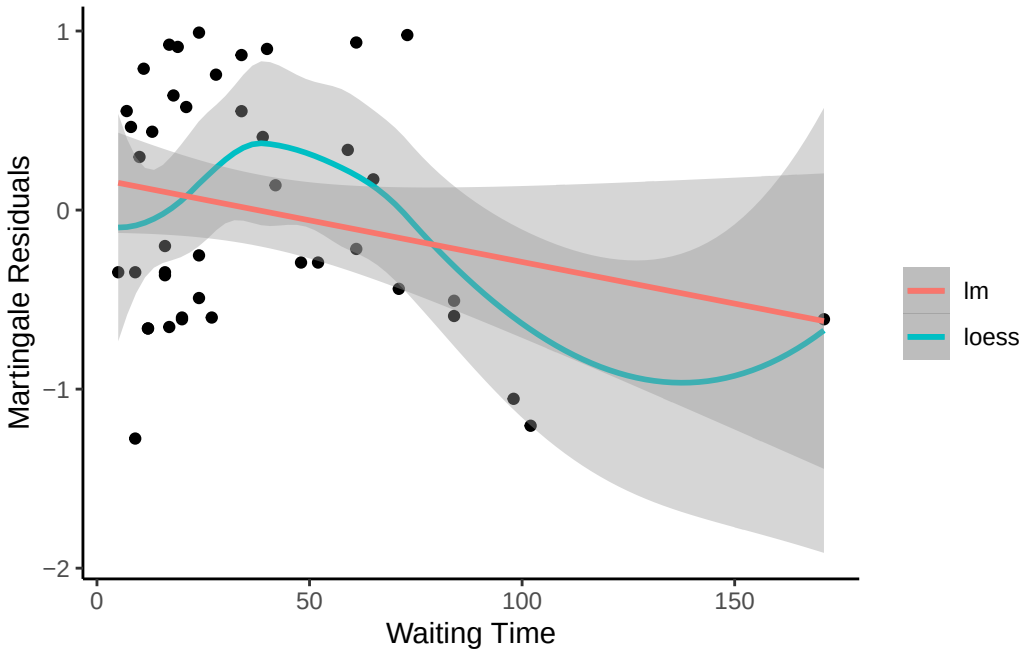


Figure 30: Martingale Residuals vs. Waiting Time

The line could suggest a step function. To see where the drop is, we can look at the largest waiting times and the associated martingale residual.

The martingale residuals are all negative for `wtime` >83 and positive for the next smallest value. A reasonable cut-point is 80 days.

Updating the model

Let's reformulate the model with dichotomized `wtime`.

```
hodg2 =
  hodg2 |>
  mutate(
    wt2 = cut(
      wtime,c(0, 80, 200),
      labels=c("short","long")))

hodg.cox2 =
  coxph(
    formula =
```

```
      Surv(time, event = delta == "dead") ~
      gtype*dtype + score + wt2,
    data = hodg2)
```

```
hodg.cox1 |> drop1(test="Chisq")
```

```
Single term deletions

Model:
surv ~ gtype * dtype + score + wtime
            Df AIC   LRT Pr(>Chi)
<none>         152
score        1 168 17.24  3.3e-05 ***
wtime        1 154  3.28     0.07 .
gtype:dtype  1 156  5.44     0.02 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
hodg.cox2 |> drop1(test="Chisq")
```

```
Single term deletions

Model:
Surv(time, event = delta == "dead") ~ gtype * dtype + score +
    wt2
            Df AIC   LRT Pr(>Chi)
<none>         149
score        1 169 21.60  3.4e-06 ***
wt2          1 154  6.61    0.010 *
gtype:dtype  1 152  4.97    0.026 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The new model has better (lower) AIC.

0.48 Checking for Outliers and Influential Observations

We will check for outliers using the deviance residuals. The martingale residuals show excess events or the opposite, but highly skewed, with the maximum possible value being 1, but the

smallest value can be very large negative. Martingale residuals can detect unexpectedly long-lived patients, but patients who die unexpectedly early show up only in the deviance residual. Influence will be examined using dfbeta in a similar way to linear regression, logistic regression, or Poisson regression.

0.48.1 Deviance Residuals

$$r_i^D = \text{sign}(r_i^M)\sqrt{-2\left[r_i^M + \delta_i \ln(\delta_i - r_i^M)\right]}$$
$$r_i^D = \text{sign}(r_i^M)\sqrt{-2\left[r_i^M + \delta_i \ln(r_i^{CS})\right]}$$

Roughly centered on 0 with approximate standard deviation 1.

0.48.2

```
hodg.mart = residuals(hodg.cox2,type="martingale")
hodg.dev = residuals(hodg.cox2,type="deviance")
hodg.dfb = residuals(hodg.cox2,type="dfbeta")
hodg.preds = predict(hodg.cox2)                    #linear predictor
```

```
plot(hodg.preds,
     hodg.mart,
     xlab="Linear Predictor",
     ylab="Martingale Residual")
```

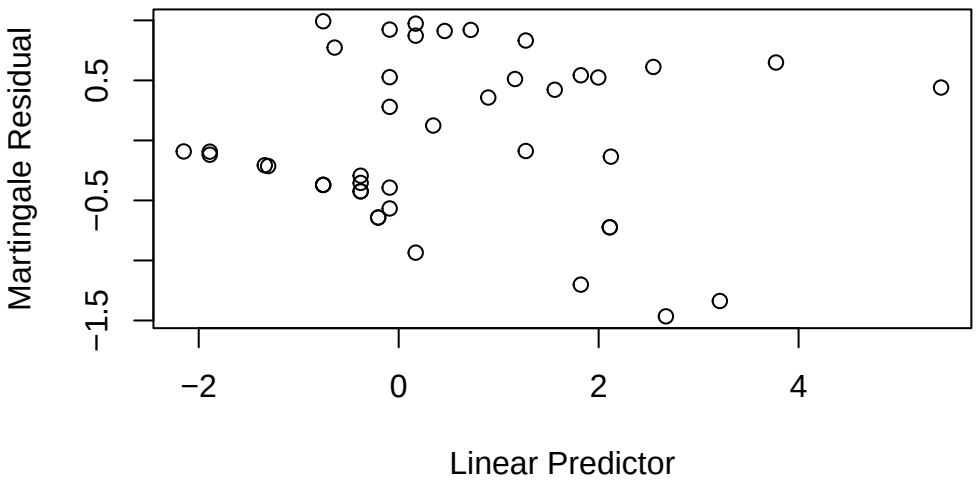


Figure 31: Martingale Residuals vs. Linear Predictor

The smallest three martingale residuals in order are observations 1, 29, and 18.

```
plot(hodg.preds,hodg.dev,xlab="Linear Predictor",ylab="Deviance Residual")
```

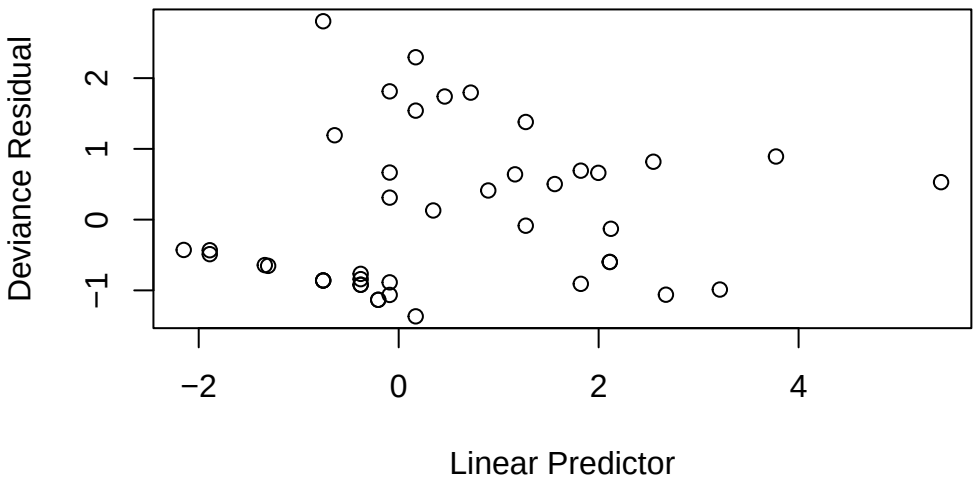


Figure 32: Deviance Residuals vs. Linear Predictor

The two largest deviance residuals are observations 1 and 29. Worth examining.

0.48.3 dfbeta

- dfbeta is the approximate change in the coefficient vector if that observation were dropped
- dfbetas is the approximate change in the coefficients, scaled by the standard error for the coefficients.

0.48.3.1 Graft type

```
plot(hodg.dfb[,1],xlab="Observation Order",ylab="dfbeta for Graft Type")
```

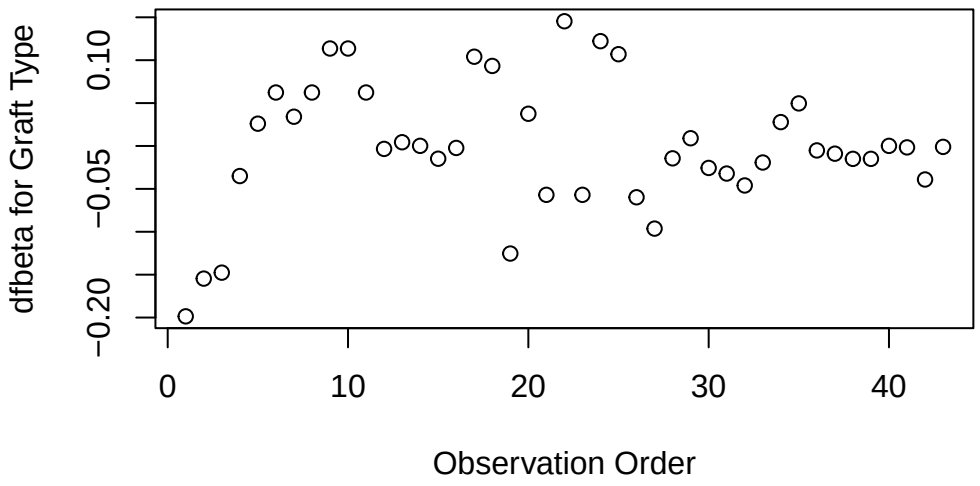


Figure 33: dfbeta Values by Observation Order for Graft Type

The smallest dfbeta for graft type is observation 1.

0.48.3.2 Disease type

```
plot(hodg.dfb[,2],
     xlab="Observation Order",
     ylab="dfbeta for Disease Type")
```

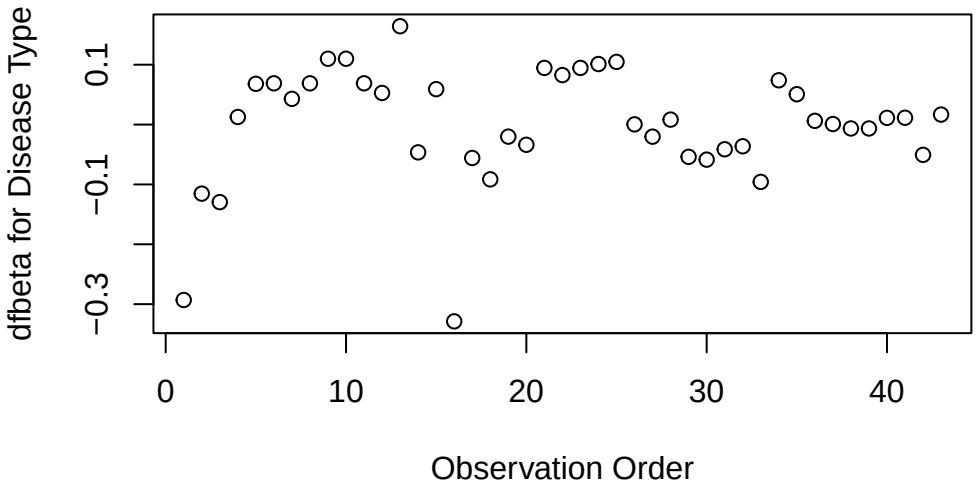


Figure 34: dfbeta Values by Observation Order for Disease Type

The smallest two dfbeta values for disease type are observations 1 and 16.

0.48.3.3 Karnofsky score

```
plot(hodg.dfb[,3],
     xlab="Observation Order",
     ylab="dfbeta for Karnofsky Score")
```

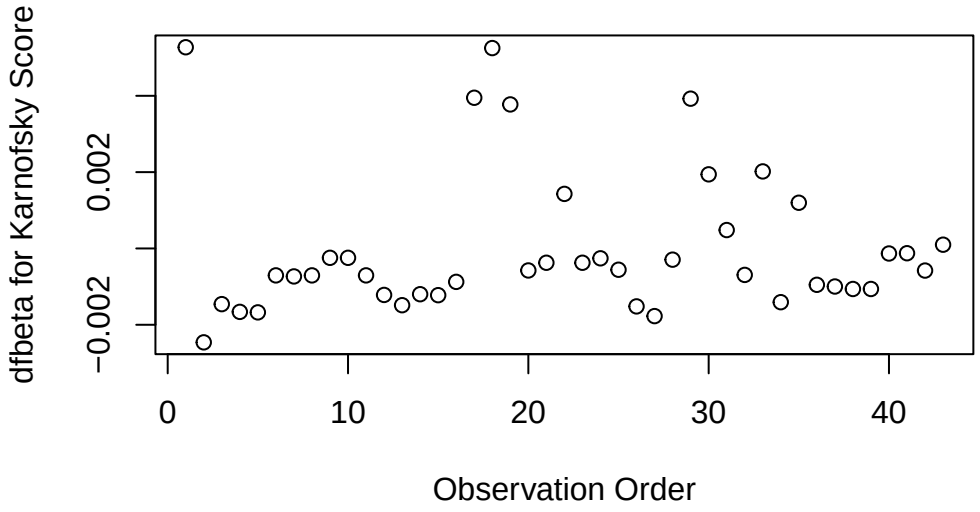


Figure 35: dfbeta Values by Observation Order for Karnofsky Score

The two highest dfbeta values for score are observations 1 and 18. The next three are observations 17, 29, and 19. The smallest value is observation 2.

0.48.3.4 Waiting time (dichotomized)

```
plot(
  hodg.dfb[,4],
  xlab="Observation Order",
  ylab="dfbeta for `Waiting Time < 80`")
```

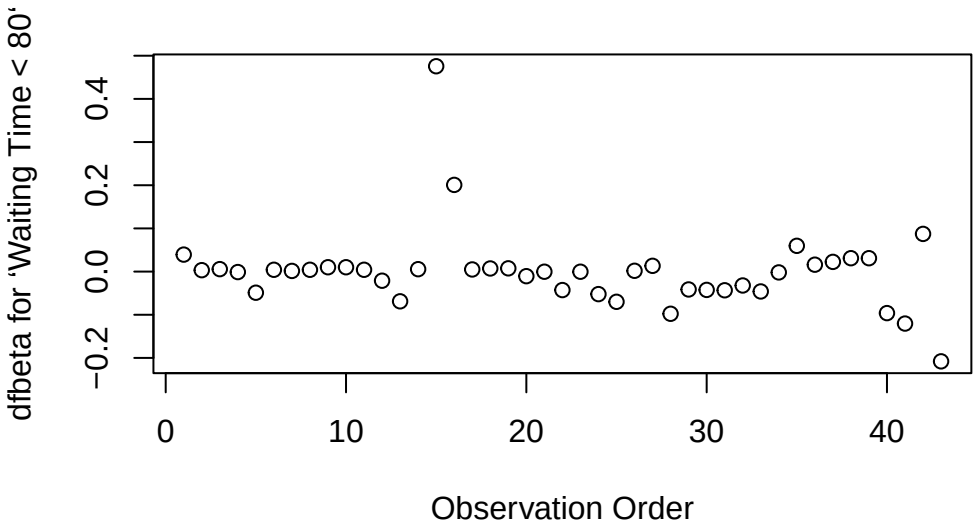


Figure 36: dfbeta Values by Observation Order for Waiting Time (dichotomized)

The two large values of dfbeta for dichotomized waiting time are observations 15 and 16. This may have to do with the discretization of waiting time.

0.48.3.5 Interaction: graft type and disease type

```
plot(hodg.dfb[,5],
     xlab="Observation Order",
     ylab="dfbeta for dtype:gtype")
```

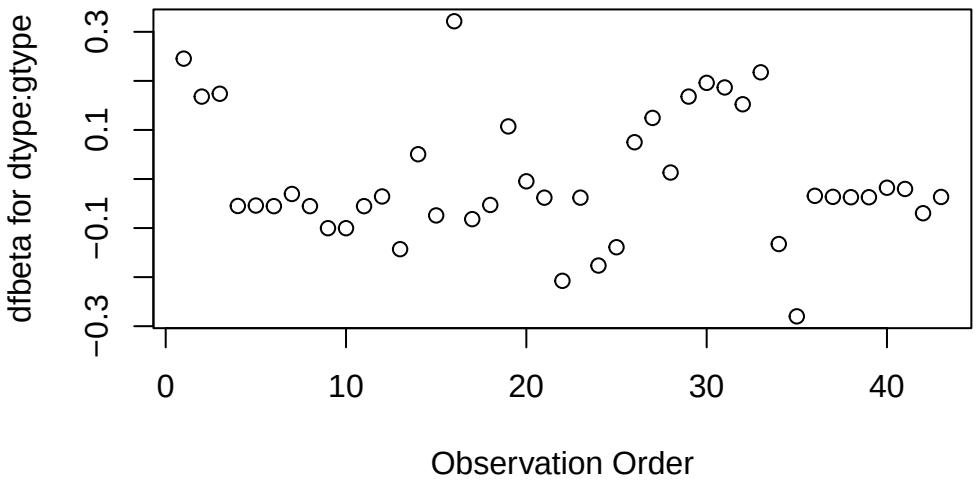


Figure 37: dfbeta Values by Observation Order for dtype:gtype

The two largest values are observations 1 and 16. The smallest value is observation 35.

0.48.4

Table 33: Observations to Examine by Residuals and Influence

Diagnostic	Observations to Examine
Martingale Residuals	1, 29, 18
Deviance Residuals	1, 29
Graft Type Influence	1
Disease Type Influence	1, 16
Karnofsky Score Influence	1, 18 (17, 29, 19)
Waiting Time Influence	15, 16
Graft by Disease Influence	1, 16, 35

The most important observations to examine seem to be 1, 15, 16, 18, and 29.

0.48.5

```
with(hodg,summary(time[delta==1]))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    2.0    41.2    62.5    97.6    83.2   524.0
```

```
with(hodg,summary(wtime))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    5.0    16.0    24.0    37.7    55.5   171.0
```

```
with(hodg,summary(score))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   20.0    60.0    80.0    76.3    90.0   100.0
```

```
hodg.cox2
```

```
Call:
coxph(formula = Surv(time, event = delta == "dead") ~ gtype *
    dtype + score + wt2, data = hodg2)

                               coef exp(coef) se(coef)  z     p
gtypeAutologous                0.67      1.94     0.59  1 0.263
dtypeHodgkins                  2.33     10.25     0.73  3 0.002
score                         -0.06      0.95     0.01 -4 8e-06
wt2long                       -2.06      0.13     1.05 -2 0.050
gtypeAutologous:dtypeHodgkins -2.07      0.13     0.93 -2 0.026

Likelihood ratio test=35  on 5 df, p=1e-06
n= 43, number of events= 26
```

```
hodg2[c(1,15,16,18,29),] |>
  select(gtype, dtype, time, delta, score, wtime) |>
  mutate(
    comment =
```

```
      c(
        "early death, good score, low risk",
        "high risk grp, long wait, poor score",
        "high risk grp, short wait, poor score",
        "early death, good score, med risk grp",
        "early death, good score, med risk grp"
      ))
```

```
# A tibble: 5 x 7
  gtype      dtype         time delta score wtime comment
  <chr>      <fct>        <int> <chr> <int> <int> <chr>
1 Allogenic  Non-Hodgkins    28 dead     90    24 early death, good score, low ~
2 Allogenic  Hodgkins        77 dead     60   102 high risk grp, long wait, poo~
3 Allogenic  Hodgkins        79 dead     70    71 high risk grp, short wait, po~
4 Autologous Non-Hodgkins    53 dead     90    17 early death, good score, med ~
5 Autologous Hodgkins        30 dead     90    73 early death, good score, med ~
```

0.48.6 Action Items

- Unusual points may need checking, particularly if the data are not completely cleaned. In this case, observations 15 and 16 may show some trouble with the dichotomization of waiting time, but it still may be useful.
- The two largest residuals seem to be due to unexpectedly early deaths, but unfortunately this can occur.
- If hazards don't look proportional, then we may need to use strata, between which the base hazards are permitted to be different. For this problem, the natural strata are the two diseases, because they could need to be managed differently anyway.
- A main point that we want to be sure of is the relative risk difference by disease type and graft type.

```
hodg.cox2 |>
  predict(
    reference = "zero",
    newdata = means |>
      mutate(
        wt2 = "short",
        score = 0),
    type = "lp") |>
  data.frame('linear predictor' = _) |>
  pander()
```

Table 34: Linear Risk Predictors for Lymphoma

	linear.predictor
Non-Hodgkins,Allogenic	0
Non-Hodgkins,Autologous	0.6651
Hodgkins,Allogenic	2.327
Hodgkins,Autologous	0.9256

For Non-Hodgkin's, the allogenic graft is better. For Hodgkin's, the autologous graft is much better.

0.49 Stratified survival models

0.49.1 Revisiting the leukemia dataset (`anderson`)

We will analyze remission survival times on 42 leukemia patients, half on new treatment, half on standard treatment.

This is the same data as the `drug6mp` data from `KMsurv`, but with two other variables and without the pairing. This version comes from the Kleinbaum and Klein survival textbook (e.g., p281):

```
anderson =
  paste0(
    "http://web1.sph.emory.edu/dkleinb/allDatasets/",
    "surv2datasets/anderson.dta") |>
  haven::read_dta() |>
  mutate(
    status = status |>
      case_match(
        1 ~ "relapse",
        0 ~ "censored"
      ),

    sex = sex |>
      case_match(
        0 ~ "female",
        1 ~ "male"
      ) |>
      factor() |>
```

```
      relevel(ref = "female"),

    rx = rx |>
      case_match(
        0 ~ "new",
        1 ~ "standard"
      ) |>
      factor() |> relevel(ref = "standard"),

    surv = Surv(
      time = survt,
      event = (status == "relapse"))
  )

print(anderson)
```

```
# A tibble: 42 x 7
   survt status   sex    logwbc rx    lwbc3   surv
   <dbl> <chr>    <fct>   <dbl> <fct> <dbl> <Surv>
 1    35 censored male     1.45 new       1    35+
 2    34 censored male     1.47 new       1    34+
 3    32 censored male     2.2  new       1    32+
 4    32 censored male     2.53 new       2    32+
 5    25 censored male     1.78 new       1    25+
 6    23 relapse  male     2.57 new       2    23
 7    22 relapse  male     2.32 new       2    22
 8    20 censored male     2.01 new       1    20+
 9    19 censored female   2.05 new       1    19+
10    17 censored female   2.16 new       1    17+
# i 32 more rows
```

0.49.2 Cox semi-parametric proportional hazards model

```
anderson.cox1 = coxph(
  formula = surv ~ rx + sex + logwbc,
  data = anderson)

summary(anderson.cox1)
```

```
Call:
```

```
coxph(formula = surv ~ rx + sex + logwbc, data = anderson)

  n= 42, number of events= 30

          coef exp(coef) se(coef)     z Pr(>|z|)
rxnew   -1.504     0.222    0.462 -3.26   0.0011 **
sexmale  0.315     1.370    0.455  0.69   0.4887
logwbc   1.682     5.376    0.337  5.00  5.8e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

        exp(coef) exp(-coef) lower .95 upper .95
rxnew       0.222      4.498     0.090     0.549
sexmale     1.370      0.730     0.562     3.338
logwbc      5.376      0.186     2.779    10.398

Concordance= 0.851  (se = 0.041 )
Likelihood ratio test= 47.2  on 3 df,   p=3e-10
Wald test            = 33.5  on 3 df,   p=2e-07
Score (logrank) test = 48  on 3 df,   p=2e-10
```

0.49.2.1 Test the proportional hazards assumption

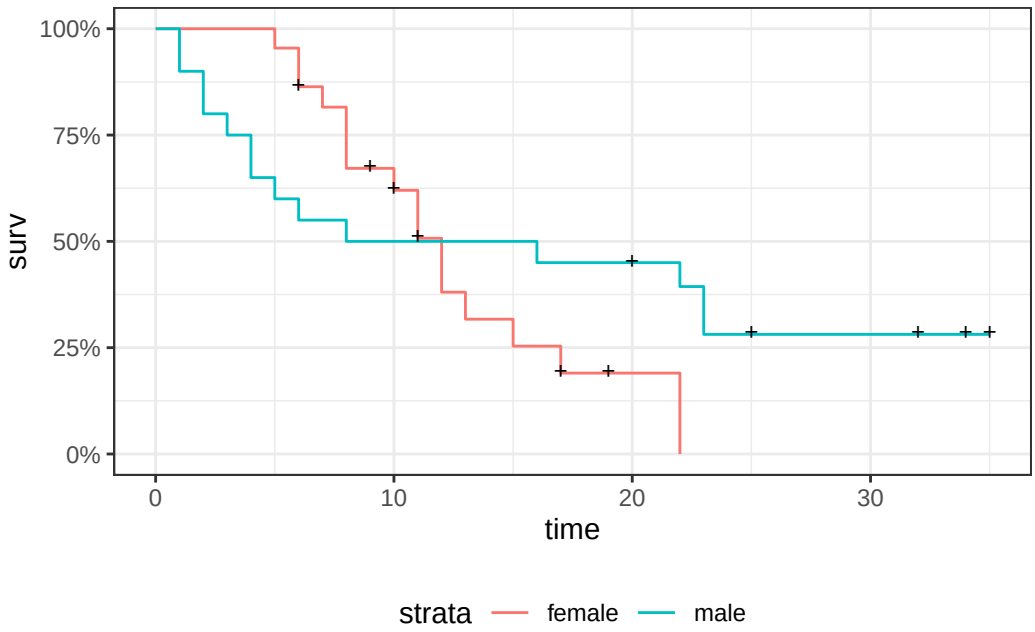
```
cox.zph(anderson.cox1)
```

```
       chisq df    p
rx     0.036  1 0.85
sex    5.420  1 0.02
logwbc 0.142  1 0.71
GLOBAL 5.879  3 0.12
```

0.49.2.2 Graph the K-M survival curves

```
anderson_km_model = survfit(
  formula = surv ~ sex,
  data = anderson)

anderson_km_model |>
  autoplot(conf.int = FALSE) +
```

```
  theme_bw() +
  theme(legend.position="bottom")
```



The survival curves cross, which indicates a problem in the proportionality assumption by sex.

0.49.3 Graph the Nelson-Aalen cumulative hazard

We can also look at the log-hazard ("cloglog survival") plots:

```
anderson_na_model = survfit(
  formula = surv ~ sex,
  data = anderson,
  type = "fleming")

anderson_na_model |>
  autoplot(
    fun = "cumhaz",
    conf.int = FALSE) +
  theme_classic() +
```

```
theme(legend.position="bottom") +
ylab("log(Cumulative Hazard)") +
scale_y_continuous(
  trans = "log10",
  name = "Cumulative hazard (H(t), log scale)") +
scale_x_continuous(
  breaks = c(1,2,5,10,20,50),
  trans = "log"
)
```

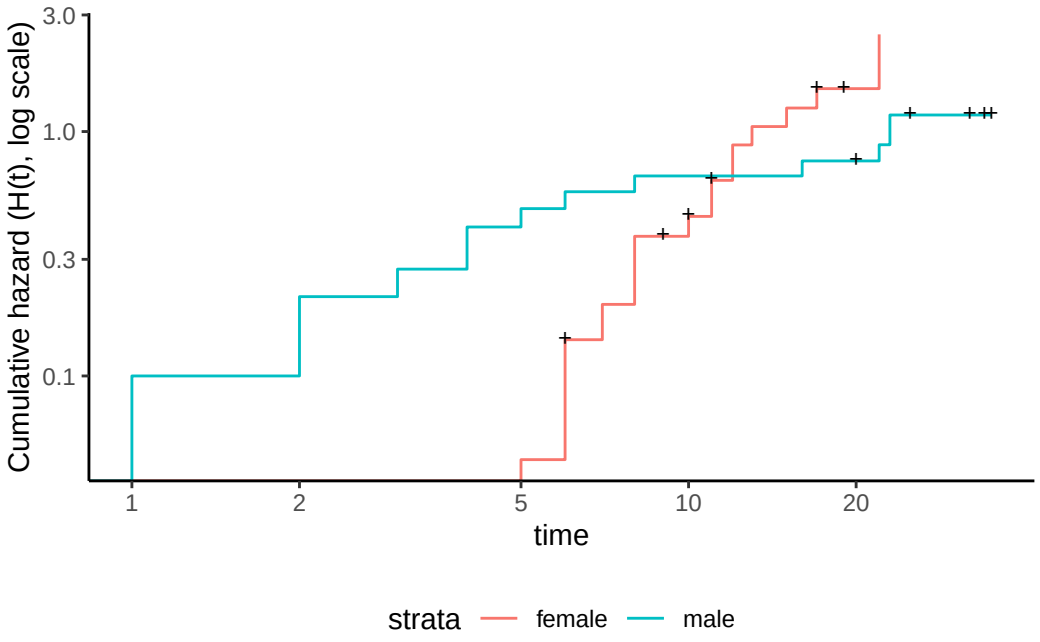


Figure 38: Cumulative hazard (cloglog scale) for `anderson` data

This can be fixed by using strata or possibly by other model alterations.

0.49.4 The Stratified Cox Model

- In a stratified Cox model, each stratum, defined by one or more factors, has its own base survival function $h_0(t)$.
- But the coefficients for each variable not used in the strata definitions are assumed to be the same across strata.

- To check if this assumption is reasonable one can include interactions with strata and see if they are significant (this may generate a warning and NA lines but these can be ignored).
- Since the `sex` variable shows possible non-proportionality, we try stratifying on `sex`.

```
anderson.coxph.strat =
  coxph(
    formula =
      surv ~ rx + logwbc + strata(sex),
    data = anderson)

summary(anderson.coxph.strat)
```

```
Call:
coxph(formula = surv ~ rx + logwbc + strata(sex), data = anderson)

  n= 42, number of events= 30

         coef exp(coef) se(coef)     z Pr(>|z|)
rxnew  -0.998     0.369    0.474 -2.11    0.035 *
logwbc  1.454     4.279    0.344  4.22  2.4e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

       exp(coef) exp(-coef) lower .95 upper .95
rxnew      0.369      2.713     0.146     0.932
logwbc     4.279      0.234     2.180     8.398

Concordance= 0.812  (se = 0.059 )
Likelihood ratio test= 32.1  on 2 df,   p=1e-07
Wald test            = 22.8  on 2 df,   p=1e-05
Score (logrank) test = 30.8  on 2 df,   p=2e-07
```

Let's compare this to a model fit only on the subset of males:

```
anderson.coxph.male =
  coxph(
    formula = surv ~ rx + logwbc,
    subset = sex == "male",
    data = anderson)
```

```
summary(anderson.coxph.male)
```

```
Call:
coxph(formula = surv ~ rx + logwbc, data = anderson, subset = sex ==
    "male")

  n= 20, number of events= 14

         coef exp(coef) se(coef)     z Pr(>|z|)
rxnew  -1.978     0.138    0.739 -2.68   0.0075 **
logwbc  1.743     5.713    0.536  3.25   0.0011 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

       exp(coef) exp(-coef) lower .95 upper .95
rxnew      0.138      7.227    0.0325     0.589
logwbc     5.713      0.175    1.9991    16.328

Concordance= 0.905  (se = 0.043 )
Likelihood ratio test= 29.2  on 2 df,   p=5e-07
Wald test            = 15.3  on 2 df,   p=5e-04
Score (logrank) test = 26.4  on 2 df,   p=2e-06
```

```
anderson.coxph.female =
  coxph(
    formula =
      surv ~ rx + logwbc,
    subset = sex == "female",
    data = anderson)

summary(anderson.coxph.female)
```

```
Call:
coxph(formula = surv ~ rx + logwbc, data = anderson, subset = sex ==
    "female")

  n= 22, number of events= 16

         coef exp(coef) se(coef)     z Pr(>|z|)
rxnew  -0.311     0.733    0.564 -0.55    0.581
```

```
logwbc  1.206     3.341    0.503  2.40    0.017 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

       exp(coef) exp(-coef) lower .95 upper .95
rxnew      0.733      1.365     0.243      2.21
logwbc     3.341      0.299     1.245      8.96

Concordance= 0.692  (se = 0.085 )
Likelihood ratio test= 6.65  on 2 df,   p=0.04
Wald test            = 6.36  on 2 df,   p=0.04
Score (logrank) test = 6.74  on 2 df,   p=0.03
```

The coefficients of treatment look different. Are they statistically different?

```
anderson.coxph.strat.intxn =
  coxph(
    formula = surv ~ strata(sex) * (rx + logwbc),
    data = anderson)

anderson.coxph.strat.intxn |> summary()
```

```
Call:
coxph(formula = surv ~ strata(sex) * (rx + logwbc), data = anderson)

  n= 42, number of events= 30

                         coef exp(coef) se(coef)     z Pr(>|z|)
rxnew                  -0.311     0.733    0.564 -0.55    0.581
logwbc                  1.206     3.341    0.503  2.40    0.017 *
strata(sex)male:rxnew  -1.667     0.189    0.930 -1.79    0.073 .
strata(sex)male:logwbc  0.537     1.710    0.735  0.73    0.465
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                       exp(coef) exp(-coef) lower .95 upper .95
rxnew                      0.733      1.365    0.2427      2.21
logwbc                     3.341      0.299    1.2452      8.96
strata(sex)male:rxnew      0.189      5.294    0.0305      1.17
strata(sex)male:logwbc     1.710      0.585    0.4048      7.23

Concordance= 0.797  (se = 0.058 )
```

```
Likelihood ratio test= 35.8  on 4 df,   p=3e-07
Wald test            = 21.7  on 4 df,   p=2e-04
Score (logrank) test = 33.1  on 4 df,   p=1e-06
```

```
anova(
  anderson.coxph.strat.intxn,
  anderson.coxph.strat)
```

```
Analysis of Deviance Table
 Cox model: response is  surv
 Model 1: ~ strata(sex) * (rx + logwbc)
 Model 2: ~ rx + logwbc + strata(sex)
  loglik Chisq Df Pr(>|Chi|)
1  -53.9
2  -55.7  3.77  2       0.15
```

We don't have enough evidence to tell the difference between these two models.

0.49.5 Conclusions

- We chose to use a stratified model because of the apparent non-proportionality of the hazard for the sex variable.
- When we fit interactions with the strata variable, we did not get an improved model (via the likelihood ratio test).
- So we use the stratifed model with coefficients that are the same across strata.

0.49.6 Another Modeling Approach

- We used an additive model without interactions and saw that we might need to stratify by sex.
- Instead, we could try to improve the model's functional form - maybe the interaction of treatment and sex is real, and after fitting that we might not need separate hazard functions.
- Either approach may work.

```
anderson.coxph.intxn =
  coxph(
    formula = surv ~ (rx + logwbc) * sex,
    data = anderson)
```

```
anderson.coxph.intxn |> summary()
```

```
Call:
coxph(formula = surv ~ (rx + logwbc) * sex, data = anderson)

  n= 42, number of events= 30

                 coef exp(coef) se(coef)     z Pr(>|z|)
rxnew          -0.3748    0.6874   0.5545 -0.68    0.499
logwbc          1.0637    2.8971   0.4726  2.25    0.024 *
sexmale        -2.8052    0.0605   2.0323 -1.38    0.167
rxnew:sexmale  -2.1782    0.1132   0.9109 -2.39    0.017 *
logwbc:sexmale  1.2303    3.4223   0.6301  1.95    0.051 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

               exp(coef) exp(-coef) lower .95 upper .95
rxnew             0.6874      1.455   0.23185     2.038
logwbc            2.8971      0.345   1.14730     7.315
sexmale           0.0605     16.531   0.00113     3.248
rxnew:sexmale     0.1132      8.830   0.01899     0.675
logwbc:sexmale    3.4223      0.292   0.99539    11.766

Concordance= 0.861  (se = 0.036 )
Likelihood ratio test= 57  on 5 df,   p=5e-11
Wald test            = 35.6  on 5 df,   p=1e-06
Score (logrank) test = 57.1  on 5 df,   p=5e-11
```

```
cox.zph(anderson.coxph.intxn)
```

```
           chisq df    p
rx         0.136  1 0.71
logwbc     1.652  1 0.20
sex        1.266  1 0.26
rx:sex     0.149  1 0.70
logwbc:sex 0.102  1 0.75
GLOBAL     3.747  5 0.59
```

0.50 Time-varying covariates

(adapted from Klein and Moeschberger 2003, §9.2: https://link.springer.com/book/10.1007/b97377)

0.50.1 Motivating example: back to the leukemia dataset

```
# load the data:
data(bmt, package = 'KMsurv')
bmt |> as_tibble() |> print(n = 5)
```

```
# A tibble: 137 x 22
  group    t1    t2    d1    d2    d3    ta    da    tc    dc    tp    dp    z1
  <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
1     1  2081  2081     0     0     0    67     1   121     1    13     1    26
2     1  1602  1602     0     0     0  1602     0   139     1    18     1    21
3     1  1496  1496     0     0     0  1496     0   307     1    12     1    26
4     1  1462  1462     0     0     0    70     1    95     1    13     1    17
5     1  1433  1433     0     0     0  1433     0   236     1    12     1    32
# i 132 more rows
# i 9 more variables: z2 <int>, z3 <int>, z4 <int>, z5 <int>, z6 <int>,
#   z7 <int>, z8 <int>, z9 <int>, z10 <int>
```

This dataset comes from the Copelan et al. (1991) study of allogenic bone marrow transplant therapy for acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL).

Outcomes (endpoints)

- The main endpoint is disease-free survival (`t2` and `d3`) for the three risk groups, "ALL", "AML Low Risk", and "AML High Risk".

Possible intermediate events

- graft vs. host disease (**GVHD**), an immunological rejection response to the transplant (bad)
- acute (**AGVHD**)
- chronic (**CGVHD**)
- platelet recovery, a return of platelet count to normal levels (good)

One or the other, both in either order, or neither may occur.

Covariates

- We are interested in possibly using the covariates `z1-z10` to adjust for other factors.
- In addition, the time-varying covariates for acute GVHD, chronic GVHD, and platelet recovery may be useful.

0.50.1.1 Preprocessing

We add reformat the data before analysis:

```
# reformat the data:
bmt1 =
  bmt |>
  as_tibble() |>
  mutate(
    id = 1:n(), # will be used to connect multiple records for the same individual

    group =  group |>
      case_match(
        1 ~ "ALL",
        2 ~ "Low Risk AML",
        3 ~ "High Risk AML") |>
      factor(levels = c("ALL", "Low Risk AML", "High Risk AML")),

    `patient age` = z1,

    `donor age` = z2,

    `patient sex` = z3 |>
      case_match(
        0 ~ "Female",
        1 ~ "Male"),

    `donor sex` = z4 |>
      case_match(
        0 ~ "Female",
        1 ~ "Male"),

    `Patient CMV Status` = z5 |>
      case_match(
        0 ~ "CMV Negative",
```

```
        1 ~ "CMV Positive"),

    `Donor CMV Status` = z6 |>
      case_match(
        0 ~ "CMV Negative",
        1 ~ "CMV Positive"),

    `Waiting Time to Transplant` = z7,

    FAB = z8 |>
      case_match(
        1 ~ "Grade 4 Or 5 (AML only)",
        0 ~ "Other") |>
      factor() |>
      relevel(ref = "Other"),

    hospital = z9 |>  # `z9` is hospital
      case_match(
        1 ~ "Ohio State University",
        2 ~ "Alferd",
        3 ~ "St. Vincent",
        4 ~ "Hahnemann") |>
      factor() |>
      relevel(ref = "Ohio State University"),

    MTX = (z10 == 1) # a prophylatic treatment for GVHD

  ) |>
  select(-(z1:z10)) # don't need these anymore

bmt1 |>
  select(group, id:MTX) |>
  print(n = 10)
```

```
# A tibble: 137 x 12
   group    id `patient age` `donor age` `patient sex` `donor sex`
   <fct> <int>         <int>       <int> <chr>         <chr>
 1 ALL       1            26          33 Male          Female
 2 ALL       2            21          37 Male          Male
 3 ALL       3            26          35 Male          Male
 4 ALL       4            17          21 Female        Male
 5 ALL       5            32          36 Male          Male
```

```
 6 ALL       6            22          31 Male          Male
 7 ALL       7            20          17 Male          Female
 8 ALL       8            22          24 Male          Female
 9 ALL       9            18          21 Female        Male
10 ALL      10            24          40 Male          Male
# i 127 more rows
# i 6 more variables: `Patient CMV Status` <chr>, `Donor CMV Status` <chr>,
#   `Waiting Time to Transplant` <int>, FAB <fct>, hospital <fct>, MTX <lgl>
```

0.50.2 Time-Dependent Covariates

- A **time-dependent covariate** ("**TDC**") is a covariate whose value changes during the course of the study.
- For variables like age that change in a linear manner with time, we can just use the value at the start.
- But it may be plausible that when and if GVHD occurs, the risk of relapse or death increases, and when and if platelet recovery occurs, the risk decreases.

0.50.3 Analysis in R

- We form a variable `precovery` which is = 0 before platelet recovery and is = 1 after platelet recovery, if it occurs.
- For each subject where platelet recovery occurs, we set up multiple records (lines in the data frame); for example one from t = 0 to the time of platelet recovery, and one from that time to relapse, recovery, or death.
- We do the same for acute GVHD and chronic GVHD.
- For each record, the covariates are constant.

```
bmt2 = bmt1 |>
  #set up new long-format data set:
  tmerge(bmt1, id = id, tstop = t2) |>

  # the following three steps can be in any order,
  # and will still produce the same result:
  #add aghvd as tdc:
  tmerge(bmt1, id = id, agvhd = tdc(ta)) |>
  #add cghvd as tdc:
  tmerge(bmt1, id = id, cgvhd = tdc(tc)) |>
  #add platelet recovery as tdc:
  tmerge(bmt1, id = id, precovery = tdc(tp))
```

```
bmt2 = bmt2 |>
  as_tibble() |>
  mutate(status = as.numeric((tstop == t2) & d3))
# status only = 1 if at end of t2 and not censored
```

Let's see how we've rearranged the first row of the data:

```
bmt1 |>
  filter(id == 1) |>
  select(id, t1, d1, t2, d2, d3, ta, da, tc, dc, tp, dp)
```

```
# A tibble: 1 x 12
     id    t1    d1    t2    d2    d3    ta    da    tc    dc    tp    dp
  <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
1     1  2081     0  2081     0     0    67     1   121     1    13     1
```

The event times for this individual are:

- `t` = 0 time of transplant
- `tp` = 13 platelet recovery
- `ta` = 67 acute GVHD onset
- `tc` = 121 chronic GVHD onset
- `t2` = 2081 end of study, patient not relapsed or dead

After converting the data to long-format, we have:

```
bmt2 |>
  select(
    id,
    tstart,
    tstop,
    agvhd,
    cgvhd,
    precovery,
    status
  ) |>
  filter(id == 1)
```

```
# A tibble: 4 x 7
     id tstart tstop agvhd cgvhd precovery status
  <int>  <dbl> <int> <int> <int>     <int>  <dbl>
```

```
1      1       0    13      0      0          0       0
2      1      13    67      0      0          1       0
3      1      67   121      1      0          1       0
4      1     121  2081      1      1          1       0
```

Note that `status` could have been `1` on the last row, indicating that relapse or death occurred; since it is false, the participant must have exited the study without experiencing relapse or death (i.e., they were censored).

0.50.4 Event sequences

Let:

- A = acute GVHD
- C = chronic GVHD
- P = platelet recovery

Each of the eight possible combinations of A or not-A, with C or not-C, with P or not-P occurs in this data set.

- A always occurs before C, and P always occurs before C, if both occur.
- Thus there are ten event sequences in the data set: None, A, C, P, AC, AP, PA, PC, APC, and PAC.
- In general, there could be as many as $1 + 3 + (3)(2) + 6 = 16$ sequences, but our domain knowledge tells us that some are missing: CA, CP, CAP, CPA, PCA, PC, PAC
- Different subjects could have 1, 2, 3, or 4 intervals, depending on which of acute GVHD, chronic GVHD, and/or platelet recovery occurred.
- The final interval for any subject has `status` = 1 if the subject relapsed or died at that time; otherwise `status` = 0.
- Any earlier intervals have `status` = 0.
- Even though there might be multiple lines per ID in the dataset, there is never more than one event, so no alterations need be made in the estimation procedures or in the interpretation of the output.
- The function `tmerge` in the `survival` package eases the process of constructing the new long-format dataset.

0.50.5 Model with Time-Fixed Covariates

```
bmt1 =
  bmt1 |>
  mutate(surv = Surv(t2,d3))
```

```r
bmt_coxph_TF = coxph(
  formula = surv ~ group + `patient age`*`donor age` + FAB,
  data = bmt1)
summary(bmt_coxph_TF)
```

```
Call:
coxph(formula = surv ~ group + `patient age` * `donor age` +
    FAB, data = bmt1)

  n= 137, number of events= 83

                               coef exp(coef)  se(coef)     z Pr(>|z|)
groupLow Risk AML          -1.090648  0.335999  0.354279 -3.08  0.00208 **
groupHigh Risk AML         -0.403905  0.667707  0.362777 -1.11  0.26555
`patient age`              -0.081639  0.921605  0.036107 -2.26  0.02376 *
`donor age`                -0.084587  0.918892  0.030097 -2.81  0.00495 **
FABGrade 4 Or 5 (AML only)  0.837416  2.310388  0.278464  3.01  0.00264 **
`patient age`:`donor age`   0.003159  1.003164  0.000951  3.32  0.00089 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                           exp(coef) exp(-coef) lower .95 upper .95
groupLow Risk AML              0.336      2.976     0.168     0.673
groupHigh Risk AML             0.668      1.498     0.328     1.360
`patient age`                  0.922      1.085     0.859     0.989
`donor age`                    0.919      1.088     0.866     0.975
FABGrade 4 Or 5 (AML only)     2.310      0.433     1.339     3.988
`patient age`:`donor age`      1.003      0.997     1.001     1.005

Concordance= 0.665  (se = 0.033 )
Likelihood ratio test= 32.8  on 6 df,   p=1e-05
Wald test            = 33  on 6 df,   p=1e-05
Score (logrank) test = 35.8  on 6 df,   p=3e-06
```

```r
drop1(bmt_coxph_TF, test = "Chisq")
```

```
Single term deletions

Model:
surv ~ group + `patient age` * `donor age` + FAB
```

```
                          Df AIC   LRT Pr(>Chi)
<none>                       726
group                     2 734 12.51   0.0019 **
FAB                       1 733  9.22   0.0024 **
`patient age`:`donor age` 1 733  9.51   0.0020 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
bmt1$mres =
  bmt_coxph_TF |>
  update(. ~ . - `donor age`) |>
  residuals(type="martingale")

bmt1 |>
  ggplot(aes(x = `donor age`, y = mres)) +
  geom_point() +
  geom_smooth(method = "loess", aes(col = "loess")) +
  geom_smooth(method = 'lm', aes(col = "lm")) +
  theme_classic() +
  xlab("Donor age") +
  ylab("Martingale Residuals") +
  guides(col=guide_legend(title = ""))
```

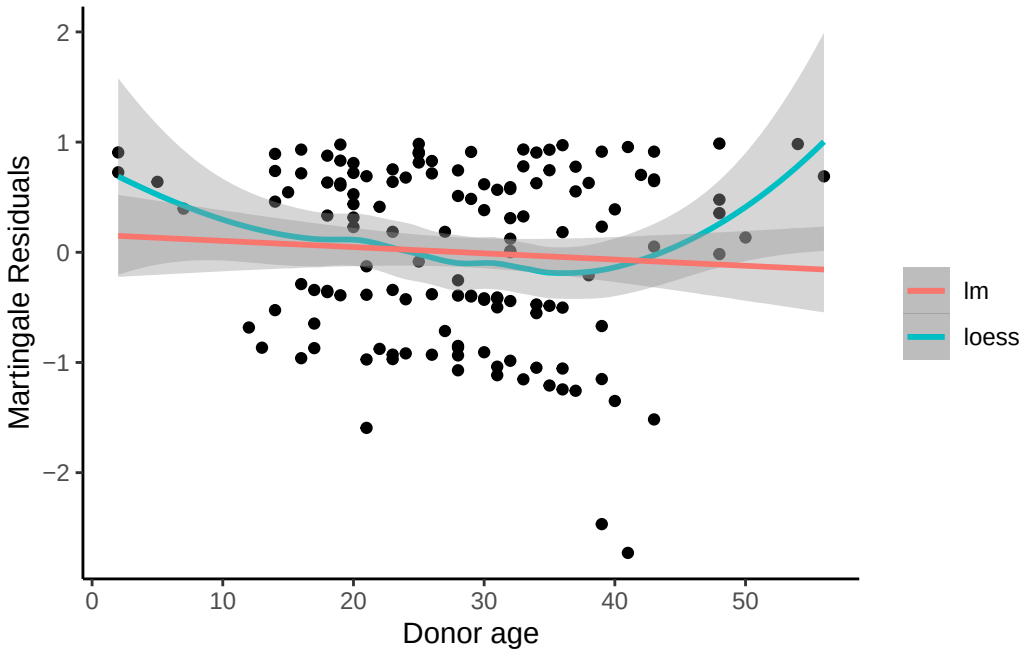


Figure 39: Martingale residuals for `Donor age`

A more complex functional form for `donor age` seems warranted; left as an exercise for the reader.

Now we will add the time-varying covariates:

```
# add counting process formulation of Surv():
bmt2 =
  bmt2 |>
  mutate(
    surv =
      Surv(
        time = tstart,
        time2 = tstop,
        event = status,
        type = "counting"))
```

0.50.6 Model with Time-Dependent Covariates

```
bmt_coxph_TV = coxph(
  formula =
    surv ~
    group + `patient age`*`donor age` + FAB + agvhd + cgvhd + precovery,
  data = bmt2)

summary(bmt_coxph_TV)
```

```
Call:
coxph(formula = surv ~ group + `patient age` * `donor age` +
    FAB + agvhd + cgvhd + precovery, data = bmt2)

  n= 341, number of events= 83

                               coef exp(coef)  se(coef)     z Pr(>|z|)
groupLow Risk AML          -1.038514  0.353980  0.358220 -2.90   0.0037 **
groupHigh Risk AML         -0.380481  0.683533  0.374867 -1.01   0.3101
`patient age`              -0.073351  0.929275  0.035956 -2.04   0.0413 *
`donor age`                -0.076406  0.926440  0.030196 -2.53   0.0114 *
FABGrade 4 Or 5 (AML only)  0.805700  2.238263  0.284273  2.83   0.0046 **
agvhd                       0.150565  1.162491  0.306848  0.49   0.6237
cgvhd                      -0.116136  0.890354  0.289046 -0.40   0.6878
precovery                  -0.941123  0.390190  0.347861 -2.71   0.0068 **
`patient age`:`donor age`   0.002895  1.002899  0.000944  3.07   0.0022 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                           exp(coef) exp(-coef) lower .95 upper .95
groupLow Risk AML              0.354      2.825     0.175     0.714
groupHigh Risk AML             0.684      1.463     0.328     1.425
`patient age`                  0.929      1.076     0.866     0.997
`donor age`                    0.926      1.079     0.873     0.983
FABGrade 4 Or 5 (AML only)     2.238      0.447     1.282     3.907
agvhd                          1.162      0.860     0.637     2.121
cgvhd                          0.890      1.123     0.505     1.569
precovery                      0.390      2.563     0.197     0.772
`patient age`:`donor age`      1.003      0.997     1.001     1.005

Concordance= 0.702  (se = 0.028 )
```

```
Likelihood ratio test= 40.3  on 9 df,   p=7e-06
Wald test            = 42.4  on 9 df,   p=3e-06
Score (logrank) test = 47.2  on 9 df,   p=4e-07
```

Platelet recovery is highly significant.

Neither acute GVHD (`agvhd`) nor chronic GVHD (`cgvhd`) has a statistically significant effect here, nor are they significant in models with the other one removed.

```
update(bmt_coxph_TV, .~.-agvhd) |> summary()
```

```
Call:
coxph(formula = surv ~ group + `patient age` + `donor age` +
    FAB + cgvhd + precovery + `patient age`:`donor age`, data = bmt2)

  n= 341, number of events= 83

                               coef exp(coef)  se(coef)     z Pr(>|z|)
groupLow Risk AML          -1.049870  0.349983  0.356727 -2.94   0.0032 **
groupHigh Risk AML         -0.417049  0.658988  0.365348 -1.14   0.2537
`patient age`              -0.070749  0.931696  0.035477 -1.99   0.0461 *
`donor age`                -0.075693  0.927101  0.030075 -2.52   0.0118 *
FABGrade 4 Or 5 (AML only)  0.807035  2.241253  0.283437  2.85   0.0044 **
cgvhd                      -0.095393  0.909015  0.285979 -0.33   0.7387
precovery                  -0.983653  0.373942  0.338170 -2.91   0.0036 **
`patient age`:`donor age`   0.002859  1.002863  0.000936  3.05   0.0023 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                           exp(coef) exp(-coef) lower .95 upper .95
groupLow Risk AML              0.350      2.857     0.174     0.704
groupHigh Risk AML             0.659      1.517     0.322     1.349
`patient age`                  0.932      1.073     0.869     0.999
`donor age`                    0.927      1.079     0.874     0.983
FABGrade 4 Or 5 (AML only)     2.241      0.446     1.286     3.906
cgvhd                          0.909      1.100     0.519     1.592
precovery                      0.374      2.674     0.193     0.726
`patient age`:`donor age`      1.003      0.997     1.001     1.005

Concordance= 0.701  (se = 0.027 )
Likelihood ratio test= 40  on 8 df,   p=3e-06
Wald test            = 42.4  on 8 df,   p=1e-06
```

```
Score (logrank) test = 47.2  on 8 df,   p=1e-07
```

```
update(bmt_coxph_TV, .~.-cgvhd) |> summary()
```

```
Call:
coxph(formula = surv ~ group + `patient age` + `donor age` +
    FAB + agvhd + precovery + `patient age`:`donor age`, data = bmt2)

  n= 341, number of events= 83

                               coef exp(coef)  se(coef)     z Pr(>|z|)
groupLow Risk AML          -1.019638  0.360725  0.355311 -2.87   0.0041 **
groupHigh Risk AML         -0.381356  0.682935  0.374568 -1.02   0.3086
`patient age`              -0.073189  0.929426  0.035890 -2.04   0.0414 *
`donor age`                -0.076753  0.926118  0.030121 -2.55   0.0108 *
FABGrade 4 Or 5 (AML only)  0.811716  2.251769  0.284012  2.86   0.0043 **
agvhd                       0.131621  1.140676  0.302623  0.43   0.6636
precovery                  -0.946697  0.388021  0.347265 -2.73   0.0064 **
`patient age`:`donor age`   0.002904  1.002908  0.000943  3.08   0.0021 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                           exp(coef) exp(-coef) lower .95 upper .95
groupLow Risk AML              0.361      2.772     0.180     0.724
groupHigh Risk AML             0.683      1.464     0.328     1.423
`patient age`                  0.929      1.076     0.866     0.997
`donor age`                    0.926      1.080     0.873     0.982
FABGrade 4 Or 5 (AML only)     2.252      0.444     1.291     3.929
agvhd                          1.141      0.877     0.630     2.064
precovery                      0.388      2.577     0.196     0.766
`patient age`:`donor age`      1.003      0.997     1.001     1.005

Concordance= 0.701  (se = 0.027 )
Likelihood ratio test= 40.1  on 8 df,   p=3e-06
Wald test            = 42.1  on 8 df,   p=1e-06
Score (logrank) test = 47.1  on 8 df,   p=1e-07
```

Let's drop them both:

```
bmt_coxph_TV2 = update(bmt_coxph_TV, . ~ . - agvhd -cgvhd)
bmt_coxph_TV2 |> summary()
```

```
Call:
coxph(formula = surv ~ group + `patient age` + `donor age` +
    FAB + precovery + `patient age`:`donor age`, data = bmt2)

  n= 341, number of events= 83

                               coef exp(coef)  se(coef)     z Pr(>|z|)
groupLow Risk AML          -1.032520  0.356108  0.353202 -2.92   0.0035 **
groupHigh Risk AML         -0.413888  0.661075  0.365209 -1.13   0.2571
`patient age`              -0.070965  0.931495  0.035453 -2.00   0.0453 *
`donor age`                -0.076052  0.926768  0.030007 -2.53   0.0113 *
FABGrade 4 Or 5 (AML only)  0.811926  2.252242  0.283231  2.87   0.0041 **
precovery                  -0.983505  0.373998  0.337997 -2.91   0.0036 **
`patient age`:`donor age`   0.002872  1.002876  0.000936  3.07   0.0021 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

                           exp(coef) exp(-coef) lower .95 upper .95
groupLow Risk AML              0.356      2.808     0.178     0.712
groupHigh Risk AML             0.661      1.513     0.323     1.352
`patient age`                  0.931      1.074     0.869     0.999
`donor age`                    0.927      1.079     0.874     0.983
FABGrade 4 Or 5 (AML only)     2.252      0.444     1.293     3.924
precovery                      0.374      2.674     0.193     0.725
`patient age`:`donor age`      1.003      0.997     1.001     1.005

Concordance= 0.7  (se = 0.027 )
Likelihood ratio test= 39.9  on 7 df,   p=1e-06
Wald test            = 42.2  on 7 df,   p=5e-07
Score (logrank) test = 47.1  on 7 df,   p=5e-08
```

0.51 Recurrent Events

(Adapted from Kleinbaum and Klein, Ch 8)

- Sometimes an appropriate analysis requires consideration of recurrent events.
- A patient with arthritis may have more than one flareup. The same is true of many recurring-remitting diseases.
- In this case, we have more than one line in the data frame, but each line may have an event.
- We have to use a "robust" variance estimator to account for correlation of time-to-events within a patient.

0.51.1 Bladder Cancer Data Set

The bladder cancer dataset from Kleinbaum and Klein contains recurrent event outcome information for eighty-six cancer patients followed for the recurrence of bladder cancer tumor after transurethral surgical excision (Byar and Green 1980). The exposure of interest is the effect of the drug treatment of thiotepa. Control variables are the initial number and initial size of tumors. The data layout is suitable for a counting processes approach.

This drug is still a possible choice for some patients. Another therapeutic choice is Bacillus Calmette-Guerin (BCG), a live bacterium related to cow tuberculosis.

0.51.1.1 Data dictionary

Table 35: Variables in the `bladder` dataset

Variable	Definition
`id`	Patient unique ID
`status`	for each time interval: 1 = recurred, 0 = censored
`interval`	1 = first recurrence, etc.
`intime`	‘`tstop - tstart` (all times in months)
`tstart`	start of interval
`tstop`	end of interval
`tx`	treatment code, 1 = thiotepa
`num`	number of initial tumors
`size`	size of initial tumors (cm)

- There are 85 patients and 190 lines in the dataset, meaning that many patients have more than one line.
- Patient 1 with 0 observation time was removed.
- Of the 85 patients, 47 had at least one recurrence and 38 had none.
- 18 patients had exactly one recurrence.
- There were up to 4 recurrences in a patient.
- Of the 190 intervals, 112 terminated with a recurrence and 78 were censored.

0.51.1.2 Different intervals for the same patient are correlated.

- Is the effective sample size 47 or 112? This might narrow confidence intervals by as much as a factor of $\sqrt{112/47} = 1.54$
- What happens if I have 5 treatment and 5 control values and want to do a t-test and I then duplicate the 10 values as if the sample size was 20? This falsely narrows confidence intervals by a factor of $\sqrt{2} = 1.41$.

```r
bladder =
  paste0(
    "http://web1.sph.emory.edu/dkleinb/allDatasets",
    "/surv2datasets/bladder.dta") |>
  read_dta() |>
  as_tibble()

bladder = bladder[-1,]  #remove subject with 0 observation time
print(bladder)
```

```
# A tibble: 190 x 8
      id event interval start  stop    tx   num  size
   <dbl> <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1     2     0        1     0     1     0     1     3
 2     3     0        1     0     4     0     2     1
 3     4     0        1     0     7     0     1     1
 4     5     0        1     0    10     0     5     1
 5     6     1        1     0     6     0     4     1
 6     6     0        2     6    10     0     4     1
 7     7     0        1     0    14     0     1     1
 8     8     0        1     0    18     0     1     1
 9     9     1        1     0     5     0     1     3
10     9     0        2     5    18     0     1     3
# i 180 more rows
```

```r
bladder =
  bladder |>
  mutate(
    surv =
      Surv(
        time=start,
        time2=stop,
        event=event,
        type="counting"))

bladder.cox1 = coxph(
  formula = surv~tx+num+size,
  data = bladder)

#results with biased variance-covariance matrix:
summary(bladder.cox1)
```

```
Call:
coxph(formula = surv ~ tx + num + size, data = bladder)

  n= 190, number of events= 112

        coef exp(coef) se(coef)     z Pr(>|z|)
tx   -0.4116    0.6626   0.1999 -2.06  0.03947 *
num   0.1637    1.1778   0.0478  3.43  0.00061 ***
size -0.0411    0.9598   0.0703 -0.58  0.55897
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     exp(coef) exp(-coef) lower .95 upper .95
tx       0.663      1.509     0.448      0.98
num      1.178      0.849     1.073      1.29
size     0.960      1.042     0.836      1.10

Concordance= 0.624  (se = 0.032 )
Likelihood ratio test= 14.7  on 3 df,   p=0.002
Wald test            = 15.9  on 3 df,   p=0.001
Score (logrank) test = 16.2  on 3 df,   p=0.001
```

> **Note**
>
> The likelihood ratio and score tests assume independence of observations within a cluster. The Wald and robust score tests do not.

0.51.1.3 adding `cluster = id`

If we add `cluster= id` to the call to `coxph`, the coefficient estimates don't change, but we get an additional column in the `summary()` output: `robust se`:

```
bladder.cox2 = coxph(
  formula = surv ~ tx + num + size,
  cluster = id,
  data = bladder)

#unbiased though this reduces power:
summary(bladder.cox2)
```

```
Call:
```

```
coxph(formula = surv ~ tx + num + size, data = bladder, cluster = id)

  n= 190, number of events= 112

        coef exp(coef) se(coef) robust se     z Pr(>|z|)
tx   -0.4116    0.6626   0.1999    0.2488 -1.65   0.0980 .
num   0.1637    1.1778   0.0478    0.0584  2.80   0.0051 **
size -0.0411    0.9598   0.0703    0.0742 -0.55   0.5799
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     exp(coef) exp(-coef) lower .95 upper .95
tx       0.663      1.509     0.407      1.08
num      1.178      0.849     1.050      1.32
size     0.960      1.042     0.830      1.11

Concordance= 0.624  (se = 0.031 )
Likelihood ratio test= 14.7  on 3 df,   p=0.002
Wald test            = 11.2  on 3 df,   p=0.01
Score (logrank) test = 16.2  on 3 df,   p=0.001,   Robust = 10.8  p=0.01

  (Note: the likelihood ratio and score tests assume independence of
     observations within a cluster, the Wald and robust score tests do not).
```

`robust se` is larger than `se`, and accounts for the repeated observations from the same individuals:

```
round(bladder.cox2$naive.var, 4)
```

```
        [,1]    [,2]   [,3]
[1,]  0.0400 -0.0014 0.0000
[2,] -0.0014  0.0023 0.0007
[3,]  0.0000  0.0007 0.0049
```

```
round(bladder.cox2$var, 4)
```

```
        [,1]    [,2]    [,3]
[1,]  0.0619 -0.0026 -0.0004
[2,] -0.0026  0.0034  0.0013
[3,] -0.0004  0.0013  0.0055
```

These are the ratios of correct confidence intervals to naive ones:

```
with(bladder.cox2, diag(var)/diag(naive.var)) |> sqrt()
```

```
[1] 1.244 1.223 1.056
```

We might try dropping the non-significant `size` variable:

```
#remove non-significant size variable:
bladder.cox3 = bladder.cox2 |> update(. ~ . - size)
summary(bladder.cox3)
```

```
Call:
coxph(formula = surv ~ tx + num, data = bladder, cluster = id)

  n= 190, number of events= 112

       coef exp(coef) se(coef) robust se     z Pr(>|z|)
tx  -0.4117    0.6625   0.2003    0.2515 -1.64   0.1017
num  0.1700    1.1853   0.0465    0.0564  3.02   0.0026 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

    exp(coef) exp(-coef) lower .95 upper .95
tx      0.663      1.509     0.405      1.08
num     1.185      0.844     1.061      1.32

Concordance= 0.623  (se = 0.031 )
Likelihood ratio test= 14.3  on 2 df,   p=8e-04
Wald test            = 10.2  on 2 df,   p=0.006
Score (logrank) test = 15.8  on 2 df,   p=4e-04,   Robust = 10.6  p=0.005

  (Note: the likelihood ratio and score tests assume independence of
     observations within a cluster, the Wald and robust score tests do not).
```

Ways to check PH assumption:

- cloglog
- schoenfeld residuals
- interaction with time

0.52 Parametric survival models

Configuring R

Functions from these packages will be used throughout this document:

```
library(pander) # format tables for markdown
library(ggplot2) # graphics
library(ggeasy) # help with graphics
library(dplyr) # manipulate data
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(conflicted) # check for conflicting function definitions
conflicts_prefer(dplyr::filter)
```

Here are some R settings I use in this document:

```
rm(list = ls()) # delete any data that's already loaded into R
knitr::opts_chunk$set(message = FALSE)
pander::panderOptions("table.emphasize.rownames", FALSE)
options('digits' = 4)
```

0.53 Parametric Survival Models

0.53.1 Exponential Distribution

- The exponential distribution is the basic distribution for survival analysis.

$$
\begin{aligned}
f(t) &= \lambda e^{-\lambda t} \\
\log\{f(t)\} &= \log\{\lambda\} - \lambda t \\
F(t) &= 1 - e^{-\lambda t} \\
S(t) &= e^{-\lambda t} \\
H(t) &= \log\{S(t)\} = -\lambda t \\
h(t) &= \lambda \\
\mathrm{E}(T) &= \lambda^{-1}
\end{aligned}
$$

0.53.2 Weibull Distribution

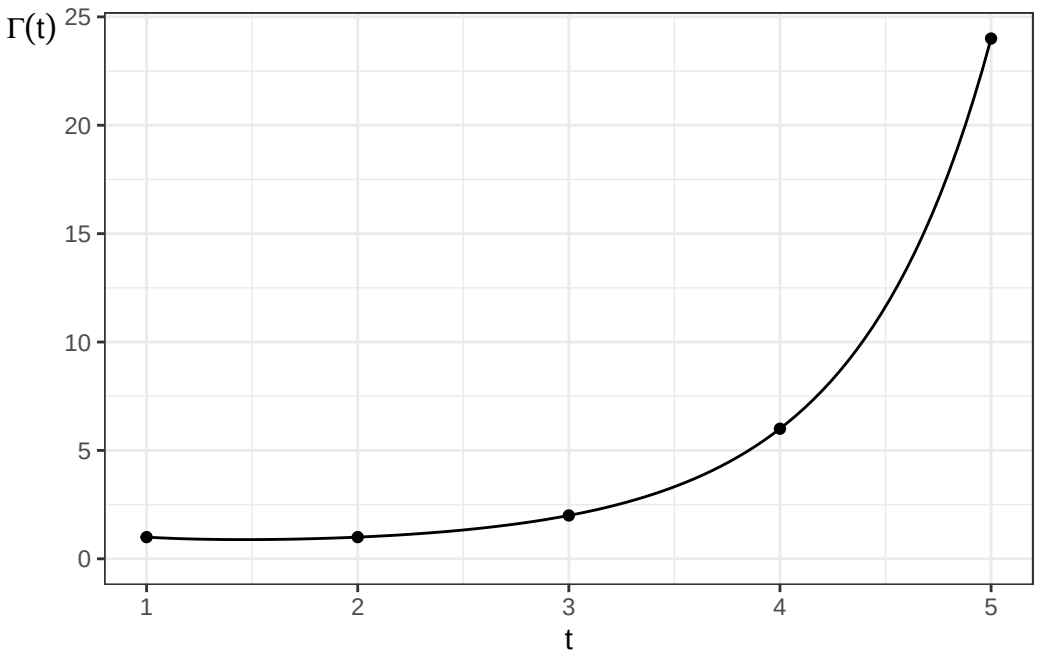
Using the Kalbfleisch and Prentice (2002) notation:

$$
\begin{aligned}
f(t) &= \lambda p(\lambda t)^{p-1} e^{-(\lambda t)^p} \\
F(t) &= 1 - e^{-(\lambda t)^p} \\
S(t) &= e^{-(\lambda t)^p} \\
h(t) &= \lambda p(\lambda t)^{p-1} \\
H(t) &= (\lambda t)^p \\
\log\{H(t)\} &= p\log\{\lambda t\} = p\log\{\lambda\} + p\log\{t\} \\
\mathrm{E}(T) &= \lambda^{-1} \cdot \Gamma\left(1 + \frac{1}{p}\right)
\end{aligned}
$$

Note

Recall from calculus:

- $\Gamma(t) \stackrel{\text{def}}{=} \int_{u=0}^{\infty} u^{t-1} e^{-u} du$
- $\Gamma(t) = (t-1)!$ for integers $t \in \mathbb{Z}$
- It is implemented by the `gamma()` function in R.



Here are some Weibull density functions, with $\lambda = 1$ and p varying:

```
library(ggplot2)
lambda = 1
ggplot() +
  geom_function(
    aes(col = "0.25"),
    fun = \(x) dweibull(x, shape = 0.25, scale = 1/lambda)) +
  geom_function(
    aes(col = "0.5"),
    fun = \(x) dweibull(x, shape = 0.5, scale = 1/lambda)) +
  geom_function(
    aes(col = "1"),
    fun = \(x) dweibull(x, shape = 1, scale = 1/lambda)) +
  geom_function(
    aes(col = "1.5"),
    fun = \(x) dweibull(x, shape = 1.5, scale = 1/lambda)) +
  geom_function(
    aes(col = "2"),
    fun = \(x) dweibull(x, shape = 2, scale = 1/lambda)) +
  geom_function(
    aes(col = "5"),
    fun = \(x) dweibull(x, shape = 5, scale = 1/lambda)) +
  theme_bw() +
  xlim(0, 2.5) +
  ylab("f(t)") +
  theme(axis.title.y = element_text(angle=0)) +
  theme(legend.position="bottom") +
  guides(
    col =
      guide_legend(
        title = "p",
        label.theme =
          element_text(
            size = 12)))
```

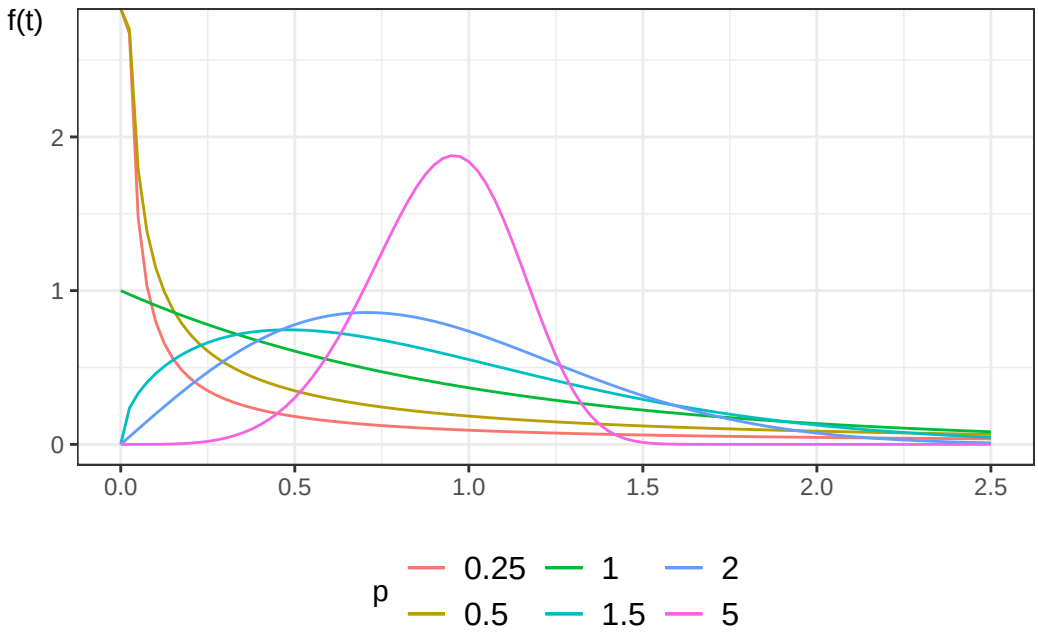


Figure 40: Density functions for Weibull distribution

0.53.2.1 Properties of Weibull hazard functions

- When $p = 1$, the Weibull distribution simplifies to the exponential distribution
- When $p > 1$, the hazard is increasing
- When $p < 1$, the hazard is decreasing

In HW: prove these properties

This distribution provides more flexibility than the exponential.

Here are some Weibull hazard functions, with $\lambda = 1$ and p varying:

```
library(ggplot2)
library(eha)
lambda = 1

ggplot() +
  geom_function(
    aes(col = "0.25"),
    fun = \(x) hweibull(x, shape = 0.25, scale = 1/lambda)) +
  geom_function(
```

```
  aes(col = "0.5"),
  fun = \(x) hweibull(x, shape = 0.5, scale = 1/lambda)) +
geom_function(
  aes(col = "1"),
  fun = \(x) hweibull(x, shape = 1, scale = 1/lambda)) +
geom_function(
  aes(col = "1.5"),
  fun = \(x) hweibull(x, shape = 1.5, scale = 1/lambda)) +
geom_function(
  aes(col = "2"),
  fun = \(x) hweibull(x, shape = 2, scale = 1/lambda)) +
theme_bw() +
xlim(0, 2.5) +
ylab("h(t)") +
theme(axis.title.y = element_text(angle=0)) +
theme(legend.position="bottom") +
guides(
  col =
    guide_legend(
      title = "p",
      label.theme =
        element_text(
          size = 12)))
```

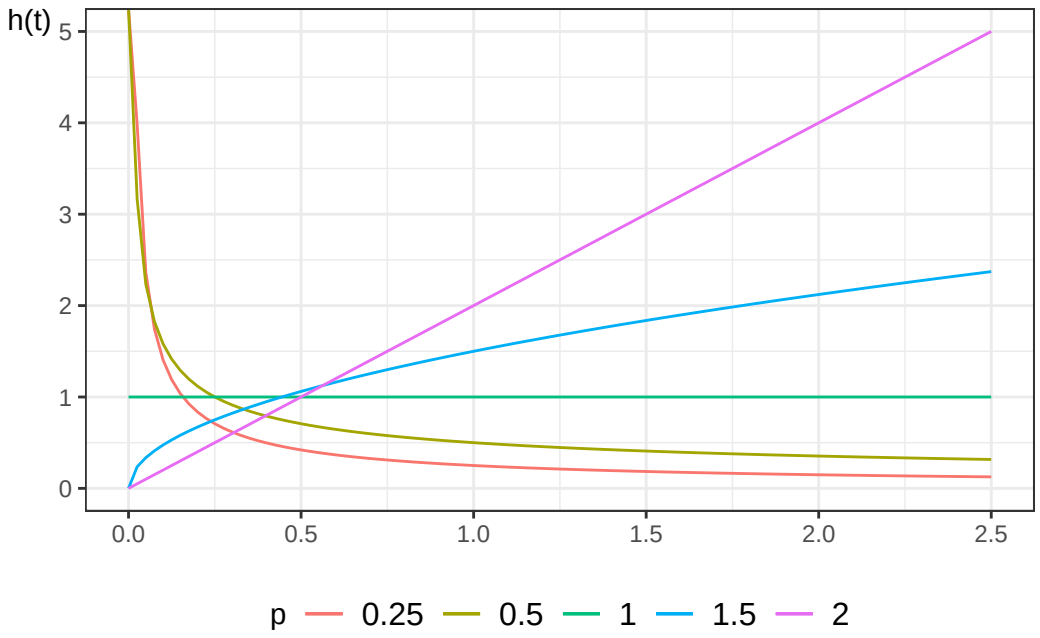


Figure 41: Hazard functions for Weibull distribution

```
library(ggplot2)
lambda = 1

ggplot() +
  geom_function(
    aes(col = "0.25"),
    fun = \(x) pweibull(lower = FALSE, x, shape = 0.25, scale = 1/lambda)) +
  geom_function(
    aes(col = "0.5"),
    fun = \(x) pweibull(lower = FALSE, x, shape = 0.5, scale = 1/lambda)) +
  geom_function(
    aes(col = "1"),
    fun = \(x) pweibull(lower = FALSE, x, shape = 1, scale = 1/lambda)) +
  geom_function(
    aes(col = "1.5"),
    fun = \(x) pweibull(lower = FALSE, x, shape = 1.5, scale = 1/lambda)) +
  geom_function(
    aes(col = "2"),
    fun = \(x) pweibull(lower = FALSE, x, shape = 2, scale = 1/lambda)) +
  theme_bw() +
```

```
xlim(0, 2.5) +
ylab("S(t)") +
theme(axis.title.y = element_text(angle=0)) +
theme(legend.position="bottom") +
guides(
  col =
    guide_legend(
      title = "p",
      label.theme =
        element_text(
          size = 12)))
```

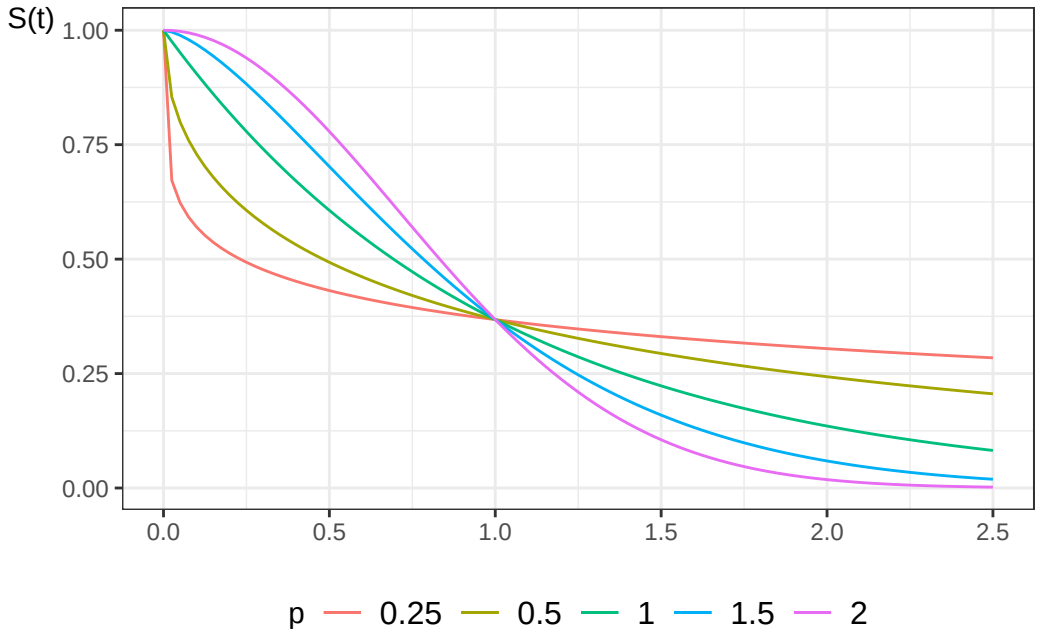


Figure 42: Survival functions for Weibull distribution

0.53.3 Exponential Regression

For each subject i, define a linear predictor:

$$
\begin{aligned}
\eta(x) &= \beta_0 + (\beta_1 x_1 + \cdots + \beta_p x_p) \\
h(t|x) &= \exp\{\eta(x)\} \\
h_0 &\stackrel{\text{def}}{=} h(t|0) \\
&= \exp\{\eta(0)\} \\
&= \exp\left\{\beta_0 + (\beta_1 \cdot 0 + \cdots + \beta_p \cdot 0)\right\} \\
&= \exp\{\beta_0 + 0\} \\
&= \exp\{\beta_0\}
\end{aligned}
$$

We let the linear predictor have a constant term, and when there are no additional predictors the hazard is $\lambda = \exp\{\beta_0\}$. This has a log link as in a generalized linear model. Since the hazard does not depend on t, the hazards are (trivially) proportional.

0.53.4 Accelerated Failure Time

Previously, we assumed the hazards were proportional; that is, the covariates multiplied the baseline hazard function:

$$
\begin{aligned}
h(T = t|X = x) &\stackrel{\text{def}}{=} p(T = t|X = x, T \geq t) \\
&= h(t|X = 0) \cdot \exp\{\eta(x)\} \\
&= h(t|X = 0) \cdot \theta(x) \\
&= h_0(t) \cdot \theta(x)
\end{aligned}
$$

and correspondingly,

$$
\begin{aligned}
H(t|x) &= \theta(x) H_0(t) \\
S(t|x) &= \exp\{-H(t|x)\} \\
&= \exp\{-\theta(x) \cdot H_0(t)\} \\
&= (\exp\{-H_0(t)\})^{\theta(x)} \\
&= (S_0(t))^{\theta(x)}
\end{aligned}
$$

An alternative modeling assumption would be

$$
S(t|X = x) = S_0(t \cdot \theta(x))
$$

where $\theta(x) = \exp\{\eta(x)\}$, $\eta(x) = \beta_1 x_1 + \cdots + \beta_p x_p$, and $S_0(t) = P(T \geq t|X = 0)$ is the base survival function.

Then

$$
\begin{aligned}
E(T|X=x) &= \int_{t=0}^{\infty} S(t|x)dt \\
&= \int_{t=0}^{\infty} S_0(t\cdot\theta(x))dt \\
&= \int_{u=0}^{\infty} S_0(u)du\cdot\theta(x)^{-1} \\
&= \theta(x)^{-1}\cdot\int_{u=0}^{\infty} S_0(u)du \\
&= \theta(x)^{-1}\cdot \mathrm{E}(T|X=0)
\end{aligned}
$$

So the mean of T given $X=x$ is the baseline mean divided by $\theta(x)=\exp\{\eta(x)\}$.

This modeling strategy is called an accelerated failure time model, because covariates cause uniform acceleration (or slowing) of failure times.

Additionally:

$$
\begin{aligned}
H(t|x) &= H_0(\theta(x)\cdot t) \\
h(t|x) &= \theta(x)\cdot h_0(\theta(x)\cdot t)
\end{aligned}
$$

If the base distribution is exponential with parameter λ then

$$
\begin{aligned}
S(t|x) &= \exp\{-\lambda\cdot t\theta(x)\} \\
&= [\exp\{-\lambda t\}]^{\theta(x)}
\end{aligned}
$$

which is an exponential model with base hazard multiplied by $\theta(x)$, which is also the proportional hazards model.

In terms of the log survival time $Y=\log\{T\}$ the model can be written as

$$
\begin{aligned}
Y &= \alpha-\eta+W \\
\alpha &= -\log\{\lambda\}
\end{aligned}
$$

where W has the extreme value distribution. The estimated parameter λ is the intercept and the other coefficients are those of η, which will be the opposite sign of those for coxph.

For a Weibull distribution, the hazard function and the survival function are

$$
\begin{aligned}
h(t) &= \lambda p(\lambda t)^{p-1} \\
S(t) &= e^{-(\lambda t)^p}
\end{aligned}
$$

We can construct a proportional hazards model by using a linear predictor η_i without constant term and letting $\theta_i = e^{\eta_i}$ we have

$$h(t) = \lambda p(\lambda t)^{p-1}\theta_i$$

A distribution with $h(t) = \lambda p(\lambda t)^{p-1}\theta_i$ is a Weibull distribution with parameters $\lambda^* = \lambda\theta_i^{1/p}$ and p so the survival function is

$$\begin{aligned} S^*(t) &= e^{-(\lambda^* t)^p} \\ &= e^{-(\lambda\theta^{1/p}t)^p} \\ &= S(t\theta^{1/p}) \end{aligned}$$

so this is also an accelerated failure time model.

In terms of the log survival time $Y = \log\{T\}$ the model can be written as

$$\begin{aligned} Y &= \alpha - \sigma\eta + \sigma W \\ \alpha &= -\log\{\lambda\} \\ \sigma &= 1/p \end{aligned}$$

where W has the extreme value distribution. The estimated parameter λ is the intercept and the other coefficients are those of η, which will be the opposite sign of those for `coxph`.

These AFT models are log-linear, meaning that the linear predictor has a log link. The exponential and the Weibull are the only log-linear models that are simultaneously proportional hazards models. Other parametric distributions can be used for survival regression either as a proportional hazards model or as an accelerated failure time model.

0.53.5 Dataset: Leukemia treatments

Remission survival times on 42 leukemia patients, half on new treatment, half on standard treatment.

This is the same data as the `drug6mp` data from KMsurv, but with two other variables and without the pairing.

```
library(haven)
library(survival)
anderson =
  paste0(
    "http://web1.sph.emory.edu/dkleinb/allDatasets",
```

```
    "/surv2datasets/anderson.dta") |>
  read_dta() |>
  mutate(
    status = status |>
      case_match(
        1 ~ "relapse",
        0 ~ "censored"
      ),
    sex = sex |>
      case_match(
        0 ~ "female",
        1 ~ "male"
      ),

    rx = rx |>
      case_match(
        0 ~ "new",
        1 ~ "standard"
      ),

    surv = Surv(time = survt,event = (status == "relapse"))
  )

print(anderson)
```

```
# A tibble: 42 x 7
   survt status   sex    logwbc rx    lwbc3   surv
   <dbl> <chr>    <chr>   <dbl> <chr> <dbl> <Surv>
 1    35 censored male     1.45 new       1    35+
 2    34 censored male     1.47 new       1    34+
 3    32 censored male     2.2  new       1    32+
 4    32 censored male     2.53 new       2    32+
 5    25 censored male     1.78 new       1    25+
 6    23 relapse  male     2.57 new       2    23
 7    22 relapse  male     2.32 new       2    22
 8    20 censored male     2.01 new       1    20+
 9    19 censored female   2.05 new       1    19+
10    17 censored female   2.16 new       1    17+
# i 32 more rows
```

0.53.5.1 Cox semi-parametric model

```
anderson.cox0 = coxph(
  formula = surv ~ rx,
  data = anderson)
summary(anderson.cox0)
```

```
Call:
coxph(formula = surv ~ rx, data = anderson)

  n= 42, number of events= 30

            coef exp(coef) se(coef)    z Pr(>|z|)
rxstandard 1.572     4.817    0.412 3.81  0.00014 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

           exp(coef) exp(-coef) lower .95 upper .95
rxstandard      4.82      0.208      2.15      10.8

Concordance= 0.69  (se = 0.041 )
Likelihood ratio test= 16.4  on 1 df,   p=5e-05
Wald test            = 14.5  on 1 df,   p=1e-04
Score (logrank) test = 17.2  on 1 df,   p=3e-05
```

0.53.5.2 Weibull parametric model

```
anderson.weib <- survreg(
  formula = surv ~ rx,
  data = anderson,
  dist = "weibull")
summary(anderson.weib)
```

```
Call:
survreg(formula = surv ~ rx, data = anderson, dist = "weibull")
             Value Std. Error     z       p
(Intercept)  3.516      0.252 13.96 < 2e-16
rxstandard  -1.267      0.311 -4.08 4.5e-05
Log(scale)  -0.312      0.147 -2.12   0.034
```

```
Scale= 0.732

Weibull distribution
Loglik(model)= -106.6   Loglik(intercept only)= -116.4
    Chisq= 19.65 on 1 degrees of freedom, p= 9.3e-06
Number of Newton-Raphson Iterations: 5
n= 42
```

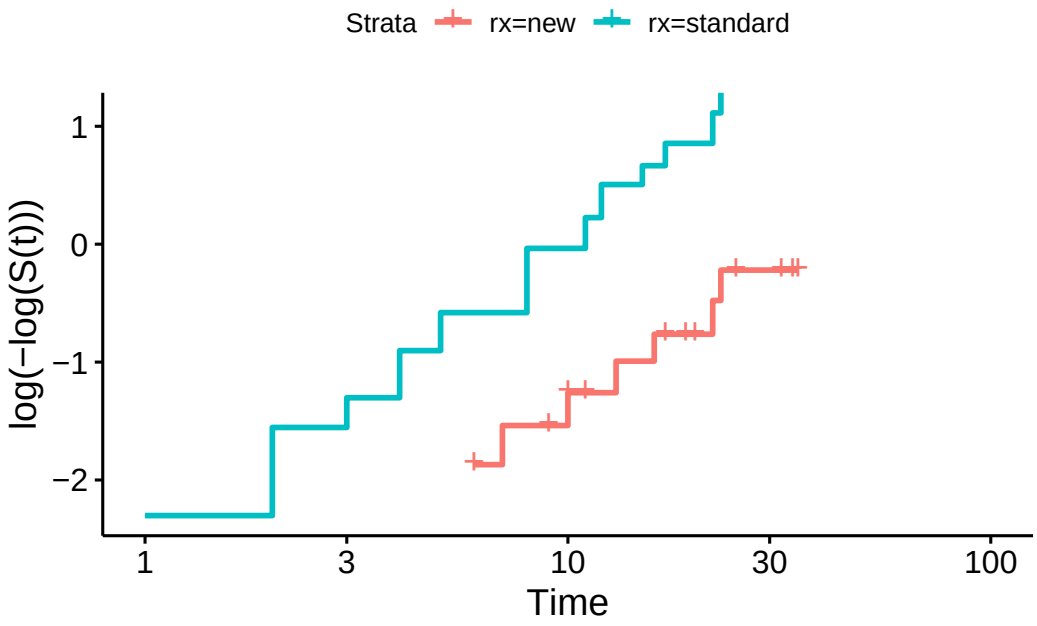
0.53.5.3 Exponential parametric model

```
anderson.exp <- survreg(
  formula = surv ~ rx,
  data = anderson,
  dist = "exp")
summary(anderson.exp)
```

```
Call:
survreg(formula = surv ~ rx, data = anderson, dist = "exp")
             Value Std. Error     z       p
(Intercept)  3.686      0.333 11.06 < 2e-16
rxstandard  -1.527      0.398 -3.83 0.00013

Scale fixed at 1

Exponential distribution
Loglik(model)= -108.5   Loglik(intercept only)= -116.8
    Chisq= 16.49 on 1 degrees of freedom, p= 4.9e-05
Number of Newton-Raphson Iterations: 4
n= 42
```

0.53.5.4 Diagnostic - complementary log-log survival plot

```
library(survminer)
survfit(
  formula = surv ~ rx,
  data = anderson) |>
  ggsurvplot(fun = "cloglog")
```



If the cloglog plot is linear, then a Weibull model may be ok.

References

Appendices

0.54 Introduction to Maximum Likelihood Inference

These notes are derived primarily from:

Dobson, Annette J., and Adrian G. Barnett. An introduction to generalized linear models. 3rd edition. CRC press, 2008 (mostly chapters 3-5).

I've added this book to the "Reading List" on canvas, so it should be uploaded there by the librarians soon.

Some material was also taken from:

McLachlan, Geoffrey J., and Thriyambakam Krishnan. The EM algorithm and extensions. John Wiley & Sons, 2007. https://onlinelibrary.wiley.com/doi/book/10.1002/9780470191613

Casella, G., & Berger, R. L. (2002). Statistical inference. 2nd ed. Australia ; Pacific Grove, CA, Thomson Learning.

0.55 Maximum likelihood inference for univariate Gaussian models

Suppose $X_1, \ldots, X_n \sim_{iid} N(\mu, \sigma^2)$. Let $X = (X_1, \ldots, X_n)^\top$ be these random variables in vector format. Let x_i and x denote the corresponding observed data. Let $\theta = (\mu, \sigma^2)^\top$ be the vector of parameters. Let Θ denote the parameters as a random vector.

Then the log-likelihood $\ell \stackrel{\text{def}}{=} \ell(X;\theta) \stackrel{\text{def}}{=} p(X = x \mid \Theta = \theta)$ is:

$$\begin{aligned}\ell &\propto -\frac{n}{2}\log\{\sigma^2\} - \frac{1}{2}\sum_{i=1}^{n}\frac{(x_i-\mu)^2}{\sigma^2} \\ &= -\frac{n}{2}\log\{\sigma^2\} - \frac{1}{2\sigma^2}\sum_{i=1}^{n} x_i^2 - 2x_i\mu + \mu^2\end{aligned}$$

0.55.1 MLE of μ:

Then:

$$\begin{aligned}\frac{d\ell}{d\mu} &= -\frac{1}{2}\sum_{i=1}^{n}\frac{-2(x_i-\mu)}{\sigma^2} \\ &= \frac{1}{\sigma^2}\left[\left(\sum_{i=1}^{n} x_i\right) - n\mu\right]\end{aligned}$$

If $\frac{d\ell}{d\mu} = 0$, then $\mu = \overline{x} \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} x_i$.

$$\frac{d^2\ell}{(d\mu)^2} = \frac{-n}{\sigma^2} < 0$$

So $\widehat{\mu}_{ML} = \overline{x}$.

0.55.2 MLE of σ^2

Reparametrizing the Gaussian distribution

When solving for $\hat{\sigma}_{ML}$, you can treat σ^2 as an atomic variable (don't differentiate with respect to σ or things get messy). In fact, you can replace σ^2 with $1/\tau$ and differentiate with respect to τ instead, and the process might be even easier.

$$\frac{d\ell}{d\sigma^2} = \frac{d}{d\sigma^2}\left(-\frac{n}{2}\log\left\{\sigma^2\right\} - \frac{1}{2}\sum_{i=1}^{n}\frac{\left(x_i-\mu\right)^2}{\sigma^2}\right)$$

$$= -\frac{n}{2}\left(\sigma^2\right)^{-1} + \frac{1}{2}\left(\sigma^2\right)^{-2}\sum_{i=1}^{n}\left(x_i-\mu\right)^2$$

If $\frac{d\ell}{d\sigma^2} = 0$, then:

$$\frac{n}{2}\left(\sigma^2\right)^{-1} = \frac{1}{2}\left(\sigma^2\right)^{-2}\sum_{i=1}^{n}\left(x_i-\mu\right)^2$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\left(x_i-\mu\right)^2$$

We plug in $\hat{\mu}_{ML} = \overline{x}$ to maximize globally (a technique called profiling):

$$\sigma^2_{ML} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i-\overline{x}\right)^2$$

Now:

$$\begin{aligned}
\frac{d^2\ell}{\left(d\sigma^2\right)^2} &= \frac{d}{d\sigma^2}\left\{-\frac{n}{2}\left(\sigma^2\right)^{-1} + \frac{1}{2}\left(\sigma^2\right)^{-2}\sum_{i=1}^{n}\left(x_i-\mu\right)^2\right\} \\
&= \left\{-\frac{n}{2}\frac{d}{d\sigma^2}\left(\sigma^2\right)^{-1} + \frac{1}{2}\frac{d}{d\sigma^2}\left(\sigma^2\right)^{-2}\sum_{i=1}^{n}\left(x_i-\mu\right)^2\right\} \\
&= \left\{\frac{n}{2}\left(\sigma^2\right)^{-2} - \left(\sigma^2\right)^{-3}\sum_{i=1}^{n}\left(x_i-\mu\right)^2\right\} \\
&= \left(\sigma^2\right)^{-2}\left\{\frac{n}{2} - \left(\sigma^2\right)^{-1}\sum_{i=1}^{n}\left(x_i-\mu\right)^2\right\}
\end{aligned}$$

Evaluated at $\mu = \overline{x}, \sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2$, we have:

$$\begin{aligned}
\frac{d^2\ell}{\left(d\sigma^2\right)^2} &= \left(\hat{\sigma}^2\right)^{-2}\left\{\frac{n}{2} - \left(\hat{\sigma}^2\right)^{-1}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2\right\} \\
&= \left(\hat{\sigma}^2\right)^{-2}\left\{\frac{n}{2} - \left(\hat{\sigma}^2\right)^{-1} n\hat{\sigma}^2\right\} \\
&= \left(\hat{\sigma}^2\right)^{-2}\left\{\frac{n}{2} - n\right\} \\
&= \left(\hat{\sigma}^2\right)^{-2} n\left\{\frac{1}{2} - 1\right\} \\
&= \left(\hat{\sigma}^2\right)^{-2} n\left(-\frac{1}{2}\right) < 0
\end{aligned}$$

Finally, we have:

$$\begin{aligned}
\frac{d^2\ell}{d\mu\ d\sigma^2} &= \frac{d}{d\mu}\left\{-\frac{n}{2}\left(\sigma^2\right)^{-1} + \frac{1}{2}\left(\sigma^2\right)^{-2}\sum_{i=1}^{n}\left(x_i - \mu\right)^2\right\} \\
&= \frac{1}{2}\left(\sigma^2\right)^{-2}\frac{d}{d\mu}\sum_{i=1}^{n}\left(x_i - \mu\right)^2 \\
&= \frac{1}{2}\left(\sigma^2\right)^{-2}\sum_{i=1}^{n} -2(x_i - \mu) \\
&= -\left(\sigma^2\right)^{-2}\sum_{i=1}^{n}\left(x_i - \mu\right)
\end{aligned}$$

Evaluated at $\mu = \hat{\mu} = \overline{x}, \sigma^2 = \hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2$, we have:

$$\frac{d^2\ell}{d\mu\ d\sigma^2} = -\left(\hat{\sigma}^2\right)^{-2}\left(n\overline{x} - n\overline{x}\right) = 0$$

0.55.3 Covariance matrix of MLEs

0.55.3.1 The score function

Let θ be the vector of all parameters; here, $\theta = \left(\mu, \sigma^2\right)^\top$.

Let $\ell'(x,\theta) \stackrel{\text{def}}{=} \frac{d}{d\theta}\ell(x,\theta) = \begin{pmatrix} \frac{d}{d\mu}\ell(\theta;x) \\ \frac{d}{d\sigma^2}\ell(\theta;x) \end{pmatrix} = \begin{pmatrix} \ell'_{\mu}(\theta;x) \\ \ell'_{\sigma^2}(\theta;x) \end{pmatrix}$.

$\ell'(x,\theta)$ is the function we set equal to 0 and solve to find the MLE:

$$\hat{\theta}_{ML} = \left\{\theta : \ell^{'}(x,\theta) = 0\right\}$$

The function $\ell^{'}(x,\theta)$ is so central that it has its own name, the "score" or "gradient" function. Statisticians also often skip writing the arguments (x,θ), so $\ell^{'} \stackrel{\text{def}}{=} \ell^{'}(x,\theta)$.[2] Some statisticians use U or S instead of $\ell^{'}$. I prefer $\ell^{'}$. Why use up extra letters?

0.55.3.2 The (Fisher) (expected) information matrix

The variance of $\ell^{'}(x,\theta)$, $Cov\left\{\ell^{'}(x,\theta)\right\}$, is also very important; we call it the "expected information matrix", "Fisher information matrix", or just "information matrix", and we represent it using the symbol $\mathfrak{I}$ (`\frakturI` in Unicode, `\mathfrak{I}` in LaTeX).

Review of variances and covariances

Variances and covariances of one-dimensional random variables

For a one-dimensional random variables X,

$$Var(X) \stackrel{\text{def}}{=} \mathrm{E}\left[(X - \mathrm{E}[X])^2\right] = \mathrm{E}\left[X^2\right] - (\mathrm{E}[X])^2$$

For any two-dimensional random variables, X, Y:

$$Cov(X,Y) = E\left[(X - \mathrm{E}X)(Y - \mathrm{E}Y)\right] = \mathrm{E}[XY] - E[X]E[Y]$$

Therefore, $Var(X) = Cov(X,X) = \mathrm{E}[XX] - E[X]E[X] = \mathrm{E}\left[X^2\right] - (E[X])^2$

$$\mathfrak{I} \stackrel{\text{def}}{=} \mathfrak{I}(\theta) \stackrel{\text{def}}{=} Cov\left(\ell^{'}|\theta\right) = \mathrm{E}\left[\ell^{'}\ell^{'\top}\right] - \mathrm{E}\left[\ell^{'}\right]\ \mathrm{E}\left[\ell^{'}\right]^{\top}$$

Sometimes we write $Cov(X) \stackrel{\text{def}}{=} Cov(X,X) = Var(X)$.

[2] I might sometimes switch the order of x, θ; this is unintentional and not meaningful.

Variances and covariances of $p \times 1$ random vectors

Now, for a $p \times 1$ dimensional random vector X,

$$\begin{aligned}
\text{Var}(X) &\stackrel{\text{def}}{=} \text{Cov}(X) \\
&\stackrel{\text{def}}{=} E\left[(X - E[X])^\top (X - E[X])\right] \\
&= E\left[(X^\top - E[X]^\top)(X - E[X])\right] \\
&= E\left[X^\top X - E[X]^\top X - X^\top E[X] + E[X]^\top E[X]\right] \\
&= E\left[X^\top X\right] - E[X]^\top E[X] - E[X]^\top E[X] + E[X]^\top E[X] \\
&= E\left[X^\top X\right] - 2E[X]^\top E[X] + E[X]^\top E[X] \\
&= E\left[X^\top X\right] - E[X]^\top E[X]
\end{aligned}$$

The elements of $\mathfrak{I}$ are:

$$\left\{\mathfrak{I}_{ij} \stackrel{\text{def}}{=} Cov\left(\ell'_i, \ell'_j\right) = \text{E}\left[\ell'_i \ell'_j\right] - \text{E}\left[\ell'_i\right] \text{E}\left[\ell'_j\right]\right\}$$

In our motivating example, $i, j \in \{\mu, \sigma^2\}$. Here,

$$\begin{aligned}
\text{E}[\ell'] &\stackrel{\text{def}}{=} \int_{x \in \mathcal{R}(x)} \ell'(x, \theta) p(X = x|\theta) dx \\
&= \int_{x \in \mathcal{R}(x)} \left(\frac{d}{d\theta} \log\left\{p\left(X = x \mid \theta\right)\right\}\right) \; p\left(X = x \mid \theta\right) \; dx \\
&= \int_{x \in \mathcal{R}(x)} \frac{\frac{d}{d\theta} p\left(X = x \mid \theta\right)}{p\left(X = x \mid \theta\right)} p\left(X = x \mid \theta\right) \; dx \\
&= \int_{x \in \mathcal{R}(x)} \frac{d}{d\theta} p\left(X = x \mid \theta\right) \; dx
\end{aligned}$$

And similarly

$$\text{E}\left[\ell' \ell'^\top\right] \stackrel{\text{def}}{=} \int_{x \in R(x)} \ell'(x, \theta) \ell'(x, \theta)^\top \; p\left(X = x \mid \theta\right) \; dx$$

Note that $\text{E}\left[\ell'\right]$ and $\text{E}\left[\ell' \ell'^\top\right]$ are functions of θ but not of x; the expectation operator removed x.

Also note that for most of the distributions you are familiar with (including Gaussian, binomial, Poisson, exponential),

$$\mathrm{E}\left[\ell'\right] = 0$$

So

$$\mathfrak{I} = \mathrm{E}\left[\ell' {\ell'}^{\top}\right]$$

Moreover, for those distributions (called the "exponential family"), we have:

$$\mathfrak{I} = -\mathrm{E}\left[\ell''\right] = \mathrm{E}\left[-\ell''\right]$$

(see Dobson and Barnett 4e, §3.17), where

$$\ell'' \stackrel{\text{def}}{=} \frac{d}{d\theta}\ell'^{(x,\theta)^{\top}} = \frac{d}{d\theta}\frac{d}{d\theta^{\top}}\ell(x,\theta)$$

is the $p \times p$ matrix whose elements are:

$$\ell''_{ij} \stackrel{\text{def}}{=} \frac{d}{d\theta_i}\frac{d}{d\theta_j}\log\left\{p\left(X = x \mid \theta\right)\right\}$$

ℓ'' could be called the "Hessian" of the log-likelihood function.

Sometimes, we use $I(\theta; x) \stackrel{\text{def}}{=} -\ell''$ (note the standard-font "I" here). $I(\theta; x)$ is the observed information matrix (Negative Hessian).

> **!** Key point
>
> The asymptotics of MLEs gives us $\hat{\theta}_{ML} \sim N\left(\theta, \mathfrak{I}^{-1}(\theta)\right)$, approximately, for large sample sizes.

We can estimate $\mathfrak{I}^{-1}(\theta)$ by working out $-\mathrm{E}\left[\ell''\right]$ or $\mathrm{E}\left[\ell' {\ell'}^{\top}\right]$ and plugging in $\hat{\theta}_{ML}$, but sometimes we instead use $I\left(\hat{\theta}_{ML}; x\right)$ for convenience; there are some cases where it's provably better according to some criteria (Efron & Hinkley 1978).

Note that later, when we are trying to find MLEs for likelihoods that we can't easily differentiate, we will "hill-climb" using the Newton-Raphson algorithm:

$$\begin{aligned}\hat{\theta} &\leftarrow \hat{\theta} + \left[I\left(\hat{\theta}, y\right)\right]^{-1} \ell'\left(y, \hat{\theta}\right) \\ &= \hat{\theta} - \left[\ell''\left(y, \hat{\theta}\right)\right]^{-1} \ell'\left(y, \hat{\theta}\right)\end{aligned}$$

Here, for computational simplicity, we will sometimes use $\mathfrak{I}^{-1}(\theta)$ in place of $I\left(\hat{\theta}, y\right)$; doing so is called "Fisher scoring" or the "method of scoring". Note that this is the opposite of the substitution that we are making for estimating the variance of the MLE; this time we should technically use the observed information but we use the expected information instead.

There's also an "empirical information matrix" (see McLachlan and Krishnan 2007).

$$I_e(\theta, y) = \sum_{i=1}^{n} \ell_i^{'} \ \ell_i^{'\top} - \frac{1}{n} \ell^{'} \ell^{'\top}$$

where ℓ_i is the log-likelihood of the ith observation. Note that $\ell^{'} = \sum_{i=1}^{n} \ell_i^{'}$.

$\frac{1}{n} I_e(\theta, y)$ is the sample equivalent of

$$\mathfrak{I} \stackrel{\text{def}}{=} \mathfrak{I}(\theta) \stackrel{\text{def}}{=} Cov\left(\ell^{'} | \theta\right) = \mathrm{E}\left[\ell^{'} \ell^{'\top}\right] - \mathrm{E}\left[\ell^{'}\right] \ \mathrm{E}\left[\ell^{'}\right]^{\top}$$

$$\left\{ \mathfrak{I}_{jk} \stackrel{\text{def}}{=} Cov\left(\ell^{'}{}_j, \ell^{'}{}_k\right) = \mathrm{E}\left[\ell_j^{'} \ell_k^{'}\right] - \mathrm{E}\left[\ell^{'}{}_j\right] \mathrm{E}\left[\ell^{'}{}_k\right] \right\}$$

$I_e(\theta, y)$ is sometimes computationally easier to compute for Newton-Raphson-type maximization algorithms.

Back to our Gaussian example:

$$I = \begin{bmatrix} \frac{n}{\sigma^2} & 0 \\ 0 & \left(\widehat{\sigma}^2\right)^{-2} n \left(-\frac{1}{2}\right) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix}$$

So:

$$I^{-1} = \frac{1}{ad} \begin{bmatrix} d & 0 \\ 0 & a \end{bmatrix} = \begin{bmatrix} \frac{1}{a} & 0 \\ 0 & \frac{1}{d} \end{bmatrix}$$

$$I^{-1} = \begin{bmatrix} \frac{\widehat{\sigma}^2}{n} & 0 \\ 0 & \frac{2\left(\widehat{\sigma}^2\right)^2}{n} \end{bmatrix}$$

See Casella-Berger p322, example 7.2.12.

To prove it's a maximum, need:

- $\ell^{'} = 0$
- At least one diagonal element of l'' is negative.
- Determinant of l'' is positive.

0.55.4 Confidence intervals for MLEs

0.55.5 p-values and hypothesis tests for MLEs

0.55.6 Likelihood ratio tests for MLEs

[We haven't gone over this yet]

Log-likelihood ratio tests (Dobson & Barnett 5.7)

$$2\left(l - \ell_0\right) \sim \chi^2(p - q)$$

0.55.7 Prediction intervals for MLEs

$$\overline{X} \in \left[\hat{\mu} \pm z_{1-\alpha/2}\frac{\sigma}{m}\right]$$

Where m is the sample size of the new data to be predicted (typically 1, except for binary outcomes, where it needs to be bigger for prediction intervals to make sense)