Practical Mixed Models for Actuaries

Ernesto Schirmacher

2024-05-13

Table of contents

Preface

Note to Reviewers

All computations were done in R and *most* of the data sets used are available in R packages. I have made all code chunks visible, but in the final version some of them are likely to be hidden (unless there is a strong preference to having them visible). Currently the code chunks do not all appear in their ideal position in the PDF document.

You will need to have the following libraries installed:

1. actuar
2. CASdatasets
3. dhglm
4. gamlss
5. GGally
6. GLMsData
7. hglm
8. insuranceData
9. kableExtra
10. knitr
11. lars
12. lme4
13. MASS
14. mdhglm
15. mvtnorm
16. patchwork
17. statmod
18. tidyverse

The data sets *not* available in packages are:

1. medpar.csv
2. bus-case.csv
3. hachemeister-data.csv
4. BS-simulated-data.csv

There is one file with R functions that needs to be sourced into the credibility chapter:

1. buehlmann-gisler-calculations.R

I have also extracted all code chunks in a chapter and placed them into individual R Notebooks:

1. `chapter-2-code.Rmd`
2. `chapter-3-code.Rmd`
3. `chapter-4-code.Rmd`
4. `chapter-5-code.Rmd`
5. `chapter-6-code.Rmd`
6. `appendix-a-code.Rmd`

The Data sets not available in R packages, the source file, and all the code chunks (each chapter separately) are included in the ZIP file with this PDF.

1 Introduction

Drafting an introduction.

2 Review of Generalized Linear Models

```
library(tidyverse)
library(patchwork)
library(kableExtra)
library(GLMsData)
library(statmod)
```

2.1 Introduction

Actuaries are well acquainted with generalized linear models and in this chapter we provide a quick review of the main ideas and concepts as we work through an example.

In the late 1970's and early 80's, researchers (Ira B. Tager et al. 1979; I. B. Tager et al. 1983) in Boston were interested in understanding how smoking affects the pulmonary function of children. A cross-sectional data set from this investigation is available in the `GLMsData` package under the name `lungcap`.

```
data(lungcap, package = "GLMsData")
```

The data set has 654 observations and 5 variables.

The pulmonary function of the subjects was assessed through their lung capacity and this capacity was measured via their forced expiratory volume. The forced expiratory volume is the amount of air a subject can expel from their lungs in the first second of a forceful exhalation. Larger volumes of exhaled air signals better pulmonary function.

The following table shows the variable name, type, and description for each variable in the data set.

Item	Variable	Type	Description
1	`FEV`	Continuous	The forced expiratory volume in liters
2	`Age`	Integer	Age of subject in completed years
3	`Ht`	Continuous	Height of the subject in inches
4	`Gender`	Binary	The gender of the subject

Item	Variable	Type	Description
5	`Smoke`	Binary	The smoking status of the subject. Non-smokers are coded with 0 and smokers are coded with 1

Forced expiratory volume, `FEV`, is our response variable and the indicator variable for smoking, `Smoke`, is the principal variable of interest. The age (`Age`), gender (`Gender`), and height (`Ht`) of each child are variables that may be related to the response and we want to control for them.

2.2 Exploratory Data Analysis

To work effectively with data we first need to understand the data we have to work with. Exploratory data analysis is a set of techniques to help us understand and uncover what the data we have may be saying. It is not about confirming that a perceived pattern is true. For that there are other techniques. It is about looking closely at the data to find out what we can do with it. It is about learning and finding insights from the data and being able to describe them as easily as possible.

In this section, we explore the lung capacity data. The response variable is forced expiratory volume, `FEV` and the remaining variables may help us explain it.

The `Age` variable ranges from 3 years to 19 years old and the height variable is between 46 and 74 inches. Thus we have a big spectrum of body sizes and so we should expect `FEV` to vary significantly as age and height varies. Table 2.2 shows summary statistics for the numeric variables. Note that the difference from the median (Q2) down to the first quartile (Q1) and up to the third quartile (Q3) are similar for each variable. This shows us that the bulk of the data, in each case, is fairly symmetric.

```
stats <- rbind(summary(lungcap$FEV),
               summary(lungcap$Age),
               summary(lungcap$Ht))
dimnames(stats) <- list(c("FEV (in liters)", "Age (in years)", "Height (in inches)"),
                        c("Min", "Q1", "Q2", "Mean", "Q3", "Max"))
kbl(round(stats, 2),
    format = "pipe",
    booktabs = TRUE)
rm(stats)
```

Table 2.2: Summary statistics for the numeric variables in the lung capacity data set. Q1 is the first quartile, Q2 is the median, and Q3 is the third quartile.

	Min	Q1	Q2	Mean	Q3	Max
FEV (in liters)	0.79	1.98	2.55	2.64	3.12	5.79
Age (in years)	3.00	8.00	10.00	9.93	12.00	19.00
Height (in inches)	46.00	57.00	61.50	61.14	65.50	74.00

During these ages children grow significantly and `Age` and `Ht` should be strongly related to each other; in fact, their linear correlation coefficient is equal to 0.79. Thus, including both of these variables in a linear model may pose some estimation problems (multicollinearity).

The remaining predictor variables are smoking status (`Smoke`) and gender (`Gender`). Both are binary variables. `Smoke` is an indicator variable where a value of 1 tells us that the child smokes and a value of zero says they do not smoke. There are 65 children who smoke in our data set (about 10%). For the variable `Gender`, the split between female and male is 49% and 51%, respectively.

```
p <- ggplot(data = lungcap,
            mapping = aes(x = Age,
                          y = FEV))
p <- p + geom_jitter(data = lungcap[lungcap$Smoke == 0,],
                     width = 0.2, height = 0,
                     pch = 3)
p <- p + geom_jitter(data = lungcap[lungcap$Smoke == 1,],
                     width = 0.2, height = 0,
                     color = "red", alpha = 0.6)
p <- p + geom_smooth(data = lungcap,
                     se = FALSE)
p <- p + labs(x = "Age [jittered] (in years)",
              y = "FEV (in liters)")

q <- ggplot(data = lungcap,
            mapping = aes(x = Ht,
                          y = FEV))
q <- q + geom_point(data = lungcap[lungcap$Smoke == 0,],
                    pch = 3)
q <- q + geom_point(data = lungcap[lungcap$Smoke == 1,],
                    color = "red", alpha = 0.6)
q <- q + geom_smooth(se = FALSE)
q <- q + labs(x = "Height (in inches)",
```

```
              y = "FEV (in liters)")
p + q
```

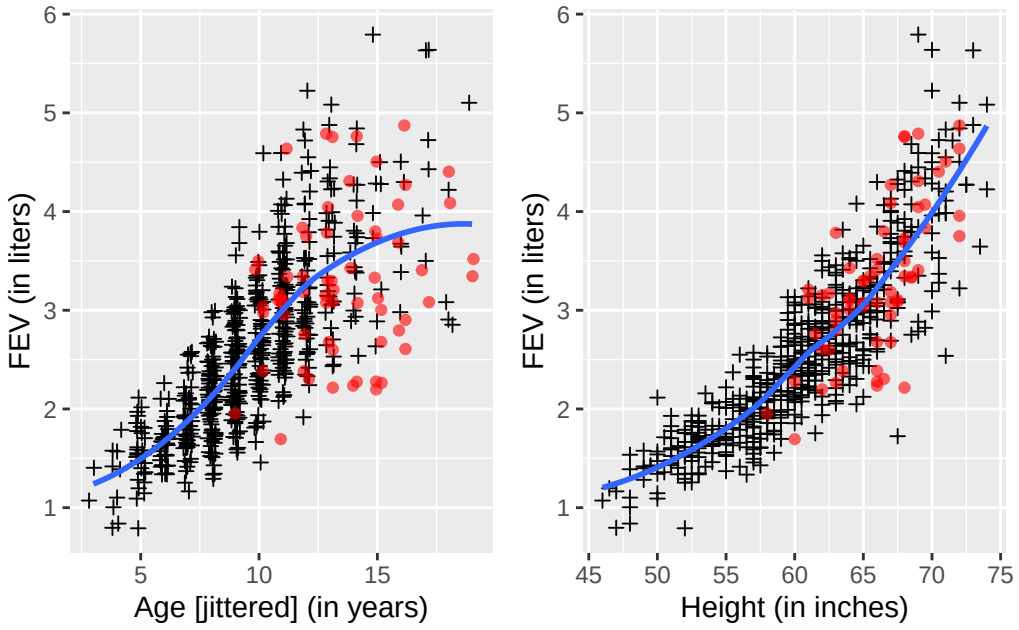


Figure 2.1: Age and height versus forced expiratory volume. The red circles denote smoking subjects and the plus signs represent non-smoking subjects. The smooth trend curves, which ignore smoking status, suggest non-linear relationships with the response variable.

In Figure 2.1 we see that both `Age` (left-hand panel) and `Ht` (right-hand panel) have a strong non-linear relationship with the response variable `FEV`. The non-linear smooth curves ignore the information about which subjects smoke and which do not.

The left-hand panel shows that the relationship between `Age` and `FEV` resembles an elongated `S` curve. Also we can see that as age increases, the cloud of points shows more dispersion as we move from the lower-left corner to the upper-right corner.

Switching to the right-hand panel we see that the relationship between height (`Ht`) and forced expiratory volume is also non-linear; but this non-linear pattern is simpler. In this case it resembles a cubic or exponential curve where increases in height lead to larger volumes. The cloud of points in this case is also more compact than the one on the left-hand panel. These observations would lead us to favor a model using height over one that uses age.

Also note that as the mean value of `FEV` increases in both scatter plots, the variability in `FEV` also increases. In other words, both plots show a *fanning out* of `FEV` as `FEV` increases. This relationship between the mean of the response and its variance, known as the **mean-variance**

relationship, is extremely important in generalized linear models, as it determines the member of the exponential family of distributions that we should use for our response variable.

The relationship between the mean and the variance of the response variable for many members of the exponential family follows the following equation

$$\text{Var}[y] = \phi\mu^b, \tag{2.1}$$

where ϕ is the dispersion parameter, μ is the mean of the distribution, and b is a non-negative number. Well known distributions correspond to different values of the exponent b.

For example, if $b = 0$, then we have the normal or Gaussian distribution. If $b = 1$, then the response variable is Poisson distributed and if $b = 2$, then it is gamma distributed. Other values of b are possible, and not all members of the exponential family have a mean-variance relationship given by Equation 2.1 (e.g., the binomial and negative binomial distributions).

Note that by applying a logarithm to both sides of Equation 2.1 we get the following equation.

$$\ln\left(\text{Var}[y]\right) = \ln(\phi) + b\ln(\mu) \tag{2.2}$$

Therefore, we can use our data and ordinary least squares to estimate the value of b.

For example, we can proceed as follows. First, create seven bins of approximately equal size for the height variable.

```
n <- 7
bks <- c(min(lungcap$Ht) - 0.1,
         quantile(lungcap$Ht, probs = (1:(n-1))/n),
         max(lungcap$Ht) + 0.1)
lungcap <- lungcap %>%
  mutate(Ht.bin = cut(Ht,
                      breaks = bks))
```

Using the height bins, summarize the value of `FEV` for each bin, by calculating the size of the bin, the mean, and the variance.

```
mv <- lungcap %>%
  group_by(Ht.bin) %>%
  summarize(sz = n(),
            mn = mean(FEV),
            vr = var(FEV))
```

Now estimate a linear regression equation where the response variable is the logarithm of the variance and predictor variable is the logarithm of the mean.

```
fm <- lm(log(vr) ~ log(mn),
         data = mv,
         weights = sz)
sfm <- summary(fm)
round(sfm$coef[,1:2], 3)
```

```
            Estimate Std. Error
(Intercept)   -3.740      0.253
log(mn)        2.015      0.260
```

The coefficient of the logarithm of the mean is our estimate of the value of b. In this case, $b = 2.02$ and since it is close to 2 in value, this suggests that we should model the forced expiratory volume, `FEV`, as a gamma distributed random variable. An approximate 95% confidence interval for the value of b is equal to $2.015 \pm 2 \times 0.260 = (1.495, 2.535)$. Clearly, this confidence interval does not include zero and so using a normal distribution for the response variable would not be a reasonable choice; that is, this choice is not supported by the data.

But what about other choices? Perhaps a Poisson distribution ($b = 1$) or an inverse gaussian distribution ($b = 2$) would be an appropriate choice. Well, the endpoints of the confidence interval are closer to these distributions and so one could try them out.

Exercise

Redo the mean-variance analysis with a different number of bins. Try various choices; maybe, $n = 3, 5, 10, 20$. Would you arrive at a similar conclusion?

Solution

```
n <- 20
bks <- c(min(lungcap$Ht) - 0.1,
         quantile(lungcap$Ht, probs = (1:(n-1))/n),
         max(lungcap$Ht) + 0.1)
lungcap <- lungcap %>%
  mutate(Ht.bin = cut(Ht,
                      breaks = bks))
```

```
mv <- lungcap %>%
  group_by(Ht.bin) %>%
  summarize(sz = n(),
            mn = mean(FEV),
            vr = var(FEV))
```

```
fm <- lm(log(vr) ~ log(mn),
         data = mv,
         weights = sz)
round(summary(fm)$coef[,1:2], 3)
```

```
            Estimate Std. Error
(Intercept)   -3.955      0.226
log(mn)        2.110      0.232
```

```
rm(fm, mv, bks, n)
```

We could have also anticipated that our response variable `FEV` is not normally distributed by carefully inspecting the scatter plots shown in Figure 2.1. Looking at `FEV` versus `Age` (left-hand panel) we see that as we move from the lower left-hand corner, where the response variable is small to the upper right-hand corner, the variability in `FEV` values increases. A similar phenomenon appears on the right-hand panel in the scatterplot of `FEV` versus height (`Ht`). Figure 2.2 shows the increase in variability as the mean of `FEV` increases.

If our response variable `FEV` was normally distributed, then all the arrows in Figure 2.2 would had the same length regardless of their position along the horizontal axis. In other words, we would have seen *constant variance* for the response variable.

```
df <- tibble(x = c(5, 10, 15),
             y = c(0.8, 1.5, 2.2),
             xend = x,
             yend = c(2.1, 4.6, 5.8))
p <- ggplot(data = lungcap,
            mapping = aes(x = Age,
                          y = FEV)) +
  geom_jitter(data = lungcap[lungcap$Smoke == 0,],
              width = 0.2, height = 0,
              pch = 3,
              color = "gray") +
  geom_jitter(data = lungcap[lungcap$Smoke == 1,],
              width = 0.2, height = 0,
              color = "pink") +
  geom_segment(data = df,
               mapping = aes(x = x, y = y,
                             xend = xend, yend = yend),
               color = "red",
               arrow = arrow(length = unit(2, "mm"),
                             ends = "both")) +
```

```
  labs(x = "Jittered Age (in years)",
       y = "FEV (in liters)")

df <- tibble(x = c(50, 60, 70),
             y = c(1.1, 1.6, 2.5),
             xend = x,
             yend = c(2.1, 3.2, 5.8))
q <- ggplot(data = lungcap,
            mapping = aes(x = Ht,
                          y = FEV)) +
  geom_point(data = lungcap[lungcap$Smoke == 0,],
             pch = 3,
             color = "gray") +
  geom_point(data = lungcap[lungcap$Smoke == 1,],
             color = "pink") +
  geom_segment(data = df,
               mapping = aes(x = x, y = y,
                             xend = xend, yend = yend),
               color = "red",
               arrow = arrow(length = unit(2, "mm"),
                             ends = "both")) +
  labs(x = "Height (in inches)",
       y = "FEV (in liters)")
p + q
rm(df, p, q)
```

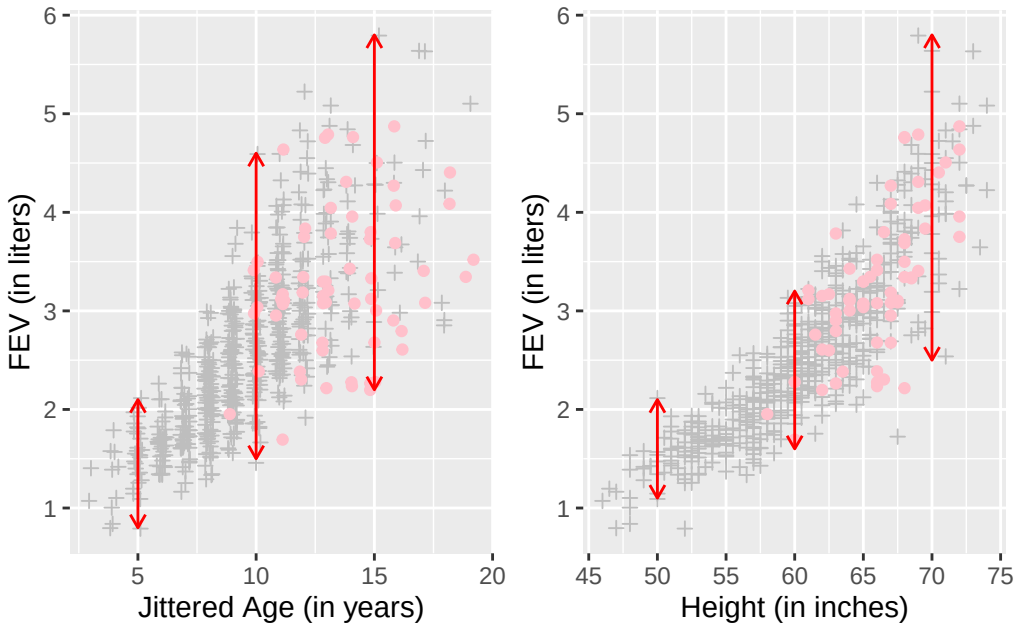


Figure 2.2: Age and height versus forced expiratory volume (`FEV`). The data has been rendered in muted gray and pink. The arrows on both panels depict the variability of `FEV` for small, medium, and large values of `FEV`. The increase in variability is evident.

Exercise

Looking at Figure 2.1 we can see that the left-hand panel shows a cloud of points centered around the blue trend line that are not as compactly arranged as the points shown on the right-hand panel.
The left-hand panel shows `FEV` vs. `Age` and the right-hand panel is `FEV` vs. `Ht`.
Why do you think we see this phenomenon?

Solution

The response variable is forced expiratory volume and so it measures size of the lungs. The relationship between the size of the lungs and height is much better defined than the size of lungs and age.
We all know children of the same age but very different heights. Some are shorter and others are taller. The taller ones have more space for larger lungs.
On the other hand, most children of the same height have similar builds and thus the variability in their lung size is smaller.

We have not yet explored how gender and smoking status may be related to the response variable. We can summarize our data by gender and smoker status and compute mean age, height, and forced expiratory volume. Table 2.3 shows the results. Note that lung capacity

(FEV) for both genders is higher for smokers than for non-smokers. Based on this alone, one might conclude that smoking would lead to higher lung capacity. But this would be an erroneous conclusion. The difference arises because the smoker and non-smoker subjects have different age and height characteristics. We can see that gender does play a role in lung capacity. For both gender groups, female participants have a smaller height and a smaller lung capacity.

```
FS <- (lungcap$Gender == "F") & (lungcap$Smoke == 1)
FN <- (lungcap$Gender == "F") & (lungcap$Smoke == 0)
MS <- (lungcap$Gender == "M") & (lungcap$Smoke == 1)
MN <- (lungcap$Gender == "M") & (lungcap$Smoke == 0)
tb <- tibble(Gender = c("Female", "Male"),
             sz.NS = c(sum(FN), sum(MN)),
             mn.Age.NS = c(mean(lungcap$Age[FN]),
                           mean(lungcap$Age[MN])),
             mn.Ht.NS = c(mean(lungcap$Ht[FN]),
                          mean(lungcap$Ht[MN])),
             mn.FEV.NS = c(mean(lungcap$FEV[FN]),
                           mean(lungcap$FEV[MN])),
             sz.S =  c(sum(FS), sum(MS)),
             mn.Age.S = c(mean(lungcap$Age[FS]),
                          mean(lungcap$Age[MS])),
             mn.Ht.S = c(mean(lungcap$Ht[FS]),
                         mean(lungcap$Ht[MS])),
             mn.FEV.S = c(mean(lungcap$FEV[FS]),
                          mean(lungcap$FEV[MS])))
kbl(tb,
    row.names = FALSE,
    col.names = c("Gender",
                  "Obs.", "Age", "Height", "FEV",
                  "Obs.", "Age", "Height", "FEV"),
    digits = c(0, 0, 1, 1, 2, 0, 1, 1, 2),
    booktabs = TRUE) %>%
  add_header_above(c(" " = 1, " " = 1, "Mean" = 3,
                     " " = 1, " Mean" = 3)) %>%
  add_header_above(c(" " = 1, "Non-Smoker" = 4, "Smoker" = 4))
rm(FS, MS, FN, MN, tb)
```

From our exploratory analysis we have learned that both age and height are strongly related to our response variable: forced expiratory volume (FEV). As FEV increases in mean value, its variability also increases and thus using an normal distribution would not be supported by the data. In fact, the data suggests that a gamma distribution is appropriate. Also, gender and smoker status seem to play a role in influencing the response.

Table 2.3: Mean age, height, and forced expiratory volume (FEV) by gender and smoker status. The number of observations (Obs.) in each cell are also given.

	Non-Smoker				Smoker			
		Mean				Mean		
Gender	Obs.	Age	Height	FEV	Obs.	Age	Height	FEV
Female	279	9.4	59.6	2.38	39	13.3	64.6	2.97
Male	310	9.7	61.5	2.73	26	13.9	68.1	3.74

2.3 Modeling and Diagnostics

From our exploratory data analysis we can propose an initial model with the following specification: the response variable is forced expiratory volume (FEV) and we will model it as a gamma distribution. The explanatory variable height (Ht) should be included in the linear predictor. Perhaps entering as a linear term, but the right-hand side panel of Figure 2.1 shows that the relationship is not perfectly linear and thus we may need to use a transformation to obtain a better model. Gender and smoker status should also be included in the model, but we will build the model in stages adding one variable at a time.

Before embarking on our gamma model we want to illustrate how using ordinary least squares regression (that is, assuming that our response variable is normally distributed) would not be optimal.

Let us fit an ordinary least squares regression model to FEV and Ht and display a plot of residuals versus fitted values.

```
ols.fit <- glm(FEV ~ Ht,
               data = lungcap,
               family = gaussian(link = "identity"))
lungcap <- lungcap %>%
  mutate(olsfit.mu = predict(ols.fit, type = "response"),
         olsfit.rD = resid(ols.fit, type = "deviance"))
```

```
p <- ggplot(data = lungcap,
            mapping = aes(x = olsfit.mu,
                          y = olsfit.rD)) +
  geom_point() +
  labs(x = "Fitted Values, FEV (in liters)",
       y = "Residuals")
q <- ggplot(data = lungcap,
            mapping = aes(x = olsfit.mu,
```

```
                          y = abs(olsfit.rD))) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Fitted Values, FEV (in liters)",
       y = "Abs. Value of Residuals")
p + q
rm(p, q)
```

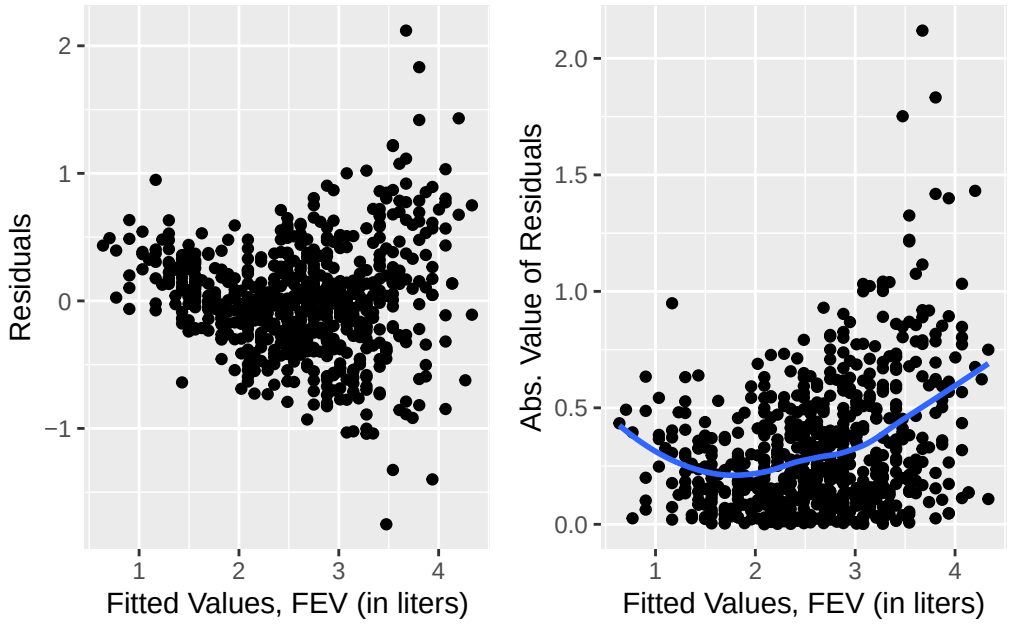


Figure 2.3: Fitted values versus deviance residuals for an ordinary least squares model for forced expiratory volume that includes height as an explanatory variable. Note the strong pattern in both panels telling us that our model is not adequate.

Notice how, in the left-hand panel of Figure 2.3, as the fitted values increase the vertical spread of the residuals also increases. This indicates that the constant variance assumption of the residuals is not met. The right-hand panel is a modification of the left-hand panel where we plot the absolute value of the residuals and include a smoothing trend line. This small alteration gives us a more nuanced view into the changes of spread as fitted values increase. For this plot, we see that the spread of residuals first decrease and then increase substantially.

Having seen that the normal distribution does not capture the true nature of our data, let us use what we learned during our exploratory data analysis and switch over to using a gamma distribution. We will fit several models and encode the key characteristics in the name. For example, a gamma model with an identity link function and having as main effects height and gender would be written as: `gi.HG.fit`.

The general scheme is:

- First letter identifies the distribution: (n: normal, p: Poisson, g: gamma, i: inverse gaussian, b: binomial, v: negative binomial, t: Tweedie)
- Second letter stands for the link function: (i: identity $g(x) = x$, l: logarithmic $g(x) = \log(x)$, r: reciprocal or inverse $g(x) = 1/x$, s: square root $g(x) = \sqrt{x}$, o: logit $g(x) = \log(x/(1-x))$)[1]
- the next group of letters indicates which variables are in the linear predictor (A: age, H: height, G: gender, S: Smoking)

For a numeric variable we may add a number, like `2`, to show that we have a polynomial of second degree in that variable as part of the linear predictor. And finally, we add the word `fit` to signal that we have a fitted generalized linear model.

To illustrate, the ordinary least squares model that we fitted above, `ols.fit`, would be named `ni.H.fit` using the proposed scheme (normal distribution, identity link, and main effect height).

Next we will fit a gamma generalized linear model to `FEV` using height as an explanatory variable and keeping the link function as the identity. The model name is `gi.H.fit`. Before performing this fit, we should should develop an idea of what the sign and size of the estimated coefficient should be.

Based on Figure 2.1 we expect the coefficient for height, `Ht`, to be positive and roughly equal to $4/30 \approx 0.13$ (the line connecting the points $(45, 1)$ and $(75, 5)$ seems a reasonable approximation).

```
gi.H.fit <- glm(FEV ~ Ht,
                data = lungcap,
                family = Gamma(link = "identity"))
(sgi.H.fit <- summary(gi.H.fit))
```

```

Call:
glm(formula = FEV ~ Ht, family = Gamma(link = "identity"), data = lungcap)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.530471   0.132547  -34.18   <2e-16 ***
Ht           0.117013   0.002296   50.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[1] Other link functions are possible, such as, probit, complementary log-log, and inverse squared.

```
(Dispersion parameter for Gamma family taken to be 0.02452508)

    Null deviance: 70.791  on 653  degrees of freedom
Residual deviance: 16.159  on 652  degrees of freedom
AIC: 641.83

Number of Fisher Scoring iterations: 5
```

The coefficient for `Ht` is roughly in line with our expectations and note that the standard errors for both estimates are quite small compared to the size of the estimate.

Is our model a reasonable representation of the data? One way to try to answer this question uses a technique knows as *predictive simulation* (Gelman and Hill 2007). The basic idea is to fit a model to your data, then replicate the data from this fitted model, and finally compare the actual data with the replicates. If you can distinguish the actual data from the replicates, then your fitted model is not a very good representation of the actual data. And if the actual data and the replicates are indistinguishable, then you have a good model.

Using our gamma, identity link, with variable height as a main effect; that is, model `gi.H.fit`, we can simulate new data sets and compare against our actual data. Using the heights in our data set the following code simulates three sets of the response variable, `FEV`, from the appropriate gamma distribution.

```
set.seed(19390349)
n <- nrow(lungcap)
disp <- sgi.H.fit$dispersion
lungcap <- lungcap %>%
  mutate(giHfit.mu = predict(gi.H.fit, type = "response"),
         giHfit.rp1 = rgamma(n,
                             shape = 1/disp,
                             scale = giHfit.mu * disp),
         giHfit.rp2 = rgamma(n,
                             shape = 1/disp,
                             scale = giHfit.mu * disp),
         giHfit.rp3 = rgamma(n,
                             shape = 1/disp,
                             scale = giHfit.mu * disp))
rm(n,disp)
```

In Figure 2.4 we have four panels showing `FEV` versus height. Three of the panels have the simulated data from our model and one panel has the actual data. Can you tell which panel has the actual data?

When comparing simulated data against actual data we should leverage everything we learned during our exploratory data analysis. Two key features we saw in the previous section were

1. the variability in `FEV` increases as height increases
2. the relationship between `FEV` and height is convex

Which panel in Figure 2.4 corresponds to the actual data?

```
p <- ggplot(data = lungcap,
            mapping = aes(x = Ht)) + ylim(0,6) +
  labs(x = "Height (in inches)",
       y = "FEV (in liters)")
p1 <- p + geom_point(mapping = aes(y = giHfit.rp3),
                     alpha = 0.2)
p2 <- p + geom_point(mapping = aes(y = FEV),
                     alpha = 0.2)
p3 <- p + geom_point(mapping = aes(y = giHfit.rp1),
                     alpha = 0.2)
p4 <- p + geom_point(mapping = aes(y = giHfit.rp2),
                     alpha = 0.2)
(p1 + p2) / (p3 + p4)
rm(list = ls(pattern = "^p"))
```

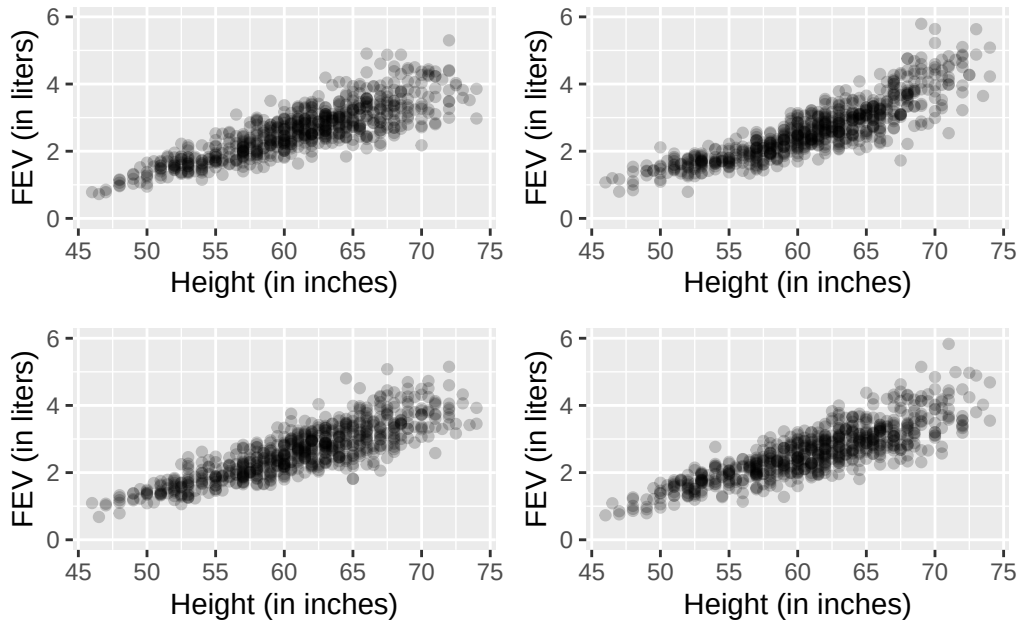


Figure 2.4: One panel contains the actual data and the other panels have simulated data from a fitted model. Can you identify the panel with the actual data?

In all four panel, we see that the variability in the response variable `FEV` increases as height increases. Thus feature 2 above holds for all four panels. Can you see which panel violates feature 1?

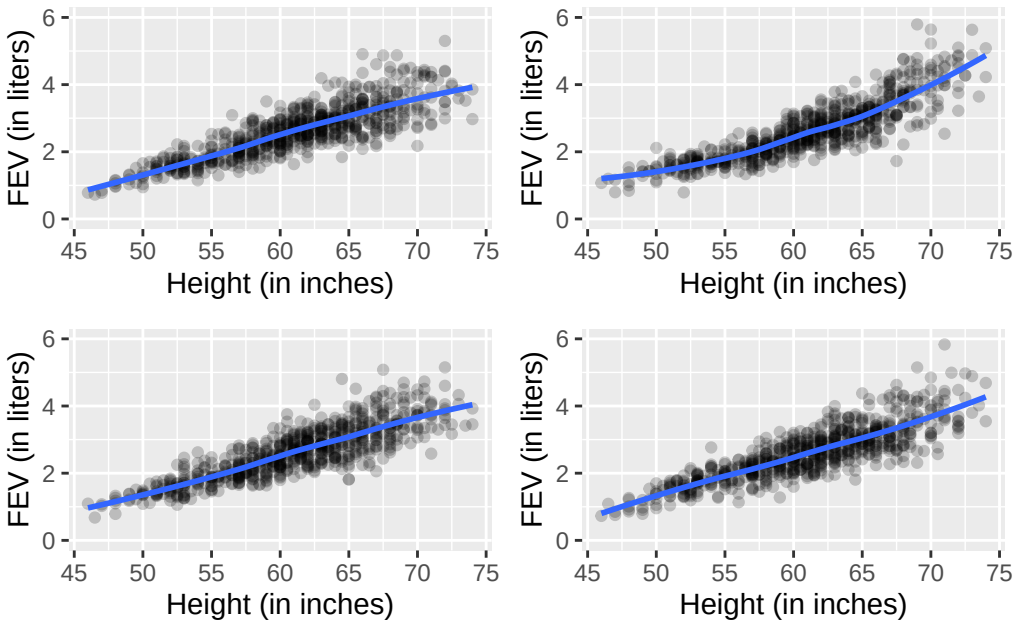
Exercise

Recreate the four graphs but include a smoothing trend line for each graph to help you see the underlying relationship between `FEV` and `Ht`.

Solution

Adding a smooth trend line to a scatter plot can be done with a locally weighted regression procedure such as loess or lowess (Cleveland 1979). These methods are implemented in the `geom_smooth()` function and can be used as follows:

```
p <- ggplot(data = lungcap,
            mapping = aes(x = Ht)) + ylim(0,6) +
  labs(x = "Height (in inches)",
       y = "FEV (in liters)")
p1 <- p + geom_point(mapping = aes(y = giHfit.rp3),
                     alpha = 0.2) +
  geom_smooth(aes(y = giHfit.rp3),
              se = FALSE)
p2 <- p + geom_point(mapping = aes(y = FEV),
                     alpha = 0.2) +
  geom_smooth(aes(y = FEV),
              se = FALSE)
p3 <- p + geom_point(mapping = aes(y = giHfit.rp1),
                     alpha = 0.2) +
  geom_smooth(aes(y = giHfit.rp1),
              se = FALSE)
p4 <- p + geom_point(mapping = aes(y = giHfit.rp2),
                     alpha = 0.2) +
  geom_smooth(aes(y = giHfit.rp2),
              se = FALSE)
(p1 + p2) / (p3 + p4)
```




```
rm(list = ls(pattern = "^p"))
```

Note that the upper right-hand is the only panel where the trend line is convex. All other trend lines are essentially straight lines.
Therefore, model `gi.H.fit` does not capture the convexity between the response variable `FEV` and the predictor variable height (`Ht`).

The curvature that we observe between `FEV` and `Ht` could be modeled via a quadratic polynomial in `Ht` or by using a log-link function. Let us fit both models and apply some diagnostics. The quadratic model in height will be named `gi.H2.fit` (gamma distribution, identity link function, height and height squared as predictors) and the log-link model with height is `gl.H.fit` (gamma distribution, log-link function, and height as main effect).

```
lungcap$Ht.sq <- lungcap$Ht^2
gi.H2.fit <- glm(FEV ~ Ht + Ht.sq,
                 data = lungcap,
                 family = Gamma(link = "identity"))
gl.H.fit <- glm(FEV ~ Ht,
                data = lungcap,
                family = Gamma(link = "log"))
```

The coefficients for model the quadratic model (`gi.H2.fit`) are

```
round(coef(gi.H2.fit), 5)
```

```
(Intercept)          Ht       Ht.sq
    5.34181    -0.22664     0.00296
```

and so we can calculate that the minimum value for the curve of predictions from this model occurs at a subject's height equal to

$$\frac{0.22664}{2 \cdot 0.00296} \approx 38.28$$

inches. This value is outside the range of our data, but is reasonably close and very plausible by continuing the smooth trend shown in the right-hand panel of Figure 2.1.

The coefficients for the log-link model (`gl.H.fit`) are

```
round(coef(gl.H.fit), 4)
```

```
(Intercept)          Ht
    -2.2679      0.0522
```

Thus we can infer that as the height for a child increases by one inch, their forced expiratory volume will increase by approximately 5.4% ($e^{0.0522} - 1$).

```
lungcap <- lungcap %>%
  mutate(giH2fit.mu = predict(gi.H2.fit, type = "response"),
         giH2fit.eta = predict(gi.H2.fit, type = "link"),
         giH2fit.rW = resid(gi.H2.fit, type = "working"),
         giH2fit.wR = giH2fit.eta + giH2fit.rW,
         giH2fit.rD = resid(gi.H2.fit, type = "deviance"),
         giH2fit.rQ = qresid(gi.H2.fit))
```

```
lungcap <- lungcap %>%
  mutate(glHfit.mu = predict(gl.H.fit, type = "response"),
         glHfit.eta = predict(gl.H.fit, type = "link"),
         glHfit.rW = resid(gl.H.fit, type = "working"),
         glHfit.wR = glHfit.eta + glHfit.rW,
         glHfit.rD = resid(gl.H.fit, type = "deviance"),
         glHfit.rQ = qresid(gl.H.fit))
```

The left-hand panels of Figure 2.5 show the fitted values versus quantile residuals and the linear predictor versus working residuals for the quadratic model in height (`Ht`). The right-hand panels show the same plots for the log-link model.

Quantile residuals were introduced in (Dunn and Smyth 1996) and an excellent overview of them appears in (Dunn and Smyth 2018). Pearson and deviance residuals are the standard choices when analyzing the adequacy of fits for generalized linear models. Both of them are approximately normal with deviance residuals being a bit more so, but for discrete distributions the approximation to normality can be particularly bad. Quantile residuals overcome these issues and are strongly recommended for discrete models and we can use them just as we would use deviance or Pearson residuals for diagnostic purposes.

The top panels display fitted values versus quantile residuals and we can see, in both plots, a nice random cloud of points centered about the line $y = 0$. There appear to be two outlying points in both plots below the line $y = -4$.

The bottom panels are an *informal* diagnostic on the link function. The plot shows the linear predictor, $\hat{\eta}_i$ on the y-axis and the working response

$$z_i = \hat{\eta}_i + e_i$$

on the x-axis (Dunn and Smyth 2018, 308). Note that the working response, z_i, is the sum of the linear predictor and the *working residuals*, e_i. The working residuals are the residuals from the last iteration of the Fisher scoring algorithm used to compute estimates for the coefficients of the model.

If the link function is correct and we have the appropriate explanatory variables in our model and on the right scale, then we expect the points to cluster around the line $y = x$. Major deviations from this null pattern are a red flag that something is wrong with our model. In this case, both panels show the appropriate behavior, but the log-link model has a more cohesive cloud of points around the reference line.

```
p1 <- ggplot(data = lungcap,
             mapping = aes(x = giH2fit.mu,
                           y = giH2fit.rQ)) +
  geom_point() +
  labs(x = "Fitted Values, FEV (in liters)",
       y = "Quantile Residuals",
       title = "Quadratic Model")

p2 <- ggplot(data = lungcap,
             mapping = aes(x = glHfit.mu,
                           y = glHfit.rQ)) +
  geom_point() +
  labs(x = "Fitted Values, FEV (in liters)",
```

```
       y = "Quantile Residuals",
       title = "Log-link Model")

p3 <- ggplot(data = lungcap,
             mapping = aes(x = giH2fit.wR,
                           y = giH2fit.eta)) +
  geom_point() + geom_abline(intercept = 0,
                             slope = 1,
                             color = "red") +
  labs(x = "Working Response",
       y = "Linear Predictor")

p4 <- ggplot(data = lungcap,
             mapping = aes(x = glHfit.wR,
                           y = glHfit.eta)) +
  geom_point() + geom_abline(intercept = 0,
                             slope = 1,
                             color = "red") +
  labs(x = "Working Response",
       y = "Linear Predictor")

(p1 + p2)/(p3 + p4)
rm(list = ls(pattern = "p[1-4]"))
```

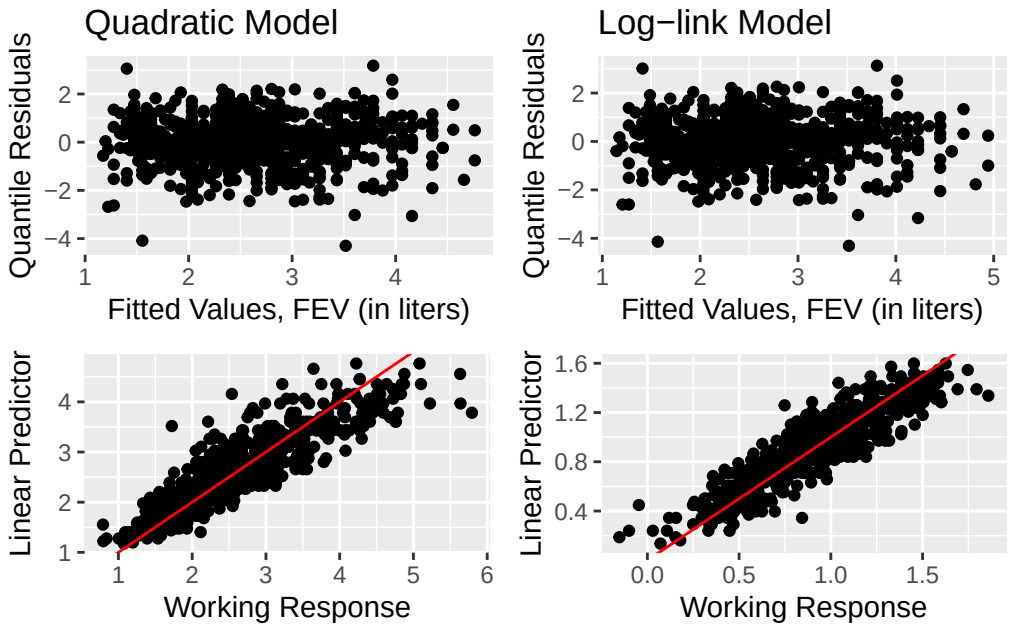


Figure 2.5: Fitted values versus quantile residuals and working response versus linear predictor for the quadratic as well as the log-link model. All four plots display the desired null pattern and in the top panels we may have two outlying observations (residuals below the horizontal line at $y = -4$).

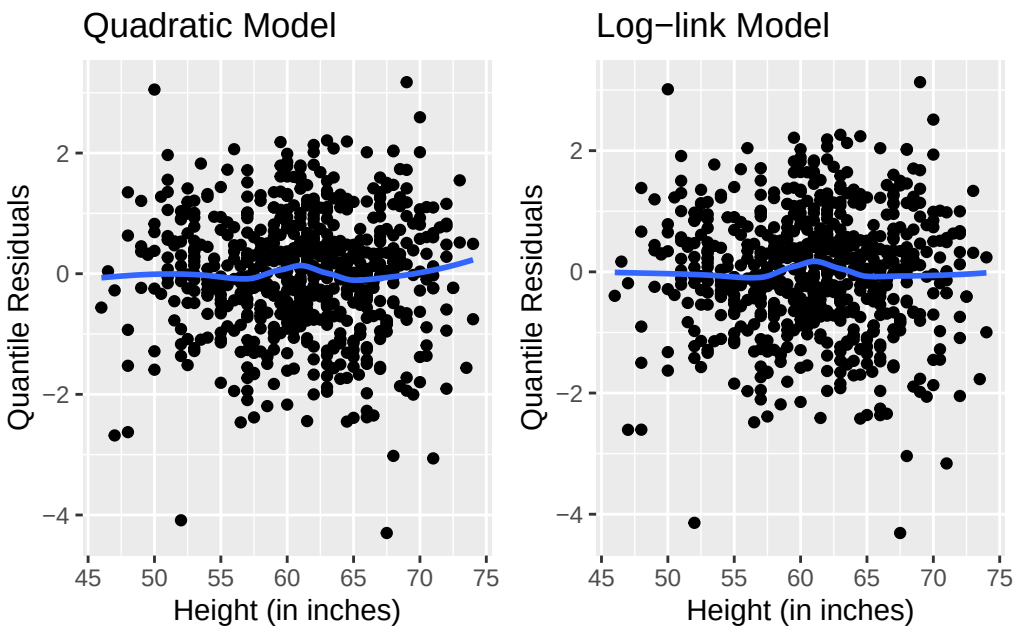
Exercise

Plot the quantile residuals versus height for both models. Are there any concerning patterns in the plots?

Solution

We can plot the quantile residuals for both models with the following code:

```
p1 <- ggplot(data = lungcap,
             mapping = aes(x = Ht,
                           y = giH2fit.rQ)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Height (in inches)",
       y = "Quantile Residuals",
       title = "Quadratic Model")
p2 <- ggplot(data = lungcap,
             mapping = aes(x = Ht,
                           y = glHfit.rQ)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Height (in inches)",
       y = "Quantile Residuals",
       title = "Log-link Model")

p1 + p2
```




```
rm(p1, p2)
```

Both panels show a random cloud of points that are centered about the line $y = 0$. Near 60 inches in height we see an upward bump on the scatterplot smoother (blue line) indicating that our models tend to underpredict in that region. But the bump is fairly small.

Currently, our best model uses a gamma distribution and a log-link function. The model equation is

$$\log\left(\mathbb{E}[\text{FEV}]\right) = \beta_0 + \beta_1 \text{Ht}.$$

Next, we incorporate the `Gender` categorical variable and the indicator variable for `Smoke`. This indicator variable has a value of 1 for subjects who smoke and 0 otherwise. We would expect the coefficient for `Smoke` to be negative because we think that smoking would have a detrimental effect on our lungs and thus diminish lung capacity. The absolute value of the coefficient will tell us how much lung capacity will be affected by smoking. As for `Gender`, the size and direction of the effect is not clear.

```
gl.HGS.fit <- glm(FEV ~ Smoke + Gender + Ht,
                  data = lungcap,
                  family = Gamma(link = "log"))
sgl.HGS.fit <- summary(gl.HGS.fit)
round(coef(sgl.HGS.fit), 4)
```

```
            Estimate Std. Error  t value Pr(>|t|)
(Intercept)  -2.2615     0.0639 -35.3758   0.0000
Smoke         0.0001     0.0201   0.0055   0.9956
GenderM       0.0187     0.0117   1.5994   0.1102
Ht            0.0520     0.0011  48.8380   0.0000
```

Even though the coefficient for `Smoke` has a positive sign, there is no evidence in the data to suggest that it is different from zero. Similarly, the effect for `Gender` is not statistically significant.

These results **do not show** that children who smoke do not impair their lung capacity.

```
lungcap <- lungcap %>%
  mutate(glHGSfit.mu = predict(gl.HGS.fit, type = "response"),
         glHGSfit.eta = predict(gl.HGS.fit, type = "link"),
         glHGSfit.rW = resid(gl.HGS.fit, type = "working"),
         glHGSfit.wR = glHGSfit.eta + glHGSfit.rW,
         glHGSfit.rD = resid(gl.HGS.fit, type = "deviance"),
         glHGSfit.rQ = qresid(gl.HGS.fit))
```

```
p1 <- ggplot(data = lungcap,
             mapping = aes(x = glHGSfit.mu,
                           y = glHGSfit.rQ)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Fitted Values, FEV (in liters)",
```

```
       y = "Quantile Residuals")
p2 <- ggplot(data = lungcap,
             mapping = aes(x = glHGSfit.mu,
                           y = abs(glHGSfit.rQ))) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Fitted Values, FEV (in liters)",
       y = "|Quantile Residuals|")
p3 <- ggplot(data = lungcap,
             mapping = aes(x = Ht,
                           y = glHGSfit.rQ)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Height (in inches)",
       y = "Quantile Residuals")
p4 <- ggplot(data = lungcap,
             mapping = aes(sample = glHGSfit.rQ)) +
  stat_qq() +
  geom_abline(intercept = 0, slope = 1, color = "red") +
  labs(x = "Theoretical Quantiles",
       y = "Sample Quantiles")
p5 <- ggplot(data = lungcap,
             mapping = aes(x = Gender,
                           y = glHGSfit.rQ)) +
  geom_boxplot() +
  labs(x = "Gender",
       y = "Quantile Residuals") +
  scale_x_discrete(labels = c("Female", "Male"))
p6 <- ggplot(data = lungcap,
             mapping = aes(x = factor(Smoke),
                           y = glHGSfit.rQ)) +
  geom_boxplot() +
  labs(x = "Smoking Status",
       y = "Quantile Residuals") +
  scale_x_discrete(labels = c("No", "Yes"))
(p1 + p2) / (p3 + p5) / (p6 + p4)
rm(list = ls(pattern = "^p"))
```

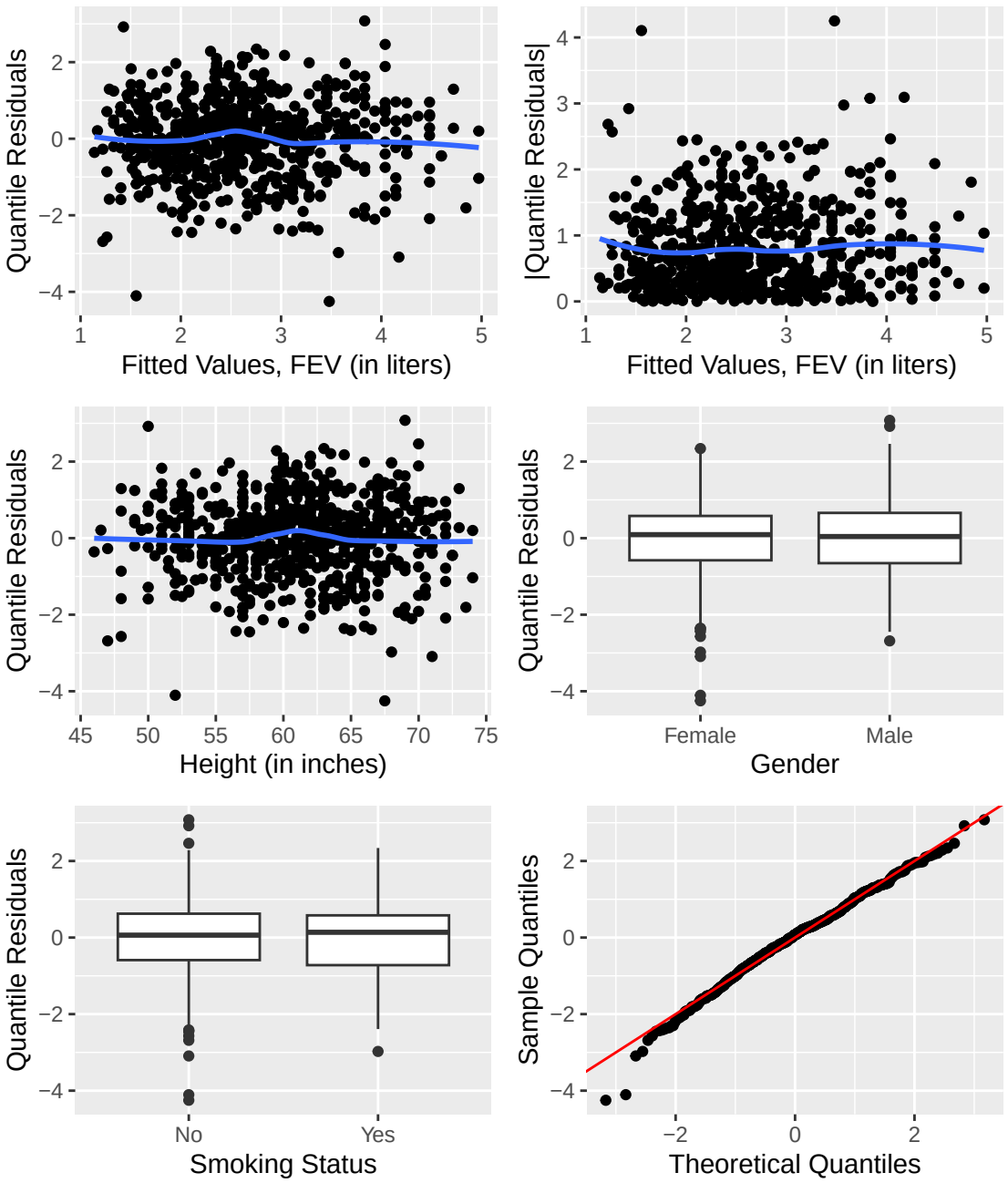


Figure 2.6: Diagnostic plots for our final model. The two left-hand panels show a random cloud of points centered about the line $y = 0$. The trend line exhibits a small positive bump in the center of the display. The upper right-hand panel shows that our model captures the increasing variablity in the response variable well. The QQ-plot on the lower right-hand panel shows that lower tail of our data is thicker than it should be.

Figure 2.6 shows some diagnostic plots for our final model `gl.HGS.fit`; gamma distribution, log-link, and main effects for height, gender, and smoking status. All six plots show that our model fits the data well. On the bottom right-hand panel we can see that the bulk of the data follows the line $y = x$, but we also see a slight deviation from the null pattern in the lower tail of the distribution. In this case, our quantile residuals have a slightly thicker lower tail than a standard normal distribution but the number of points exhibiting this behavior is very small.

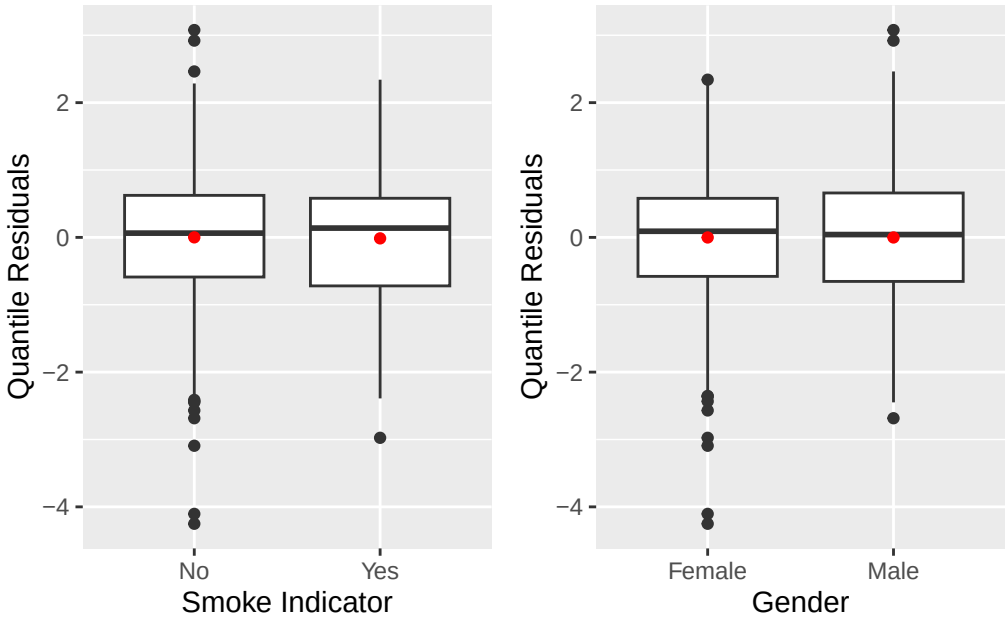
Exercise

In Figure 2.6 we include diagnostic plots for the variables `Smoke` and `Gender`, but did not include any commentary. Also the boxplot do not show the mean value.
Recreate these two boxplots and add a point showing the mean value of the residuals for each category and comment on what information do they contribute to the final model.

Solution

To add the mean value to the boxplot we will use a `stat_summary()` function to compute the mean and add a point to the display.

```
p1 <- ggplot(data = lungcap,
             mapping = aes(x = as.factor(Smoke),
                           y = glHGSfit.rQ)) +
  geom_boxplot() +
  stat_summary(fun = mean,
               geom = "point",
               color = "red") +
  labs(x = "Smoke Indicator",
       y = "Quantile Residuals") +
  scale_x_discrete(labels = c("No", "Yes"))
p2 <- ggplot(data = lungcap,
             mapping = aes(x = as.factor(Gender),
                           y = glHGSfit.rQ)) +
  geom_boxplot() +
  stat_summary(fun = mean, geom = "point", color = "red") +
  labs(x = "Gender",
       y = "Quantile Residuals") +
  scale_x_discrete(labels = c("Female", "Male"))
p1 + p2
```




```
rm(p1, p2)
```

For both variables, the mean of the residuals are centered at zero, they have equal spread, and a few outlying points. There are no indications that our model is missing any information from these variables.
Two points have quantile residuals whose absolute value is greater than 4. These points require further investigation. Perhaps the value of the response or predictor variables was entered incorrectly.

2.4 Summary

This chapter focused on refreshing some of the main concepts for generalized linear models by working through a concrete example. Generalized linear models provide a richer set of regression models for the analyst to draw upon. Many GLMs exhibit a mean-variance relationship of the form $\text{Var}[y] = \phi\mu^b$, where ϕ is the dispersion parameter, μ is the mean of the distribution, and the value of b determines the distribution. Specific values are as follows: $b = 0$ normal (that is, we are back to ordinary least squares), $b = 1$ Poisson, $1 < b < 2$ Tweedie, $b = 2$ gamma, and $b = 3$ inverse gaussian.

In many situation we can use the data to estimate the mean-variance relationship and select an appropriate distribution. We also showed several diagnostic plots, such as the standard residuals versus fitted values, absolute value of residuals against fitted values, an informal

check on the link function by plotting the linear predictor against the working responses, and a QQ-plot of residuals.

3 Credibility Theory

```
library(tidyverse)
library(patchwork)
library(kableExtra)
library(GLMsData)
library(statmod)
library(actuar)
library(lme4)
```

```
source("buehlmann-gisler-calculations.R")
```

3.1 Introduction

Consider the following scenario: you are preparing the renewal offer for a policyholder that has been insured for a number of years. While there are many approaches to setting next year's premium, consider the following two extreme positions:

1. base it entirely on the historical claims experience of this policyholder; that is, use their average claims experience.
2. completely ignore this policyholder's own claims experience and use the *complement of credibility*; such as, the insurance company's average claims experience for all similar policyholders or industry loss history.

Both positions are extreme. In the first one, you are in essence saying that your policyholder's experience is completely trustworthy for setting next year's premium. Maybe your policyholder is so large and their claims experience so stable, that barring any extraordinary events, next year's claims will be spot on with their historical record. In adopting the second position, you acknowledge that this policyholder's claim experience is not trustworthy (whether good, bad, or mixed) and so you look for the overall average claims for the entire portfolio of policies that this policyholder belongs to or to industry loss experience.

These two extreme positions are not the only alternatives. There is some middle ground, where we can blend some of this policyholder's historical experience together with the experience of the block of business where this policyholder belongs. Credibility theory is the body of

knowledge, tools, and techniques that allows us to blend these two extreme positions into a far better estimate for our policyholder next year's premium.

Quoting from (Venter 1996)

> *Credibility*, simply put, is the weighting together of different estimates to come up with a combined estimate. For instance, an insured's own experience might suggest a different premium from that in the manual. These are two different estimates of the needed premium, which can be combined using credibility concepts to yield an adjusted premium.

And we can summarize it in a formula as

$$\text{AP} = Z \times \text{EP} + (1 - Z) \times \text{MP} \tag{3.1}$$

where AP is the adjusted premium, EP is the own experience premium, MP is the manual premium, and $Z \in [0, 1]$ is the credibility factor. The adjusted premium is also known as the *credibility premium*. Even though Equation 3.1 is a deceptively simple interpolation formula between EP and MP it has far reaching consequences and applications.

Note that as Z approaches 1, the adjusted premium gets closer to the own experience premium. And as Z approaches 0, the adjusted premium converges to the manual premium. Thus, if the insured's own experience is highly credible (Z close to 1), we would assign an adjusted premium close to its own experience. If the experience is not credible, then we would assign a premium close to the premium suggested by the manual.

The key question is how to calculate the credibility factor based on observed data. An intuitive understanding is that the **more extensive** the observed data is and the **less it fluctuates**, then the closer the credibility factor will be to 1.

Credibility theory began a little over 100 years with the publication of (Mowbray 1914) whose title "How Extensive a Payroll is Necessary to Give a Dependable Pure Premium" succinctly encapsulated one of the main problems facing casualty actuaries at that time. In the intervening time, credibility theory has developed tremendously and today there are many approaches and directions. Practicing actuaries are most familiar with two main methods of calculating credibility:

1. Limited Fluctuation Credibility
2. Greatest Accuracy Credibility

Limited fluctuation credibility is also known as classical credibility and greatest accuracy credibility as Bühlmann credibility.

In the next section, we give a very brief introduction to classical credibility and then move on to a more extensive treatment of Bühlmann credibility.

3.2 Limited Fluctuation Credibility

Limited fluctuation credibility is the oldest development and it started with the publication of (Mowbray 1914). We will closely follow chapter 5 of (Herzog 2010).

Consider the following aggregate loss model:

$$S = X_1 + X_2 + \cdots + X_N, \tag{3.2}$$

where S is the aggregate loss, the X_i are individual losses, and N represents the number of losses. (If $N = 0$, then we take $S = 0$.) We assume that the random variables N and X_i are independent of each other. We also assume that the individual losses, X_i, are independent and identically distributed with mean μ and variance σ^2. And finally, we assume that the number of claims, N, follows a Poisson distribution with mean λ (and consequently the variance of N is also equal to λ).

With these assumptions we can see that the mean value of S is equal to

$$\mathbb{E}[S] = \mathbb{E}[N] \cdot \mathbb{E}[X] = \lambda \cdot \mu.$$

To calculate the variance of S we need the following result concerning conditional means and variances: if Y and Z are random variables, then we have

$$\mathrm{Var}[Z] = \mathbb{E}_Y[\mathrm{Var}_Z[Z|Y]] + \mathrm{Var}_Y[\mathbb{E}_Z[Z|Y]]; \tag{3.3}$$

that is, the variance of Z is equal to the mean of the conditional variance plus the variance of the conditional mean.

We can use the above result to calculate the variance of the aggregate claims. Let $S = Z$ and $Y = N$ and we have

$$\begin{aligned}
\mathrm{Var}[S] &= \mathbb{E}_N[\mathrm{Var}_S[S|N]] + \mathrm{Var}_N[\mathbb{E}_S[S|N]] \\
&= \mathbb{E}_N\left[\mathrm{Var}\left[\sum_{i=1}^{N} X_i|N\right]\right] + \mathrm{Var}_N\left[\mathbb{E}_S\left[\sum_{i=1}^{N} X_i|N\right]\right] \\
&= \mathbb{E}_N\left[N \cdot \sum_{i=1}^{N} \mathrm{Var}[X_i]\right] + \mathrm{Var}_N\left[N \cdot \mathbb{E}_S[X_i]\right] \\
&= \mathbb{E}_N[N] \cdot \mathrm{Var}[X_i] + \left(\mathbb{E}_S[X_i]\right)^2 \cdot \mathrm{Var}_N[N] \\
&= \lambda \cdot \sigma^2 + \mu^2 \cdot \lambda \\
&= \lambda \cdot (\sigma^2 + \mu^2)
\end{aligned}$$

Summarizing, we consider a insured's total loss for a period to be given by the aggregate loss model, $S = X_1 + \cdots + X_N$ where the individual losses have a mean equal to μ and a variance of σ^2, and the number of claims in the period, N, follows a Poisson distribution with mean λ. Hence, we have

$$\mathbb{E}[S] = \lambda \cdot \mu \quad \text{and} \quad \mathrm{Var}[S] = \lambda \cdot (\sigma^2 + \mu^2)$$

Exercise

Suppose that an insured, over the next accounting period, is expected to have 10 claims and each claim, on average, costs 50 thousand dollars and has a variance equal to 625 thousand dollars squared.
What is the expected total cost for this insured and its standard deviation?

Solution

We have $\lambda = 10$ thousand dollars, $\mu = 50$ thousand dollars, and $\sigma^2 = 625$ thousand dollars squared. Therefore, the expected value will be 500 $(= 10 \cdot 50)$ thousand dollars and the standard deviation is

$$\sqrt{10 \cdot (625 + 50^2)} = 176.777$$

thousand dollars.

Now that we know the mean and the variance of our aggregate losses we can talk about what a dependable pure premium is. In essence, we want our pure premium to be stable; that is, we do not want our aggregate loss to fluctuate too much. We can achieve this by having a large large enough book of business.

We can frame this stability as follows: with 90% probability we want our pure premium to be within 5% of its true value. The 90% probability and the 5% margin are parameters for us to choose. This stability statement can be translated as follows

$$\Pr\left(0.95\lambda\mu < S < 1.05\lambda\mu\right) = 0.9$$

Equivalently, by subtracting the mean and dividing by the standard deviation of S,

$$\Pr\left(\frac{-0.05\lambda\mu}{\sqrt{\lambda(\sigma^2+\mu^2)}} < \frac{S-\lambda\mu}{\sqrt{\lambda(\sigma^2+\mu^2)}} < \frac{0.05\lambda\mu}{\sqrt{\lambda(\sigma^2+\mu^2)}}\right) = 0.9 \tag{3.4}$$

And since S is a sum of independent and identically distributed random variables we can appeal to the central limit theorem and treat the middle term as a normal random variable with mean zero and unit variance.

Central Limit Theorem

Suppose that $X_1, X_2, ...$ is a sequence of independent and identically distributed random variables, each having mean μ and variance σ^2. Define the sum $S_n = \sum_{i=1}^{n} X_i$, so that $\mathbb{E}[S_n] = n\mu$ and $\text{Var}[S_n] = n\sigma^2$. Then

$$\lim_{n\to\infty} \Pr\left(\frac{S_n - n\mu}{\sigma\sqrt{n}} \leq x\right) = \Phi(x)$$

where Φ is the cumulative distribution function of normal random variable with mean zero and unit variance.

From Equation 3.4 we can solve the following equation for λ, the expected number of claims,

$$\frac{0.05\lambda\mu}{\sqrt{\lambda(\sigma^2+\mu^2)}} = 1.645$$

giving us

$$\lambda = \left(\frac{1.645}{0.05}\right)^2 \left(1 + \left(\frac{\sigma}{\mu}\right)^2\right). \tag{3.5}$$

Note that the first factor contains our selections for probability (90%) and the margin (5%). The second factor only depends on the distribution of the individual losses via the coefficient of variation (σ/μ). This last equation tells us the minimum number of expected claims that a portfolio of policies should have if we want to consider it to be *fully credible*. In this case, fully credible means that we want aggregate losses to be within 5% of its mean with a 90% probability.

As an illustration, consider an insurer's auto physical damage portfolio. The distribution of auto physical damage losses tends to have a small coefficient of variation (σ/μ), say around 10%. Then the criterion for **full credibility** would be to have at least

$$\left(\frac{1.645}{0.05}\right)^2 (1 + 0.1^2) = 1{,}082 \cdot 1.01 = 1{,}093$$

expected claims in the portfolio.

If the line of business is more volatile, for example, professional liability, then the coefficient of variation may be about 50%. In this case we would need at least 1,353 expected claims to reach full credibility.

Exercise

Construct two tables of full credibility standards. One based on a coefficient of variation of 0% (a fixed payout), 10% (say, auto physical damage), and another one with 40% (say, medical malpractice). Use margins of 0.01, 0.05, 0.1 and probabilities equal to 0.9, 0.95, 0.975, 0.99.

Solution

Set up the margins, probabilities, and the coefficients of variation.

```
margin <- c(0.01, 0.05, 0.1)
probs <- c(0.9, 0.95, 0.975, 0.99)
cv <- c(0.00, 0.10, 0.40)
```

Compute the entries of the tables using Equation 3.5 with the appropriate entries.

```
dnm <- list(paste(probs * 100, "%", sep = ""),
            paste(margin * 100, "%", sep = ""))
alpha <- 1 - probs
qs <- qnorm(1 - alpha/2, mean = 0, sd = 1)
tb <- outer(qs, margin, function(x,y) {(x/y)^2})
tb1 <- tb * (1 + cv[1]^2)
tb2 <- tb * (1 + cv[2]^2)
tb3 <- tb * (1 + cv[3]^2)
dimnames(tb1) <- dnm
dimnames(tb2) <- dnm
dimnames(tb3) <- dnm
```

For a line of business with fixed payouts (that is, a coefficient of variation equal to zero) we have

```
round(tb1,0)
```

```
         1%   5% 10%
90%   27055 1082 271
95%   38415 1537 384
97.5% 50239 2010 502
99%   66349 2654 663
```

Display the table for the line of business with a coefficient of variation equal to 10%.

```
round(tb2, 0)
```

```
         1%   5% 10%
90%   27326 1093 273
95%   38799 1552 388
97.5% 50741 2030 507
99%   67012 2680 670
```

Display the table for the other line of business (40% coefficient of variation).

```
round(tb3, 0)
```

```
          1%   5% 10%
90%    31384 1255 314
95%    44561 1782 446
97.5% 58277 2331 583
99%    76965 3079 770
```

Thus far we have developed the expected number of claims necessary for a portfolio to be considered fully credible. If we have such a fully credible portfolio, then we are confident in using its own experience and so in the credibility formula shown in Equation 3.1 and reproduced here

$$\text{AP} = Z \times \text{EP} + (1 - Z) \times \text{MP}$$

where AP is the adjusted premium, EP is the portfolio experience premium, and MP stands for the manual premium, we would set the credibility factor Z equal to one.

Now consider the situation where our portfolio is not fully credible; that is, our portfolio lacks the expected number of claims necessary to be fully credible. In this case, we would like to blend some of its own experience (which we deem *partially credible*) with the manual premium. In other words, what value between 0 and 1 should we select for the credibility factor Z?

If you reach full credibility, then it is clear that we should set $Z = 1$. But what should we do if we do not have enough expected claims to reach full credibility? How big should Z be?

One way to think about this situation is that our portfolio's claim experience is too volatile to be within the parameters we have selected, say a margin of 5% and a probability of 90%. We can consider taming that volatility by multiplying our claims experience by a factor Z and choosing its value so that we meet the criterion for full credibility; that is, let Z be such that

$$\Pr\left(-0.05\lambda\mu < ZS - Z\lambda\mu < 0.05\lambda\mu\right) = 0.9$$

Dividing the above expression by Z and the standard deviation of S we get:

$$\Pr\left(\frac{-0.05\lambda\mu}{Z\sqrt{\lambda(\sigma^2+\mu^2)}} < \frac{S - \lambda\mu}{\sqrt{\lambda(\sigma^2+\mu^2)}} < \frac{0.05\lambda\mu}{Z\sqrt{\lambda(\sigma^2+\mu^2)}}\right) = 0.9$$

And finally solving for Z and rearranging a few terms we have

$$Z = \sqrt{\frac{\lambda}{\left(\frac{1.645}{0.05}\right)^2\left(1 + \left(\frac{\sigma}{\mu}\right)^2\right)}}$$

And we see that the credibility factor Z is equal to the square root of the ratio of expected claims for the portfolio to the full credibility standard.

```
fc <- (1.645/0.05)^2 * (1 + 0.1^2)
x <- seq(0, 1300)
y <- pmin(sqrt(x / fc), 1)
p <- ggplot(data = tibble(x = x, y = y),
            mapping = aes(x = x, y = y)) +
  geom_line() +
  labs(x = "Expected Number of Claims",
       y = "Credibility Factor (Z)")
p <- p + geom_point(data = tibble(x = 1093.23, y = 0),
                    mapping = aes(x = x, y = y),
                    color = "red")
p <- p + annotate(geom = "text",
                  x = 1093.23,
                  y = 0.25,
                  label = "Full\nCredibility",
                  color = "red")
p <- p + geom_segment(data = tibble(x = 1093.23,
                                    y = 0.17,
                                    xend = 1093.23,
                                    yend = 0.02),
                      aes(x = x, y = y,
                          xend = xend, yend = yend),
                          arrow = arrow(length = unit(0.03, "npc")),
                          color = "red")
p <- p + geom_segment(data = tibble(x = 1093.23,
                                    y = 0.32,
                                    xend = 1093.23,
                                    yend = 0.98),
                      aes(x = x, y = y,
                          xend = xend, yend = yend),
                          arrow = arrow(length = unit(0.03, "npc")),
                          color = "red")
p
rm(fc, x, y, p)
```

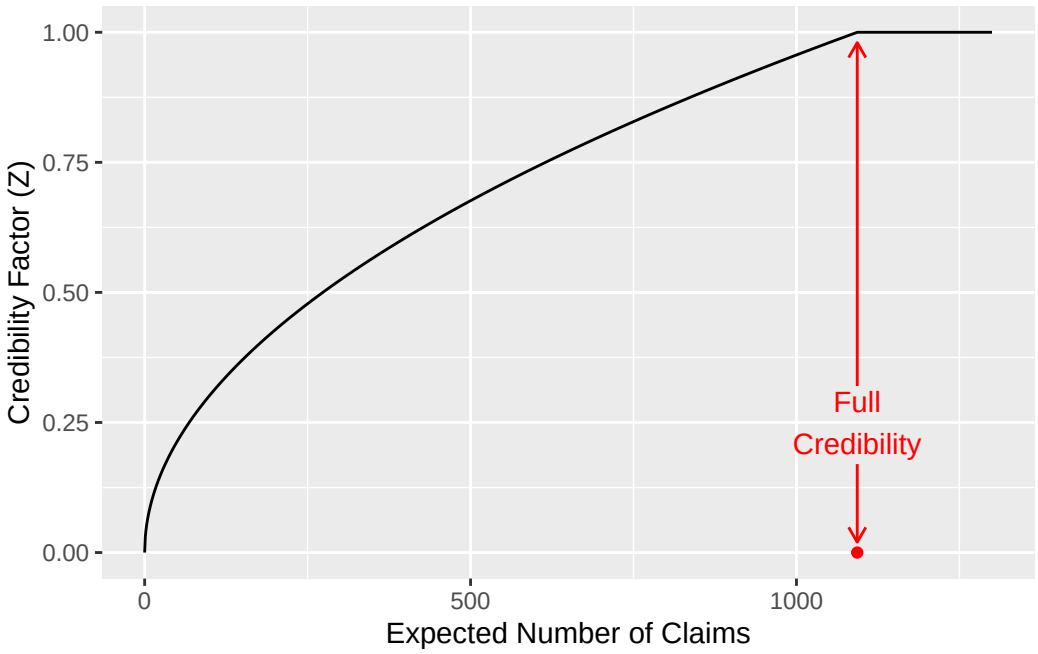


Figure 3.1: Limited fluctuation credibility factors. The coefficient of variation is 10% with a margin of 5% and a probability equal to 90%. Full credibility is reached with 1,093 expected claims. Note that if a portfolio has 250 expected claims, then it would have a credibility factor of almost 50%. A credibility factor of 75% requires about 625 claims.

3.3 Greatest Accuracy Credibility

Greatest accuracy credibility, also known as *Bühlmann credibility*, was developed in the 1960's by (Hans Bühlmann 1967), who derived the optimal credibility factors by minimizing a squared error in the context of a Bayesian statistical model.

We will start with a basic model and expand to a more complex treatment. While the basic model is too simple to be effectively used in practice, it is important for understanding how more complex models work. Our development closely follows the presentations in (Straub 1997) and (Kaas 2009).

To keep things concrete, consider the following example (a slightly modified version of problem 5.84 from (Klugman, Panjer, and Willmot 1998, 500)). We have a portfolio of policyholders where we have three, $(J = 3)$, different risk classes that we have observed their claims experience over the last four, $T = 4$, years. Let X_{jt} be the experience for risk class $j = 1, 2, \ldots, J$ in time period $t = 1, 2, \ldots, T$. Table 3.1 shows the data and Figure 3.2 provides a graphical representation.

Table 3.1: Claims experience for a portfolio of three risk classes that have been observed over four calendar years.

	Time			
Class	1	2	3	4
1	625	675	600	700
2	750	800	650	800
3	900	700	850	950

```
dta <- tibble(class = factor(rep(1:3, each = 4),
                             levels = 1:3),
              time = rep(1:4, times = 3),
              value = c(625, 675, 600, 700,
                        750, 800, 650, 800,
                        900, 700, 850, 950))
```

```
dtb <- pivot_wider(dta,
                   names_from = time,
                   values_from = value)
```

```
kbl(dtb,
    col.names = c("Class", "1", "2", "3", "4"),
    align = "ccccc",
    booktabs = TRUE) %>%
  kable_styling(full_width = FALSE) %>%
  add_header_above(c(" " = 1, "Time" = 4))
```

We would like to estimate the experience each risk class will have during the next year ($T = 5$).

```
ggplot(data = dta,
       mapping = aes(x = time,
                     y = value,
                     group = class,
                     pch = class)) +
  geom_line(color = "gray") +
  geom_point(size = 2) +
  xlim(1,5) +
  annotate("text",
           x = rep(5,3),
```

```
          y = c(834.375, 750, 665.625),
          label = rep("?", 3)) +
  labs(x = "Observation Year",
       y = "Claims Experience")
```

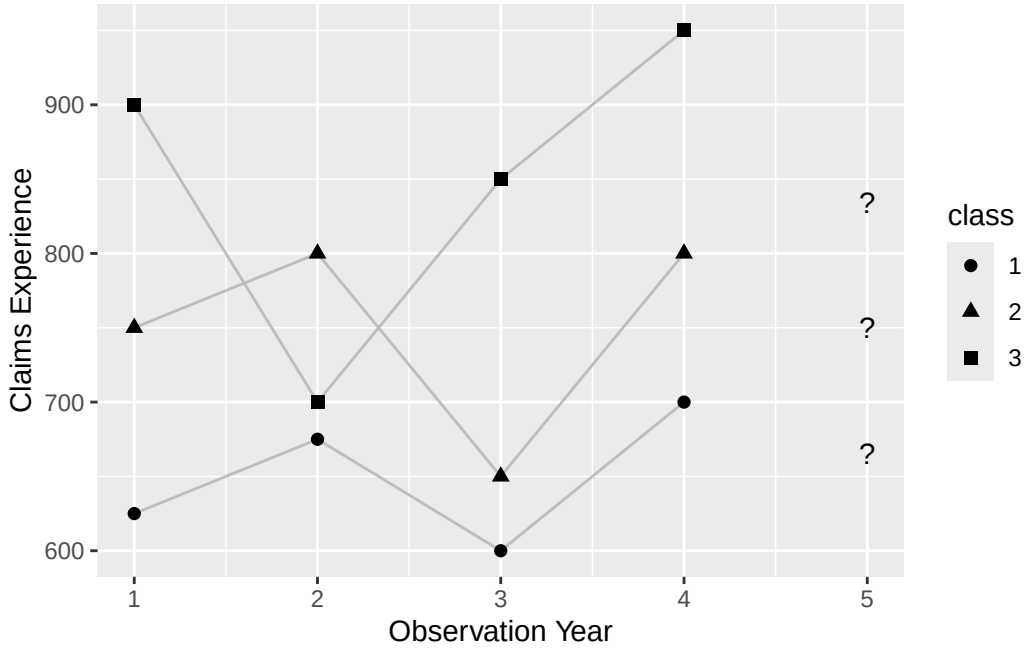


Figure 3.2: Claims experience for a portfolio of three risk classes that have been observed over four years. What should the estimate, for each risk class, be in year five?

Looking closely at Figure 3.2 and focusing on each risk class at a time we could say the following: if we continue observing these risk classes for many years (and assuming that these risks are stable over time), each one of them would fluctuate around a mean claim cost, say $\bar{X}_j = (\sum_{t=1}^{T} X_{jt})/T$. For example, looking at risk class $j = 3$ (square symbol) which starts in year 1 with a value of 900, it seems plausible that its long-term average cost might be around $\bar{X}_3 = 850$. For risk class $j = 2$ (triangle symbol) with claim cost $X_{21} = 750$ at time $t = 1$; its long-term average might be close to $\bar{X}_2 = 750$ and for risk class $j = 1$ (circle symbol), starting with $X_{11} = 625$ that long-term average may equal $\bar{X}_1 = 650$. The portfolio as a whole also has a long-term average claim cost that, in this case, would be around $\bar{X} = 750$.

From the experience that we see in Figure 3.2 we might be inclined to say that these three risk classes have **different** long-term claim averages. Is there evidence in this data that this is the case? How might we quantify such evidence?

One way to quantify the evidence for or against different long-term averages would be to use a statistical model for this data. To this end, the experience X_{jt} for risk class $j = 1, 2, \ldots, J$

in time period $t = 1, 2, \ldots, T$, could be decomposed as

$$X_{jt} = m_j + \epsilon_{jt},$$

where m_j is the mean for risk class j and ϵ_{jt} represents an error term. We assume that the error terms are independent and identically distributed with $\epsilon_{jt} \in N(0, \sigma^2)$. Hence all the X_{jt} are independent and $N(m_j, \sigma^2)$ distributed, with possibly unequal means m_j, but all with equal variance σ^2, even across risk classes. We can test for the equality of all group means via an analysis of variance.

The analysis of variance entails computing two statistics that will be relevant for credibility calculations. The first one is the *sum of squares between*

$$\text{SSB} = \sum_{j=1}^{J} T(\bar{X}_j - \bar{X})^2.$$

This statistic has $J-1$ degrees of freedom. The second statistic is the *sum of squares within*

$$\text{SSW} = \sum_{j=1}^{J} \sum_{t=1}^{T} (X_{jt} - \bar{X}_j)^2,$$

and it has $J(T-1)$ degrees of freedom.

Under the assumption that the group means, m_j, are equal (this is the null hypothesis), the random variable SSB has a mean equal to $(J-1)\sigma^2$ and the random variable SSW has a mean equal to $J(T-1)\sigma^2$. The ratio of these means follows a Fisher distribution with $J-1$ and $J(T-1)$ degrees of freedom:

$$F = \frac{\text{MSB}}{\text{MSW}} = \frac{\text{SSB}/(J-1)}{\text{SSW}/(J(T-1))}$$

For our example, we can calculate the sum of squares between (SSB) and the mean sum of squares between (MSB) as follows:

```
J <- length(levels(dta$class))
Tm <- length(unique(dta$time))
X_jt <- dta$value

X.bar <- mean(X_jt)
Xj.bar <- tapply(X_jt, dta$class, mean)
SSB <- Tm * sum((Xj.bar - X.bar)^2)
MSB <- SSB/(J - 1)
```

The sum of squares within (SSW) and the mean sum of squares within (MSW) are

```
SSW <- sum((X_jt - rep(Xj.bar, each = Tm))^2)
MSW <- SSW / (J * (Tm - 1))
```

Their values are:

```
c("SSB" = SSB, "MSB" = MSB, "SSW" = SSW, "MSW" = MSW)
```

```
  SSB   MSB   SSW   MSW
80000 40000 56250  6250
```

And so we have that the F-statistic, its critical value at 5%, and its p-value are

```
round(c("F.stat" =  MSB / MSW,
"5% Cr.Vl" = qf(0.95, J - 1, J * (Tm - 1)),
"p-value" = pf(MSB/MSW, J - 1, J * (Tm - 1),
               lower.tail = FALSE)), 4)
```

```
  F.stat 5% Cr.Vl  p-value
  6.4000   4.2565   0.0187
```

So in this case, we have evidence that at least two of the means, m_1, m_2, m_3, are not equal (we are able to reject the null hypothesis of equal means) and thus we would consider our portfolio to be heterogeneous.

Had the F-statistic been below the critical value, then we would not have been able to reject the null hypothesis that all the means are equal and in this case we would consider our portfolio to be homogeneous.

The calculations we just did for the sum of squares between, sum of squares within, and F-statistic can be easily done by fitting a linear model to the data and creating an analysis of variance table.

```
fm <- lm(value ~ class, data = dta)
anova(fm)
```

```
Analysis of Variance Table

Response: value
          Df Sum Sq Mean Sq F value  Pr(>F)
class      2  80000   40000     6.4 0.01867 *
Residuals  9  56250    6250
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Exercise

Consider the claims experience for class 3 which has values equal to

```
dta$value[dta$class == 3]
```

```
[1] 900 700 850 950
```

Suppose we subtract the same amount m from each of these values to bring them closer to the values for classes 1 and 2.
What is the smallest value of m such that we would no longer consider our portfolio heterogeneous? In other words, what value of m would yield an F-statistic equal to its critical value at the 5% level?

Solution

We are looking for the smallest value of m so that the F-statistic for our portfolio is equal to 4.256. To search for the value of m we can construct a function of one argument, m, so that its minimum value is achieved at the value of m that we are looking for.
For a given value of m we would need to perform the following steps:

1. Decrease all the values for class 3 by m
2. Fit a linear model to the data
3. Compute the F-statistic for this data, `f.val`
4. Return the square of the difference between `f.val` and the given critical value

The following function implements these steps. The argument `c.value` is the target critical value we want to achieve, `dt` is the data frame containing our portfolio, and `cls` is the class we want to modify.

```
f <- function(m, c.value, dt, cls) {
  idx <- dt$class == cls
  dt$value[idx] <- dt$value[idx] - m
  fm <- lm(value ~ class, data = dt)
  f.val <- anova(fm)[1,4]
  ans <- (f.val - c.value)^2
  return(ans)
}
```

Now that we have our function we can search for its minimum value via `optimize()`. The first argument is the function we want to minimize. The second argument provides an interval to conduct the search. The remaining named arguments are needed by our function `f` to do its calculations. Based on Figure 3.2 it seems reasonable to assume that m should be in the interval from zero to 100.

```
optimize(f, c(0, 100),
         c.value = qf(0.95, 2, 9),
         dt = dta,
         cls = 3)
```

```
$minimum
[1] 38.41004

$objective
[1] 4.28568e-16
```

Therefore, when we reduce the experience for class 3 by $m = 38.41$ we obtain a portfolio that we would consider homogeneous.
Note that in the analysis of variance table shown before the exercise, it is the residual sum of squares; namely, the sum of squares within, remains constant regardless of the value of m. It is the sum of squares between; that is, the `class` sum of squares, that gets smaller as m increases. Thus, the denominator of the F-statistic is fixed while the numerator gets smaller.

In the analysis above, we have treated the risk class means, m_j, as fixed but unknown. If our portfolio is heterogeneous, we may try to find a way to relate these means to other information we may have about the risk classes. In a practical application, we may have thousands of risk classes. Think about a classification system for automobile insurance with variables such as age, gender, socioeconomic status, years licensed, prior claims, garage location, make and model of car, safety features, engine size, and so forth. Many of these cells would be common and have plenty of data, but many would also be rare and have little data. Our linear model would have to estimate parameters for all these risk categories and this would present a significant estimation problem. Moreover, as the number of risk classes increases, so do the number of parameters that we need to estimate.

Another way to look at our portfolio would be to assume that the risk class mean, m_j, is a random draw from a distribution. Thus we would decompose our data as follows:

$$X_{jt} = \mu + \Xi_j + \epsilon_{jt}, \qquad j = 1, 2, \dots, J, \quad t = 1, 2, \dots, T, \tag{3.6}$$

where Ξ_j (the capital version of the letter ξ) and ϵ_{jt} are independent random variables with mean zero and

$$\text{Var}[\Xi_j] = \tau^2, \qquad \text{Var}[\epsilon_{jt}] = \sigma^2.$$

Note that from Equation 3.6 we have

$$\mathbb{E}[X_{jt}] = \mu \quad \text{and} \quad \text{Var}[X_{jt}] = \tau^2 + \sigma^2;$$

that is, the variance of each cell is equal to the sum of the variance for each component in Equation 3.6.

In terms of our portfolio we can interpret the above model as follows: the overall mean is given by μ and it is the expected value of claim costs for a contract picked at random from our portfolio. The term Ξ_j is a random variable and it represents a deviation from the grand mean μ specific to risk class j. The conditional mean of X_{jt} given that $\Xi_j = \xi$ is equal to $\mu + \xi$. This would be the long-term average for risk class j. The variance of Ξ_j is equal to τ^2 and so this parameter controls how spread out individual risk classes are from the overall mean. A large value of τ^2 would lead to a heterogeneous portfolio. A small value of τ^2 suggests a homogeneous portfolio where all risk classes have similar long-term means. The last component, ϵ_{jt}, gives us a deviation for risk class j from its long-term mean $\mu + \xi$ in year t. It represents the fluctuation of the experience around its long-term average.

It is important to note that this model has three parameters: the overall mean μ, the variance component τ^2, and another variance component σ^2. These three parameters are independent of the number of risk classes in our portfolio. We may have just three risk classes, as in the example above, but we could also have hundreds or thousands of risk classes and we would still only need to estimate three parameters.

As depicted in Figure 3.2 with the questions marks at time $t = 5$, we are interested in estimating the value of the unobserved random variables $X_{j,T+1}$. While there may be many ways of estimating this random variable, we will require it to be a linear combination of the observed data that we have, namely, $X_{11}, X_{12}, \ldots, X_{JT}$. We also want our linear combination to have the same expected value as $X_{j,T+1}$ (we want our estimator to be unbiased) and that its squared error be smallest among all possible linear combinations.

The following theorem (Kaas 2009, Theorem 8.2.2) tells us that the best estimate for the next period is a credibility weighted average between $\bar{X}_j$ and $\bar{X}$, where the weight depends on the number of observed periods, T, and the variance components τ^2 and σ^2.

Theorem 3.1 (Balanced Bühlmann; homogeneous estimator)**.** *Assume that the claim figures X_{jt} for contract j in period t can be written as the sum of stochastically independent components, as follows:*

$$X_{jt} = \mu + \Xi_j + \epsilon_{jt}, \quad j = 1, 2, \ldots, J, \quad t = 1, 2, \ldots, T+1, \tag{3.7}$$

where the random variables Ξ_j are independent and identically distributed with mean $\mathbb{E}[\Xi_j] = 0$ and $Var[\Xi_j] = \tau^2$, and also the random variables ϵ_{jt} are independent and identically distributed with $\mathbb{E}[\epsilon_{jt}] = 0$ and $Var[\epsilon_{jt}] = \sigma^2$ for all j and t. Furthermore, assume that the variables Ξ_j to be independent of the ϵ_{jt}.

Under these conditions, the homogeneous linear combination $g_{11}X_{11} + \cdots + g_{JT}X_{JT}$ that is the best unbiased predictor of $X_{J,T+1}$ in the sense of minimal mean squared error (MSE)

$$\mathbb{E}[(X_{J,T+1} - g_{11}X_{11} - \cdots - g_{JT}X_{JT})^2] \tag{3.8}$$

equals the credibility premium

$$z\bar{X}_j + (1-z)\bar{X} \tag{3.9}$$

where

$$z = \frac{\tau^2 T}{\tau^2 T + \sigma^2} = \frac{T}{T + \sigma^2/\tau^2}$$

is the resulting best credibility factor (which in this case is equal for all j),

$$\bar{X} = \frac{1}{JT} \sum_{j=1}^{J} \sum_{t=1}^{T} X_{jt} \tag{3.10}$$

is the collective estimator of μ, and

$$\bar{X}_j = \frac{1}{T} \sum_{t=1}^{T} X_{jt} \tag{3.11}$$

is the individual estimator of m_j.

Non-parametric estimators of τ^2, σ^2, and μ are developed in (Klugman, Panjer, and Willmot 1998, sec. 5.5.1). The overall mean μ can be estimated via $\bar{X}$. To estimate σ^2, also known as the expected value of the process variance (EVPV), consider

$$\hat{\sigma}_j^2 = \frac{1}{T-1} \sum_{t=1}^{T} (X_{jt} - \bar{X}_j)^2. \tag{3.12}$$

This is an estimate for the variance of risk class j and we can take the average of these estimates

$$\hat{\sigma}^2 = \frac{1}{J} \sum_{j=1}^{J} \hat{\sigma}_j^2 = \frac{1}{J(T-1)} \sum_{j=1}^{J} \sum_{t=1}^{T} (X_{jt} - \bar{X}_j)^2 \tag{3.13}$$

to obtain the **expected value of the process variance** (EVPV) (Klugman, Panjer, and Willmot 1998, equation 5.75).

To estimate the **variance of the hypothetical means** (VHM) we use the relationship (Klugman, Panjer, and Willmot 1998, 465)

$$\text{Var}[\bar{X}_j] = \tau^2 + \frac{\sigma^2}{T}. \tag{3.14}$$

The left-hand side is equal to the mean square between (MSB) and so we have that our estimator for τ^2 is

$$\hat{\tau}^2 = \frac{1}{J-1} \sum_{j=1}^{J} (\bar{X}_j - \bar{X})^2 - \frac{1}{TJ(T-1)} \sum_{j=1}^{J} \sum_{T=1}^{T} (X_{jt} - \bar{X}_j)^2 \tag{3.15}$$

All three estimators, $\hat{\mu}, \hat{\tau}^2, \hat{\sigma}^2$ are unbiased. Note that the estimator for $\hat{\tau}^2$ is the difference between two expressions, and so in practice it may yield a negative answer. This is clearly

nonsense as we are estimating a variance component. In these cases, it is common to set its value equal to zero and to take the credibility factor as $z = 0$. Intuitively, this makes sense. If the variance of the hypothetical means is zero (or close to zero), then all risk classes have very similar individual means that do not differ from the overall mean.

Using the data for the example we can implement the above formulas. There are many ways to organize the data necessary for these calculations and we will take an approach that closely corresponds to the above formulas even though it may not be the best approach for a large scale project. Thus, our first step will be to set some key variables, such as J, T, and X_{jt}, and sort the data, X_{jt}, by class and time period.

```
J <- length(levels(dta$class))
Tm <- length(unique(dta$time))
cls <- dta$class

o <- order(dta$class, dta$time)
Xjt <- dta$value[o]
```

First we calculate the overall mean, $\bar{X}$, and the mean for each risk class, $\bar{X}_j$ using Equation 3.10 and Equation 3.11.

```
X.bar <- mean(Xjt)
Xj.bar <- tapply(Xjt, cls, mean)
```

Next we calculate $\hat{\sigma}_j^2$ (Equation 3.12) and the expected value of the process variance (EVPV), $\hat{\sigma}^2$ using Equation 3.13.

```
sigmaj.sq <- tapply(
  (Xjt - rep(Xj.bar, each = Tm))^2, cls, sum) / (Tm - 1)
sigma.sq <- mean(sigmaj.sq)
```

Next comes the calculation of $\text{Var}[\bar{X}_j]$ via Equation 3.14.

```
Var.Xj.bar <- sum((Xj.bar - X.bar)^2) / (J - 1)
```

And finally, the calculation of the variance of the hypothetical means (Equation 3.15).

```
tau.sq <- Var.Xj.bar - sigma.sq / Tm
```

With all these values we have that our credibility factor is equal to

$$Z = \frac{T}{T + \hat{\sigma}^2/\hat{\tau}^2} = \frac{4}{4 + 6250/8437.5} = 0.84375,$$

and the credibility premiums will be equal to

$$Z\bar{X}_j + (1 - Z)\bar{X}$$

yielding the following credibility weighted premiums:

```
Z <- Tm / (Tm + sigma.sq / tau.sq)
Z * Xj.bar + (1 - Z) * X.bar
```

```
      1       2       3
665.625 750.000 834.375
```

Adding these forecasts to our earlier graph gives us Figure 3.3.

```
tb <- tibble(class = factor(c(1,2,3,1,2,3)),
             time = c(4, 4, 4, 5, 5, 5),
             value = c(700, 800, 950, 665.625, 750, 834.375))
ggplot(data = dta,
       mapping = aes(x = time,
                     y = value,
                     group = class,
                     pch = class)) +
  geom_line(color = "gray") +
  geom_line(data = tb,
            mapping = aes(x = time,
                          y = value,
                          group = class),
            color = "red") +
  geom_point() +
  geom_point(data = subset(tb, time == 5),
             mapping = aes(x = time,
                           y = value),
             color = "red") +
  xlim(1,5) +
  annotate("text",
           x = rep(5,3),
           y = c(834.375, 750, 665.625),
           label = rep("?", 3)) +
  labs(x = "Observation Year",
       y = "Claims Experience")
```

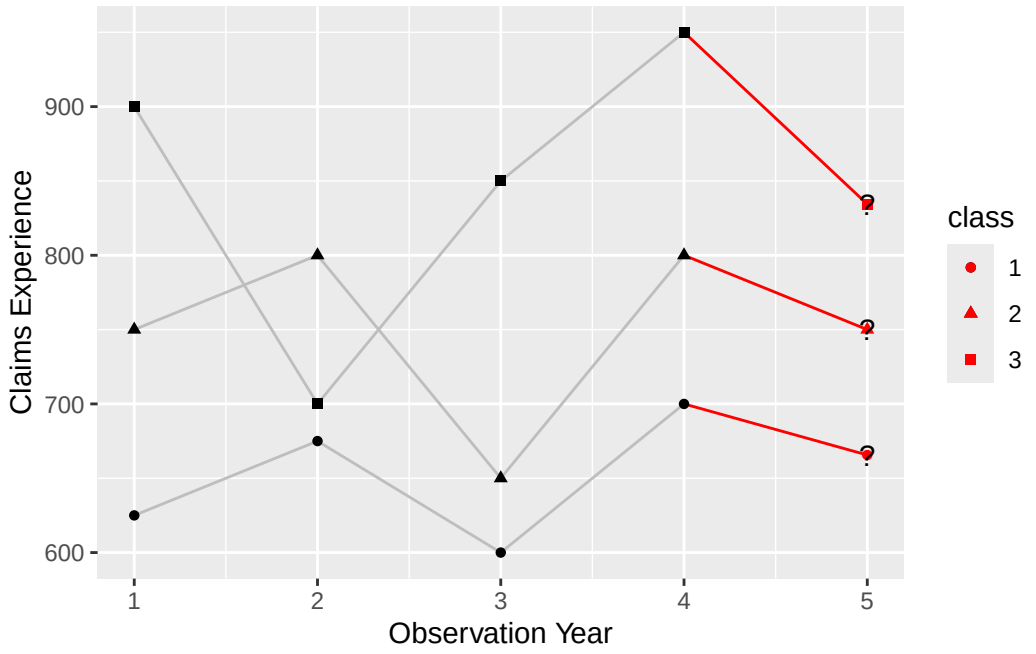


Figure 3.3: Claims experience for a portfolio of three risk classes that have been observed over four years. The credibility weighted estimate for the next year is shown

The above calculations for the *balanced* Bühlmann model and other credibility models, have been coded into the `cm()` function of the `actuar` R package. We illustrate its use next. The data is required to be in a different format where the time variable is expressed through different columns in the data set and the rows represent the different contracts or classes.

```
dtb <- pivot_wider(dta,
                   names_from = time,
                   values_from = value)
dtb
```

```
# A tibble: 3 x 5
  class   `1`   `2`   `3`   `4`
  <fct> <dbl> <dbl> <dbl> <dbl>
1 1       625   675   600   700
2 2       750   800   650   800
3 3       900   700   850   950
```

In this case columns 2 through 5 represent our data. The balanced Bühlmann model, `BB`, can be fit as follows and a summary of the fitted object is:

```
BB <- cm( ~ class,
          data = dtb,
          ratios = 2:5)
summary(BB)
```

```
Call:
cm(formula = ~class, data = dtb, ratios = 2:5)

Structure Parameters Estimators

  Collective premium: 750

  Between class variance: 8437.5
  Within class variance: 6250

Detailed premiums

    class Indiv. mean Weight Cred. factor Cred. premium
    1     650         4      0.84375      665.625
    2     750         4      0.84375      750.000
    3     850         4      0.84375      834.375
```

Here the collective premium is $\bar{X}$, the variance of the hypothetical means, τ^2, has been labeled "Between class variance," and the expected value of the process variance, σ^2, is the "Within class variance." In the section "Detailed premiums" we see that the individual means, $\bar{X}_j$ are in the second column, the next column "Weight" has the number of observed time periods, T, the credibility factor and the credibility premiums (the last two columns) also match our previous calculations.

```
rm(cls, J, o, sigma.sq, sigmaj.sq, tau.sq, Tm,
   Var.Xj.bar, X.bar, Xj.bar, Xjt, Z, tb, BB, dtb)
```

The form of the credibility factor may not look particularly nice

$$Z = \frac{T}{T + \sigma^2/\tau^2},$$

but it has some very appealing and intuitive properties:

1. Since all elements involved are positive, the credibility factor is also positive. Moreover, regardless of the values of T, σ^2, or τ^2, its value is always between zero and one.

2. As the number of periods of observation, T, increases, the credibility factor increases towards 1
3. As the expected value of the process variance (EVPV), σ^2 decreases, the credibility factor increases towards 1
4. As the variance of the hypothetical means (VHM), τ^2, increases, the credibility factor increases towards 1

All these make sense. The more observations you have about your insureds, all else the same, the more confident you should be about their experience. If the process variance is very small, then you should also be more confident about their experience. A small process variance means that the insured's claims experience does not fluctuate too much. And finally, if the variance of the hypothetical means is large, then you know that your insureds have different means, and so you should be more confident about using their own experience compared to imposing the overall average experience.

Another extremely important property of the Bühlmann credibility model is that to calculate next period's premiums we only need to estimate two parameters: the expected value of the process variance (EVPV), also known as the within class variance, σ^2, and the variance of the hypothetical means (VHM), also known as the between class variance, τ^2. This is always the case regardless of the number of levels that the class variable might have.

Unfortunately, the balanced Bühlmann model is not always applicable in practice. One shortcoming is that in the decomposition of the experience, X_{jt},

$$X_{jt} = m + \Xi_j + \epsilon_{jt}$$

we have assumed that the deviation, Ξ_j, from the overall mean, m, for each risk class, j, has the same variance; namely, $\text{Var}[\Xi_j] = \sigma^2$. In other words, all risk classes have, in essence, been measured with the same precision. In practice, this may not be a good assumption.

Imagine that the experience X_{jt} is actually the average over individual policyholders that belong in the j-th risk class. In this case, the variance across risk classes will not be the same since risk classes will, most likely, have different number of policyholders. Another reason we may not have equal variances, even though we may have same number of policyholders, comes about by having policyholders of different sizes within the same risk class. Consider a small supermarket versus a large one.

In these cases we would want to consider the experience X_{jt} along with a weight w_{jt} such that the bigger the weight, the smaller the variance and vice versa.

In the next section, we present the Bühlmann-Straub model that takes care of these two issues.

3.4 The Bühlmann-Straub Model

In the last section we saw that the balanced Bühlmann credibility model assumes that each observation in risk class j and time t, X_{jt}, has the same variance and this may not always reflect reality.

In this section, we introduce the Bühlmann-Straub model which incorporates different weights into the balanced Bühlmann model. We start with the same decomposition of the observations, X_{jt}, as in the previous section

$$X_{jt} = m + \Xi_j + \epsilon_{jt}, \quad j = 1, 2, \ldots, J, t = 1, 2, \ldots, T+1, \tag{3.16}$$

where the unobservable risk components Ξ_j (deviations from the overall mean m for risk class j) are independent and identically distributed with mean zero, and the components ϵ_{jt} (deviations across time from the long term average of risk class j) are also independent and identically distributed with zero mean. We also assume that Ξ_j and ϵ_{jt} are independent of each other.

Next we keep the variance of Ξ_j the same as in the previous section

$$\text{Var}[\Xi_j] = \tau^2,$$

but change the assumption for the variance of the components ϵ_{jt} to include the weights, w_{jt}, to

$$\text{Var}[\epsilon_{jt}] = \frac{\sigma^2}{w_{jt}}. \tag{3.17}$$

Note that if we set each of the weights, w_{jt}, equal to 1; that is, let $w_{jt} = 1$ for all j and t, then we are back to the balanced Bühlmann model.

Just as in the balanced Bühlmann model, we would like to find the best homogeneous *unbiased* linear predictor $\sum h_{jt} X_{jt}$ of the risk premium $m + \Xi_j$. The following theorem, from (Kaas 2009, Theorem 8.4.1, p. 215), provides the answer using the following quantities and notation. An open circle, '$\circ$', in an index location means that we are summing across that index. A filled circle '$\bullet$', in an index location means that we are creating a weighted average over that index with weights provided by the appropriate w_{jt}.

The first expression below sums across all time periods and so we put an open circle on the time index. The second expression sums across both indices, time and risk class, and so two open circles are used.

$$w_{j\circ} = \sum_{t=1}^{T} w_{jt} \quad \text{and} \quad w_{\circ\circ} = \sum_{j=1}^{J} \sum_{t=1}^{T} w_{jt} \tag{3.18}$$

Also note that the first expression above has a value for each value of the index j and so we can think of it as vector of length J. In contrast, the second expression is a single number since we have collapsed along both indices.

The next expression shows us how to compute the J credibility factors. Note that the formula is the same as in the balanced Bühlmann model with T replaced by the sum of the weights across time; that is, $w_{j\circ}$. Therefore, if all the weights are set equal to 1, then we have that $w_{j\circ} = T$ for all j and so this model becomes the balanced Bühlmann model and all J credibility factors, Z_j, are identical.

$$Z_j = \frac{\tau^2\, w_{j\circ}}{\tau^2\, w_{j\circ} + \sigma^2} = \frac{w_{j\circ}}{w_{j\circ} + \sigma^2/\tau^2} \quad \text{and} \quad Z_\circ = \sum_{j=1}^{J} Z_j \tag{3.19}$$

Finally, the experience, X_{jt}, can be first summarized by taking a weighted average along the time dimension leaving us with one average for each risk class. The second expression then takes the average of the J averages to compute an overall weighted average. The last expression is the weighted average of the individual risk class averages, but using the credibility factors as weights. Keep in mind that that the overall average $X_{\bullet\bullet}$ is, in general, not equal to credibility weighted average, X_z.

$$X_{j\bullet} = \sum_{t=1}^{T} \frac{w_{jt}}{w_{j\circ}} X_{jt} \quad \text{and} \quad X_{\bullet\bullet} = \sum_{j=1}^{J} \frac{w_{j\circ}}{w_{\circ\circ}} X_{j\bullet} \quad \text{and} \quad X_z = \sum_{j=1}^{J} \frac{Z_j}{Z_\circ} X_{j\bullet} \tag{3.20}$$

Theorem 3.2 (Bühlmann-Straub model)**.** *The mean squared error best homogeneous unbiased predictor $\sum_{it} h_{it} X_{it}$ of the risk premium $m + \Xi_j$ in model Equation 3.16, that is the solution to the following restricted minimization problem*

$$\min_{h_{it}} \mathbb{E}\left[\left(m + \Xi_j - \sum_{it} h_{it} X_{it}\right)^2\right] \tag{3.21}$$

$$\textit{subject to } \mathbb{E}[m + \Xi_j] = \sum_{it} h_{it} \mathbb{E}\left[X_{it}\right], \tag{3.22}$$

is the following credibility estimator:

$$Z_j X_{j\bullet} + (1 - Z_j) X_z \tag{3.23}$$

The above theorem has the same structure as the balanced Bühlmann theorem. Both results tell us that next periods' pure premium can be estimated as the weighted average of a risk class' average experience, $X_{j\bullet}$ and the overall average experience for the whole portfolio, X_z.

In the balanced Bühlmann model, the overall average experience is equal to the simple average across all observations; namely,

$$\bar{X} = \frac{\sum_{jt} X_{jt}}{J \cdot T}.$$

Whereas, in the Bühlmann-Straub model, the overall experience should be taken as the credibility weighted risk class experience, X_z; that is,

$$X_z = \sum_{j=1}^{J} \frac{Z_j}{Z_\circ} X_{j\bullet}.$$

Exercise

In the Bühlmann-Straub model set all weights, w_{jt}, equal to 1 and show that you reproduce the results for the balanced Bühlmann model.

Solution

By letting $w_{jt} = 1$ for all $j = 1, 2, \ldots, J$ and $t = 1, 2, \ldots, T$ we have that

$$w_{j\circ} = \sum_{t=1}^{T} w_{jt} = T, \quad \text{and} \quad X_{j\bullet} = \sum_{t=1}^{T} \frac{w_{jt}}{w_{j\circ}} X_{jt} = \frac{1}{T} \sum_{t=1}^{T} X_{jt},$$

and the credibility factors become

$$Z_j = \frac{\tau^2 \, w_{j\circ}}{\tau^2 \, w_{j\circ} + \sigma^2} = \frac{w_{j\circ}}{w_{j\circ} + \sigma^2/\tau^2} = \frac{T}{T + \sigma^2/\tau^2}$$

for all j. Therefore,

$$Z_\circ = \sum_{j=1}^{J} Z_j = \frac{JT}{T + \sigma^2/\tau^2}.$$

Finally, substituting all the different pieces into X_z we get

$$X_z = \sum_{j=1}^{J} \frac{Z_j}{Z_\circ} X_{j\bullet} = \sum_{j=1}^{J} \frac{\frac{T}{T+\sigma^2/\tau^2}}{\frac{JT}{T+\sigma^2/\tau^2}} X_{j\bullet} = \sum_{j=1}^{J} \frac{1}{J} \left(\frac{1}{T} \sum_{t=1}^{T} X_{jt} \right) = \frac{1}{JT} \sum_{j=1}^{J} \sum_{t=1}^{T} X_{jt} = \bar{X},$$

showing that when the weights in the Bühlmann-Straub model are all equal to 1 we revert back to the balanced Bühlmann model.

For the Bühlmann-Straub model we also need estimators for the model parameters m, σ^2, and τ^2. These estimators are also based on the *sum of squared errors within* and *sum of squared errors between* we have seen before, but incorporating the weights for each observation; namely,

$$\text{SSE}_W = \sum_{jt} w_{jt} (X_{jt} - X_{j\bullet})^2,$$

and

$$\text{SSE}_B = \sum_{jt} w_{j\circ}(X_{j\bullet} - X_{\bullet\bullet})^2.$$

Figure 3.4 shows one way to think about the between and within risk variances. The blue circles represent the actual observations, X_{jt}, we have available. The open circles at the bottom labeled $X_{1\bullet}, X_{2\bullet}$, and $X_{3\bullet}$ represent an estimate of the hypothetical means for groups 1, 2, and 3. For each group, we have the deviations, shows as blue arrows, between the estimated hypothetical mean, $X_{j\bullet}$, and its actual observations, X_{jt}. The magnitude of those differences squared result in the *within* sum of squared errors, SSE_W.

From the estimated hypothetical means, $X_{j\bullet}$, we compute an overall mean shown as a filled red circle, $X_{\bullet\bullet}$. This represents the collective average for the entire portfolio of risks. The square of the deviations between the overall mean and the hypothetical means, shown in the upper section of Figure 3.4, form the *between* sum of squared errors, SSE_B.

```
\begin{tikzpicture}
  \draw[->] (-5.6, 0)--(5.6, 0);
  \draw[fill,blue!10!white] (-5.2, 0.5) rectangle (5.6, 3.05);
  \draw[fill,red!10!white] (-5.2, 3.75) rectangle (5.6, 5.25);

  \pgfsetshortenend{4pt};
  \draw (-3.2, 0) circle[radius=5pt];
  \draw[gray] (-3.2, 0)--(-3.2, 6);
  \draw[->,blue] (-3.2, 0.75)--(-4, 0.75);
  \draw[blue] (-4, 0.75) circle[radius=3pt];
  \draw[->,blue] (-3.2, 1.25)--(-2.8, 1.25);
  \draw[blue] (-2.8, 1.25) circle[radius=3pt];
  \draw[->,blue] (-3.2, 1.75)--(-3.6, 1.75);
  \draw[blue] (-3.6, 1.75) circle[radius=3pt];
  \draw[->,blue] (-3.2, 2.25)--(-4.8, 2.25);
  \draw[blue] (-4.8, 2.25) circle[radius=3pt];
  \draw[->,blue] (-3.2, 2.75)--(-2, 2.75);
  \draw[blue] (-2, 2.75) circle[radius=3pt];

  \draw (1.6, 0) circle[radius=5pt];
  \draw[gray] (1.6, 0)--(1.6, 6);
  \draw[->,blue] (1.6, 0.75)--(2.4, 0.75);
  \draw[blue] (2.4, 0.75) circle[radius=3pt];
  \draw[->,blue] (1.6, 1.25)--(2.8, 1.25);
  \draw[blue] (2.8, 1.25) circle[radius=3pt];
  \draw[->,blue] (1.6, 1.75)--(-0.4, 1.75);
  \draw[blue] (-0.4, 1.75) circle[radius=3pt];
  \draw[->,blue] (1.6, 2.25)--(1.2, 2.25);
```

```
\draw[blue] (1.2, 2.25) circle[radius=3pt];
\draw[->,blue] (1.6, 2.75)--(2, 2.75);
\draw[blue] (2, 2.75) circle[radius=3pt];

\draw (4, 0) circle[radius=5pt];
\draw[gray] (4, 0)--(4, 6);
\draw[->,blue] (4, 1.25)--(3.2, 1.25);
\draw[blue] (3.2, 1.25) circle[radius=3pt];
\draw[->,blue] (4, 1.75)--(5.2, 1.75);
\draw[blue] (5.2, 1.75) circle[radius=3pt];
\draw[->,blue] (4, 2.25)--(4.8, 2.25);
\draw[blue] (4.8, 2.25) circle[radius=3pt];
\draw[->,blue] (4, 2.75)--(2.8, 2.75);
\draw[blue] (2.8, 2.75) circle[radius=3pt];

\pgfsetshortenend{0pt};
\draw[gray] (0, 0)--(0, 6);
\draw[->,red] (0, 4)--(-3.2, 4);
\draw[->,red] (0, 4.5)--(1.6, 4.5);
\draw[->,red] (0, 5)--(4, 5);
\draw[fill,red] (0, 0) circle[radius=5pt];

\pgfsetshortenend{6pt};
\pgfsetshortenstart{6pt};
\node at (0, -1) {$X_{\bullet\bullet}$};
\draw[->] (0, -1)--(0, 0);
\node at (-3.2, -1) {$X_{3\bullet}$};
\draw[->] (-3.2, -1)--(-3.2, 0);
\node at (1.6, -1) {$X_{1\bullet}$};
\draw[->] (1.6, -1)--(1.6, 0);
\node at (4, -1) {$X_{2\bullet}$};
\draw[->] (4, -1)--(4, 0);

\pgfsetshortenend{4pt};
\pgfsetshortenstart{8pt};
\node at (-1.6, 1.75) {$X_{jt}$};
\draw[->] (-1.6, 1.75)--(-2.8, 1.25);
\draw[->] (-1.6, 1.75)--(-0.4, 1.75);
\draw[->] (-1.6, 1.75)--(-2, 2.75);

\node[align=center] at (-6.6, 1.75) {Within risk\\ deviations};
\node[align=center] at (-6.6, 4.5) {Between risk\\ deviations};
```

```
\end{tikzpicture}
```

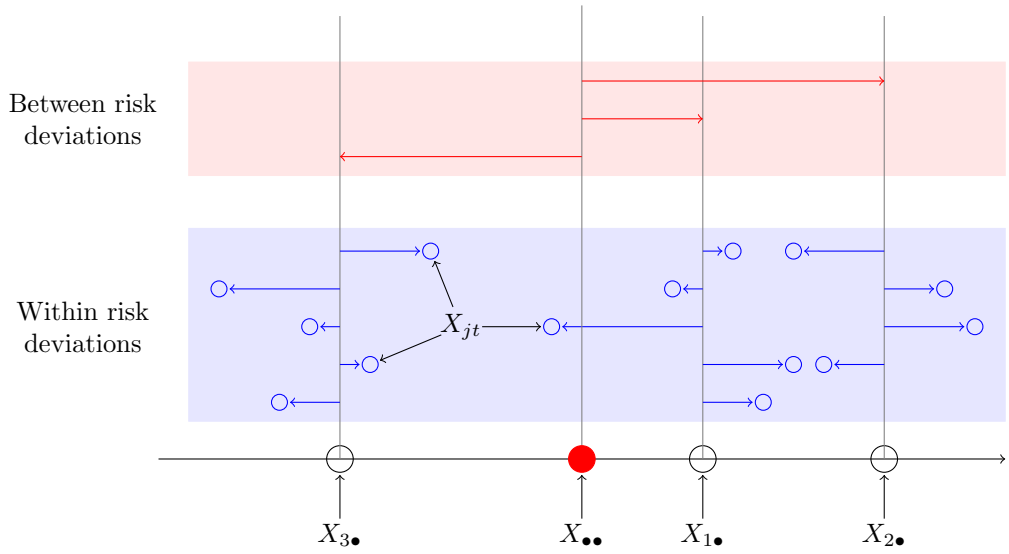


Figure 3.4: Graphical representation of the within risk and between risk deviations. The small open circles represent the actual observations we have available. The open circles at the bottom are the estimated hypothetical means and the filled circle is the overall average. The blue arrows represent the within risk deviations and the red arrows are the between risk deviations.

The following theorem (Kaas 2009, Theorem 8.4.2, p. 218) tells us how to calculate unbiased estimators for m, σ^2, and τ^2.

Theorem 3.3 (Unbiased parameter estimates)**.** *In the Bühlmann-Straub model, the following statistics are unbiased estimators of the corresponding model parameters:*

$$\hat{m} = X_{\bullet\bullet} \tag{3.24}$$

$$\hat{\sigma}^2 = \frac{1}{J(T-1)} \sum_{jt} w_{jt}(X_{jt} - X_{j\bullet})^2 \tag{3.25}$$

$$\hat{\tau}^2 = \frac{\sum_j w_{j\circ}(X_{j\bullet} - X_{\bullet\bullet})^2 - (J-1)\hat{\sigma}^2}{w_{\circ\circ} - \sum_j w_{j\circ}^2 / w_{\circ\circ}} \tag{3.26}$$

Note that the estimator for τ^2 is the difference between two expressions and so it is possible that in applying the model the computed value will be negative. In this case, most practitioners will set the value of this parameter to zero.

To illustrate the Bühlmann-Straub model we will generate a synthetic portfolio and compute predictions for the next period. Our discussion follows the development in (Kaas 2009, example 8.4.5 p. 220).

Let's set up our portfolio with $J = 100$ risk classes and $T = 5$ years of observations that follow the model in Equation 3.16 with the following parameters: $m = 80$, $\tau^2 = 64$, and $\sigma^2 = 100$.

```
J <- 100; Tm <- 5
j <- as.factor(rep(1:J, each = Tm))

m <- 80
t2 <- 64
s2 <- 100
```

We will set up weights ranging from 0.5 to 1.5 and simulate the observations $X_{jt} = m + \Xi_j + \epsilon_{jt}$ with both Ξ_j and ϵ_{jt} as normal random variables with mean zero and variance τ^2 and σ^2/w_{jt}, respectively.

```
set.seed(12094851)
w.jt <- 0.5 + runif(J * Tm)
X.jt <- m + rep(rnorm(J, 0, sqrt(t2)), each = Tm) +
            rnorm(J * Tm, 0, sqrt(s2/w.jt))
```

```
dta <- tibble(risk = j,
              X.jt = X.jt,
              W.jt = w.jt)
write_csv(dta, "BS-simulated-data.csv")
rm(dta)
```

Before embarking on credibility calculations we should check if our portfolio is homogeneous or not. If it is homogeneous, then we do not need to apply credibility and we could estimate next years' experience by the overall weighted average. To check we will do an analysis of variance.

```
(av <- anova(lm(X.jt ~ j, weights = w.jt)))
```

```
Analysis of Variance Table

Response: X.jt
           Df Sum Sq Mean Sq F value    Pr(>F)
j          99  40395  408.03     3.9 < 2.2e-16 ***
```

```
Residuals 400  41850  104.62
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the above output we can immediately tell that our portfolio is heterogeneous; the risk classes differ from each other. The F-value statistic is too large given the 99 and 400 degrees of freedom (the 95% critical value would be 1.2842).

Also from the analysis of variance output we can immediately see that the estimated value of σ^2 is equal to

```
(s2.hat <- av[2,3])
```

```
[1] 104.6239
```

The estimator for the overall mean m is the overall weighted average for the data.

```
(m.hat <- X.bb <- sum(w.jt * X.jt) / sum(w.jt))
```

```
[1] 78.43628
```

Finally, we need to implement the calculation for the estimator of τ^2. Start with some preliminary setup where the sum of the weights across time, $w_{j\circ}$, corresponds to `w.jc`, the sum of all weights $w_{\circ\circ}$ is `w.cc`, and the weighted average of the experience across time for each risk class $X_{j\bullet}$ is given by `X.jb`.

```
w.jc <- tapply(w.jt, j, sum)
w.cc <- sum(w.jc)
X.jb <- tapply(w.jt * X.jt / w.jc[j], j, sum)
```

Hence, the estimator for τ^2 is given by

```
num <- sum(w.jc * (X.jb - X.bb)^2) - (J - 1) * s2.hat
den <- w.cc - sum(w.jc^2 / w.cc)
(t2.hat <- num / den)
```

```
[1] 60.9652
```

Finally, the credibility factors are

```
Zj.hat <- w.jc / (w.jc + s2.hat / t2.hat)
```

and the collective premium is

```
(X.z <- sum(Zj.hat * X.jb) / sum(Zj.hat))
```

```
[1] 78.53833
```

and putting them together we obtain the following credibility premiums

```
P.hat <- Zj.hat * X.jb + (1 - Zj.hat) * X.z
```

The first 20 estimated credibility premiums are

```
P.hat[1:20]
```

```
       1        2        3        4        5        6        7        8
67.41990 74.65462 82.37383 81.77987 75.53623 84.26384 80.16259 71.72899
       9       10       11       12       13       14       15       16
95.02067 71.41841 86.42743 80.35881 90.36646 73.34605 74.71319 78.70976
      17       18       19       20
71.75083 92.52850 72.31943 69.29793
```

And as we did for the Bühlmann model we can use the function `cm()` from package `actuar` to do the calculations necessary for the Bühlmann-Straub model. We first create a data frame with both the observations, X_{jt} and the weights w_{jt} along with a column telling us which row of data belongs to which risk class.

First we create a data frame with the data we have

```
D <- cbind(risk.class = 1:J,
           as.data.frame(matrix(X.jt,
                                nrow = J,
                                ncol = Tm,
                                byrow = TRUE)),
           as.data.frame(matrix(w.jt,
                                nrow = J,
                                ncol = Tm,
                                byrow = TRUE)))
```

and then we estimate the model via

```
(BS <- cm(~ risk.class,
          data = D,
          ratios = 2:6,
          weights = 7:11))
```

```
Call:
cm(formula = ~risk.class, data = D, ratios = 2:6, weights = 7:11)

Structure Parameters Estimators

  Collective premium: 78.53833

  Between risk.class variance: 60.9652
  Within risk.class variance: 104.6239
```

For the first 5 risk classes, our by-hand calculations match those from the `cm()` function.

```
rbind("   cm():" = predict(BS)[1:5],
      "by-hand:" = P.hat[1:5])
```

```
                1        2        3        4        5
   cm(): 67.4199 74.65462 82.37383 81.77987 75.53623
by-hand: 67.4199 74.65462 82.37383 81.77987 75.53623
```

And they match across all risk classes:

```
all(round(abs(predict(BS) - P.hat), 10) == 0)
```

```
[1] TRUE
```

3.5 Hachemeister Regression

In this chapter we have been working with the following basic model

$$X_{jt} = m + \Xi_j + \epsilon_{jt},$$

where j indexes a risk class and t represents time. Both Ξ_j and ϵ_{jt} are random variables and m is a fixed parameter.

There are many ways in which this basic model can be extended. For example, one may consider a tree-like structure of an insurance line of business where at the top level we have the entire business. This can be divided into different sectors and then each sector can again be divided into risk classes. Finally, each risk class has the observed experience. This model is known as Jewell's hierarchical model (Jewell 1975) and the statistical model can be written as

$$X_{sjt} = m + \Xi_s + \Xi_{sj} + \epsilon_{sjt},$$

where $s = 1, 2, \ldots, S$ represents the different sectors, $j = 1, 2, \ldots, J$ corresponds to the different risk classes, and $t = 1, 2, \ldots, T + 1$ indexes the time periods. Extensions to more levels follows the same pattern.

In this case, Ξ_s is the deviation from the overall mean m for sector s, Ξ_{sj} is then the deviation from the sector mean for risk class j, and ϵ_{sjt} represents the fluctuations through time.

In this section, we are interested in a different extension of the basic model that will take us in the direction of the classical linear regression model. This model, known by actuaries as a credibility regression model, was first introduced into the actuarial literature in (Hachemeister 1975).

Due to the high inflation rates starting in the late 1960's and continuing into the early 1970's, Hachemeister was interested in understanding loss severity trends and used data from private passenger automobile (bodily injury coverage) for five different states to illustrate his ideas. The data is on a quarterly basis from 3Q 1970 through 2Q 1973 (12 quarters of observations).

Table 3.2 shows the claim severity and the number of claims by state. Note that state 1 has a very large number of claims in each quarter and state 4 has the least number of claims of all states. The ratio of the number of claims, in each quarter, for state 1 to state 4 is almost always in excess of 20. Figure 3.5 displays a multiple time series plot for severity. The data for each state has been been connected with a light gray line and each state has been labeled on the right-hand side. State 4 has been highlighted, with thick connecting segments, to illustrate the data for a single state.

```
data("hachemeister", package = "actuar")
dta <- hachemeister
rm(hachemeister)
```

```
time <- c("3Q'70", "4Q'70",
          paste(1:4, "Q'71", sep = ""),
          paste(1:4, "Q'72", sep = ""),
          "1Q'73", "2Q'73")
tb <- cbind(time, as.data.frame(t(dta[,2:13])),
            as.data.frame(t(dta[,14:25])))
kbl(tb,
    digits = 0,
```

Table 3.2: The Hachemeister data. Number of claims and severity for five different states from private passenger automobile insurance (bodily injury coverage).

	Claim Severity by State					Number of Claims by State				
Period	1	2	3	4	5	1	2	3	4	5
3Q'70	1,738	1,364	1,759	1,223	1,456	7,861	1,622	1,147	407	2,902
4Q'70	1,642	1,408	1,685	1,146	1,499	9,251	1,742	1,357	396	3,172
1Q'71	1,794	1,597	1,479	1,010	1,609	8,706	1,523	1,329	348	3,046
2Q'71	2,051	1,444	1,763	1,257	1,741	8,575	1,515	1,204	341	3,068
3Q'71	2,079	1,342	1,674	1,426	1,482	7,917	1,622	998	315	2,693
4Q'71	2,234	1,675	2,103	1,532	1,572	8,263	1,602	1,077	328	2,910
1Q'72	2,032	1,470	1,502	1,953	1,606	9,456	1,964	1,277	352	3,275
2Q'72	2,035	1,448	1,622	1,123	1,735	8,003	1,515	1,218	331	2,697
3Q'72	2,115	1,464	1,828	1,343	1,607	7,365	1,527	896	287	2,663
4Q'72	2,262	1,831	2,155	1,243	1,573	7,832	1,748	1,003	384	3,017
1Q'73	2,267	1,612	2,233	1,762	1,613	7,849	1,654	1,108	321	3,242
2Q'73	2,517	1,471	2,059	1,306	1,690	9,077	1,861	1,121	342	3,425

```
    row.names = FALSE,
    col.names = c("Period", rep(1:5, 2)),
    align = "lrrrrrrrrrr",
    format.args = list(big.mark = ','),
    booktabs = TRUE) %>%
  kable_styling(full_width = FALSE) %>%
  add_header_above(c(" ", "Claim Severity by State" = 5,
                     "Number of Claims by State" = 5))
rm(time, tb)
```

```
db.1 <- pivot_longer(as.data.frame(dta[,1:13]),
                     cols = 2:13,
                     names_to = "time",
                     names_prefix = "ratio.",
                     values_to = "severity")
db.2<- pivot_longer(as.data.frame(dta[,c(1,14:25)]),
                    cols = 2:13,
                    names_to = "time",
                    names_prefix = "weight.",
                    values_to = "claims")
db <- inner_join(db.1, db.2, by = c("state", "time"))
```

```
db$state <- as.character(db$state)
db$time <- as.numeric(db$time)
write_csv(db, "hachemeister-data.csv")
rm(db.1, db.2)
```

```
A <- filter(db, state %in% c(1:3,5))
B <- filter(db, state %in% 4)
C <- cbind(state = "ALL",
           summarize(group_by(db, time),
                     clms = sum(claims),
                     loss = sum(claims * severity),
                     sev = loss / clms))
p <- ggplot(data = A,
       mapping = aes(x = time,
                     y = severity,
                     group = state)) +
  geom_line(color = "gray") +
  geom_point(color = "gray") +
  labs(x = "Time",
       y = "Severity") +
  scale_x_continuous(breaks = 1:12,
                     labels = c(paste(3:4, "Q'70", sep = ""),
                                paste(1:4, "Q'71", sep = ""),
                                paste(1:4, "Q'72", sep = ""),
                                paste(1:2, "Q'73", sep = "")),
                     minor_breaks = NULL)
p <- p + geom_line(data = B,
                   mapping = aes(x = time,
                                 y = severity),
                   col = "gray", linewidth = 1.25) +
  geom_point(data = B,
             mapping = aes(x = time,
                           y = severity),
             color = "darkgray") +
  annotate("text",
           x = rep(12.3, 5),
           y = db$severity[db$time == 12],
           label = paste("State", 1:5, sep = " "),
           size = 3)
p
```

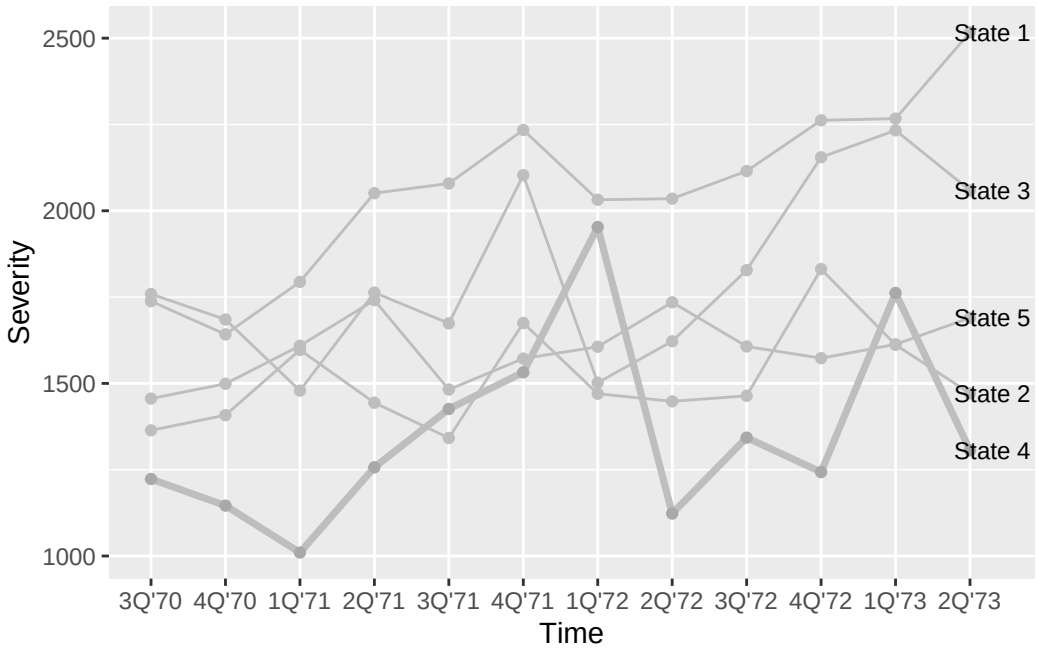


Figure 3.5: Hachemeister data showing quarterly experience for private passenger auto (bodily injury coverage) severity for five different states from the third quarter of 1970 to the second quarter of 1973 (12 observations). The observations for each state have been connected with light gray lines to emphasize the state individual trends. State 4 has been highlighted with thicker line segments.

Notice that the variability in severity from quarter to quarter for state 4 is much greater than for other states and state 1 appears to be the most stable. This feature arises because the volume of claims underlying the severity is quite different between state 1 and state 4.

The severity trend for each state is positive and we could measure its magnitude by fitting a weighted least squares regression line to each state. Doing so yields the five light purple straight lines shown in Figure 3.6. We have also included a 'countrywide' (data combined for all five states) weighted regression line shown in red.

```
q <- p + geom_smooth(data = A,
                     method = "lm",
                     mapping = aes(x = time,
                                   y = severity,
                                   weight = claims),
                     se = FALSE,
                     color = "#f1b6da",
                     linewidth = 0.5) +
  geom_smooth(data = B,
```

```
                method = "lm",
                mapping = aes(x = time,
                              y = severity,
                              weight = claims),
                se = FALSE,
                color = "#f1b6da",
                linewidth = 0.5) +
  geom_smooth(data = C,
                method = "lm",
                mapping = aes(x = time,
                              y = sev,
                              weight = clms),
                se = FALSE,
                color = "red",
                linewidth = 0.5)
q
rm(A, B, C, p, q)
```

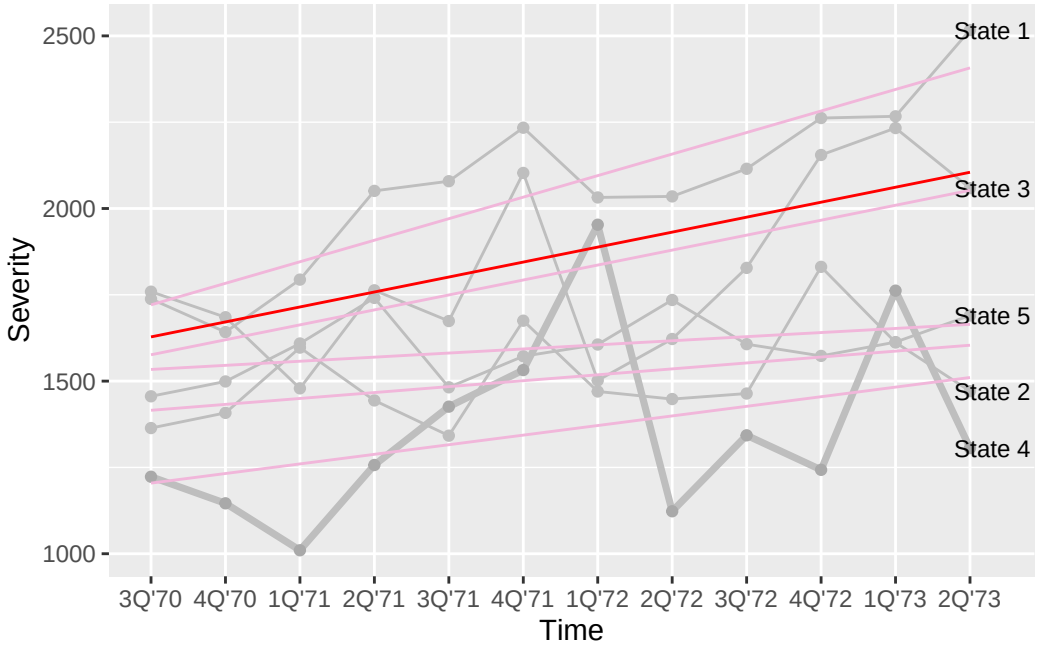


Figure 3.6: Hachemeister data including individual regression lines for each state (shown in light purple) and a 'countrywide' regression line (shown in red) for the combined data of all five states.

Looking at the trend lines for each state shown in Figure 3.6 we can see that each one has a

different level of severity and each one has a different positive slope. State 1 has the largest slope and state 5 the smallest.

The basic credibility model that we have been working with cannot accommodate this setup, but we can extend it as follows:

$$X_{jt} = \left\{m^{(1)} + \Xi_j^{(1)}\right\} + \left\{m^{(2)} + \Xi_j^{(2)}\right\} y_{jt} + \epsilon_{jt}, \tag{3.27}$$

where $m^{(1)}$ and $m^{(2)}$ are unknown fixed parameters, $\Xi_j^{(1)}$ and $\Xi_j^{(2)}$ are random variables with mean zero, y_{jt} is, in this particular case, the explanatory variable time, but in general could be any explanatory variable.

If the random variables $\Xi_j^{(1)}$ and $\Xi_j^{(2)}$ are identically zero (that is, their variance is zero), then the model becomes a classical regression. If only $\Xi_j^{(2)}$ is identically zero, then we have a random intercept model. In our example, every state would have their own intercept, but all states would share the same slope. That is, we would have parallel regression lines. We can estimate such a model via least squares by including an indicator variable for each state (and omitting the intercept). If only $\Xi_j^{(1)}$ is identically zero, then all states have the same intercept, but each one has a different slope. This would be a random slope model.

The regression lines shown in light purple in Figure 3.6 have been estimated using **only** the information contained in each state. If we feel that the information in each state is fully credible, then these estimated regression lines are appropriate. If, on the other hand, the information in each state is not credible, then we would ignore the `state` grouping variable by using all the data together to estimate the collective regression line.

As we have seen with the Bühlmann and the Bühlmann-Straub models these are two extreme positions and we can search for an in-between compromise. So Hachemeister set out to develop a credibility regression model. We will follow the discussion in (Hans Bühlmann and Gisler 2005, chap. 8) closely and focus on the example presented above of linear regression (intercept and slope) even though the ideas clearly apply to more general regression models. Also, we will change our notation slightly and use vectors and matrices to make the connection to linear regression more apparent. To this end, we can restate Equation 3.27 as follows

$$X_j = Y_j\, \beta(\Theta_j) + \epsilon_j, \tag{3.28}$$

where X_j is a $N \times 1$ column vector of responses for risk $j = 1, 2, \ldots, J$ and N is the number of time periods we have observed. In Hachemeister example, this corresponds to the observed severities for state $j = 1, 2, \ldots, 5$ during time periods $t = 1, 2, \ldots, 12$ so we have

$$X_j = \begin{bmatrix} X_{j1} \\ X_{j2} \\ \vdots \\ X_{jN} \end{bmatrix}.$$

The matrix Y_j is our regression design matrix for state j with dimensions $T \times 2$,

$$Y_j = \begin{bmatrix} 1 & t_{j1} \\ 1 & t_{j2} \\ \vdots & \vdots \\ 1 & t_{jN} \end{bmatrix}.$$

The first column of Y_j corresponds to the intercept and the second column is the time variable. For our example, we have $t_{j1} = 1$, $t_{j2} = 2, ...$, for all states j. The 2×1 vector $\beta(\Theta_j)$ is our regression coefficient. The reason for having the β's depend on Θ_j is that we are thinking of these regression coefficients as dependent on a random variable for each state. Thus we have,

$$\beta(\Theta_j) = \begin{bmatrix} \beta_0(\Theta_j) \\ \beta_1(\Theta_j) \end{bmatrix}$$

where $\beta_0(\Theta_j)$ is our intercept and $\beta_1(\Theta_j)$ is our slope for state j. Finally, the $N \times 1$ vector ϵ_j of error terms is

$$\epsilon_j = \begin{bmatrix} \epsilon_{j1} \\ \epsilon_{j2} \\ \vdots \\ \epsilon_{jN} \end{bmatrix}.$$

Definition 3.1 (Hachemeister Model Assumptions)**.** The risk j is characterized by an individual risk profile ϑ_j, which is itself the realization of a random variable Θ_j. We make the following assumptions:

Conditionally, given Θ_j, the entries X_{jt}, $j = 1, \dots, J$ are independent and we have

$$\mathbb{E}[X_j|\Theta_j] = Y_j\, \beta(\Theta_j),$$

where $\beta(\Theta_j)$ is the regression vector and Y_j is the known design matrix, and

$$\mathrm{Cov}[X_j, X_j'|\Theta_j] = \Sigma_j(\Theta_j)$$

is the covariance matrix conditional on Θ_j. The pairs (Θ_1, X_1), $(\Theta_2, X_2), ...$ are independent, and also $\Theta_1, \Theta_2, ...$ are independent and identically distributed.

Theorem 3.4 (Hachemeister formula)**.** *Under the Hachemeister model assumptions we get that the credibility estimator for $\beta(\Theta_j)$ satisfies*

$$\widehat{\beta(\Theta_j)} = A_j B_j + (I - A_j)\beta, \tag{3.29}$$

where

$$
\begin{aligned}
A_j &= T\left(T + (Y_j' S_j^{-1} Y_j)^{-1}\right)^{-1}, \\
B_j &= (Y_j' S_j^{-1} Y_j)^{-1} Y_j' S_j^{-1} X_j \\
S_j &= \mathbb{E}[\Sigma_j(\Theta_j)] = \mathbb{E}[Cov[X_j, X_j'|\Theta_j]], \\
T &= Cov[\beta(\Theta_j), \beta(\Theta_j)'], \\
\beta &= \mathbb{E}[\beta(\Theta_j)].
\end{aligned}
$$

The credibility matrices, A_j, are for the example we are considering of dimension 2×2. The 2×1 vector B_j is the intercept and slope for each state j. The matrices S_j have dimension $N \times N$ and are of the form

$$
S_j = \sigma^2 \begin{bmatrix} w_{j1} & 0 & \cdots & 0 \\ 0 & w_{j2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & w_{jN} \end{bmatrix}^{-1},
$$

where the diagonal entries w_{jt} are known weights. We will use W_j as the diagonal matrix with entries w_{jt} along the main diagonal and zeroes everywhere else. Thus we can write $S_j = \sigma^2 W_j^{-1}$ In our example, the w_{jt} are the number of claims at time t in state j.

The matrix T, of dimension 2×2, is the variance-covariance matrix of the estimated coefficients. Since the matrix T is symmetric there are at most 3 distinct entries. And finally, the 2×1 vector β is the collective intercept and slope.

Let us apply Hachemeister formula (Theorem 3.4) to the data we have on hand. In several places we will need to calculate the product $Y_j' S_j^{-1} Y_j$ with S_j being the diagonal matrix in the previous paragraph. This product is closely related to $Y_j' W_j Y_j$ which comes up several times, so let us define

$$
V_j = Y_j' W_j Y_j,
$$

which is a 2×2 matrix. Specializing for the Hachemeister data we have

$$
V_j = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ 1 & 2 & \ldots & N \end{bmatrix} \begin{bmatrix} w_{j1} & 0 & \ldots & 0 \\ 0 & w_{j2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & w_{jN} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & N \end{bmatrix}.
$$

Doing the matrix multiplications gives us the following

$$
V_j = \begin{bmatrix} \sum_{t=1}^N w_{jt} & \sum_{t=1}^N t w_{jt} \\ \sum_{t=1}^N t w_{jt} & \sum_{t=1}^N t^2 w_{jt} \end{bmatrix}
$$

The entries in the matrix V_j almost look like weighted averages. They are missing a denominator equal to $\sum_{t=1}^N w_{jt}$. To keep the notation cleaner we will use the following convention:

a '$\bullet$' symbol in an index position means that we sum all entries along that index. Hence, we have $w_{j\bullet} = \sum_{t=1}^{N} w_{jt}$. Therefore, we can rewrite the above matrix V_j as

$$V_j = w_{j\bullet} \begin{bmatrix} \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} & \sum_{t=1}^{N} t\frac{w_{jt}}{w_{j\bullet}} \\ \sum_{t=1}^{N} t\frac{w_{jt}}{w_{j\bullet}} & \sum_{t=1}^{N} t^2\frac{w_{jt}}{w_{j\bullet}} \end{bmatrix}.$$

To simplify the notation further, note that the off-diagonal entries look like the calculation of the expected value of t because the weights $w_{jt}/w_{j\bullet}$ sum to 1 and so, we are thinking of them as a sampling distribution. Similarly the entry in position $2,2$ looks like the calculation of the second moment.

In view of this, define the following notation

$$E_j^{(s)}[t] = \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} t \qquad \text{and} \qquad E_j^{(s)}[X_j] = \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} X_{jt},$$

where the superscript (s) signals that we are thinking of the weights $w_{jt}/w_{j\bullet}$ as a sampling distribution. With this notation we can also write

$$\mathrm{Var}_j^{(s)}[t] = E_j^{(s)}[t^2] - \left(E_j^{(s)}[t]\right)^2.$$

Using all of this, the matrix V_j is now

$$V_j = w_{j\bullet} \begin{bmatrix} 1 & E_j^{(s)}[t] \\ E_j^{(s)}[t] & E_j^{(s)}[t^2] \end{bmatrix}$$

and note that its determinant is equal to $\det(V_j) = w_{j\bullet}^2 \mathrm{Var}^{(s)}[t]$.

Using Theorem 3.4 we can calculate B_j as follows:

$$\begin{aligned} B_j &= V_j^{-1} Y_j' W_j X_j \\ &= \frac{1}{w_{j\bullet}\mathrm{Var}^{(s)}[t]} \begin{bmatrix} E_j^{(s)}[t^2] & -E_j^{(s)}[t] \\ -E_j^{(s)}[t] & 1 \end{bmatrix} \begin{bmatrix} \sum_{t=1}^{N} w_{jt} X_{jt} \\ \sum_{t=1}^{N} w_{jt} t X_{jt} \end{bmatrix} \\ &= \frac{1}{\mathrm{Var}^{(s)}[t]} \begin{bmatrix} E_j^{(s)}[t^2] & -E_j^{(s)}[t] \\ -E_j^{(s)}[t] & 1 \end{bmatrix} \begin{bmatrix} \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} X_{jt} \\ \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} t X_{jt} \end{bmatrix} \\ &= \frac{1}{\mathrm{Var}^{(s)}[t]} \begin{bmatrix} E_j^{(s)}[t^2] & -E_j^{(s)}[t] \\ -E_j^{(s)}[t] & 1 \end{bmatrix} \begin{bmatrix} E_j^{(s)}[X_{jt}] \\ E_j^{(s)}[tX_{jt}] \end{bmatrix} \\ &= \frac{1}{\mathrm{Var}^{(s)}[t]} \begin{bmatrix} E_j^{(s)}[t^2]E_j^{(s)}[X_{jt}] - E_j^{(s)}[t]E_j^{(s)}[tX_{jt}] \\ E_j^{(s)}[tX_{jt}] - E_j^{(s)}[t]E_j^{(s)}[X_{jt}] \end{bmatrix} \end{aligned}$$

Let us implement these calculations using Hachemeister's data. First lets define some quantities: the weights `W.jt`, the time points `T.jt`, the observed severities `X.jt` and the vector `S` which tells us which state these observations belong to.

```
W.jt <- db$claims
T.jt <- db$time
X.jt <- db$severity
S <- db$state
N <- length(unique(T.jt))
J <- length(unique(S))
```

Next we calculate the summary statistics that we will need. The following table shows the correspondence between our programming variable names and our written notation.

Variable Name	Written Notation
`W.jb`	$w_{j\bullet}$
`W.bb`	$\sum_{j=1}^{J} w_{j\bullet}$
`Ej.t`	$E_j^{(s)}[t]$
`Ej.t2`	$E_j^{(s)}[t^2]$
`Ej.X`	$E_j^{(s)}[X_j]$
`Ej.tX`	$E_j^{(s)}[tX_j]$
`Vj.t`	$\mathrm{Var}_j^{(s)}[t]$
`Ws.jb`	$\mathrm{Var}_j^{(s)}[t]w_{j\bullet}$
`Ws.bb`	$\sum_{j=1}^{J} \mathrm{Var}_j^{(s)}[t]w_{j\bullet}$

```
W.jb <- tapply(W.jt, S, sum)
W.bb <- sum(W.jb)
Ej.t <- tapply(W.jt * T.jt, S, sum) / W.jb
Ej.t2 <- tapply(W.jt * T.jt^2, S, sum) / W.jb
Ej.X <- tapply(W.jt * X.jt, S, sum) / W.jb
Ej.tX <- tapply(W.jt * T.jt * X.jt, S, sum) / W.jb
Vj.t <- Ej.t2 - Ej.t^2
Ws.jb <- Vj.t * W.jb
Ws.bb <- sum(Ws.jb)
```

With these definitions we can calculate the intercept and slope for each state using

```
Bj <- rbind((Ej.t2 * Ej.X - Ej.t * Ej.tX) / Vj.t,
            (Ej.tX - Ej.t * Ej.X) / Vj.t)
dimnames(Bj) <- list(c("Intercept", "Slope"),
                     1:5)
round(Bj, 2)
```

```
                1       2       3       4       5
Intercept 1658.47 1398.30 1533.00 1176.70 1521.90
Slope       62.39   17.14   43.31   27.81   11.87
```

Exercise

Verify that the intercept and slope we calculated for each state are correct by doing it the easy way; that is, fit a weighted linear regression to the data for each state.

Solution

For state 4 we would compute as follows:

```
lm.st4 <- lm(severity ~ time,
             data = db,
             subset = state == 4,
             weights = claims)
coef(lm.st4)
```

```
(Intercept)        time
 1176.70407    27.80702
```

And indeed these coefficients match those we computed earlier. We leave similar calculations, for the remaining states, to the reader.

Next in (Hans Bühlmann and Gisler 2005, 209) assume that the 2×2 matrix $T = \text{Cov}[\beta(\Theta_j), \beta(\Theta_j)']$ is diagonal with entries τ_0^2 and τ_1^2; that is,

$$T = \begin{bmatrix} \tau_0^2 & 0 \\ 0 & \tau_1^2 \end{bmatrix}.$$

This assumes that the intercept and the slope are independent of each other.

The credibility matrices A_j are given in Theorem 3.4 as

$$A_j = T\left(T + (Y_j' S_j^{-1} Y_j)^{-1}\right)^{-1}$$

and we can re-write, by using V_j, as follows

$$A_j = T\left(T + \sigma^2 V_j^{-1}\right)^{-1}.$$

We could substitute expressions for T and V_j and compute, but this requires a lot of calculations. It is best to re-write as follows

$$\begin{aligned} A_j &= T\left(T + \sigma^2 V_j^{-1}\right)^{-1} \\ &= \left(I + \sigma^2 V_j^{-1} T^{-1}\right)^{-1} \\ &= \left(V_j + \sigma^2 T^{-1}\right)^{-1} V_j. \end{aligned}$$

The inverse of T is easy because T is a 2×2 diagonal matrix. We have

$$T^{-1} = \frac{1}{\tau_0^2 \tau_1^2} \begin{bmatrix} \tau_1^2 & 0 \\ 0 & \tau_0^2 \end{bmatrix}$$

and noting that we need to multiply by σ^2, we can make the following substitutions. Let $\kappa_0 = \sigma^2/\tau_0^2$ and $\kappa_1 = \sigma^2/\tau_1^2$, then we have

$$T^{-1} = \frac{1}{\sigma^2} \begin{bmatrix} \kappa_0 & 0 \\ 0 & \kappa_1 \end{bmatrix}$$

and so

$$V_j + \sigma^2 T^{-1} = \begin{bmatrix} w_{j\bullet} + \kappa_0 & w_{j\bullet} E_j^{(s)}[t] \\ w_{j\bullet} E_j^{(s)}[t] & w_{j\bullet} E_j^{(s)}[t^2] + \kappa_1 \end{bmatrix}.$$

The inverse of the above matrix is

$$\frac{1}{D} \begin{bmatrix} w_{j\bullet} E_j^{(s)}[t^2] + \kappa_1 & -w_{j\bullet} E_j^{(s)}[t] \\ -w_{j\bullet} E_j^{(s)}[t] & w_{j\bullet} + \kappa_0 \end{bmatrix},$$

where D is the determinant given by

$$D = (w_{j\bullet} + \kappa_0)\left(w_{j\bullet} E_j^{(s)}[t^2] + \kappa_1\right) - \left(w_{j\bullet} E_j^{(s)}[t]\right)^2.$$

Finally, multiplying the above expression by T and recalling that $\mathrm{Var}_j^{(s)}[t] = E_j^{(s)}[t^2] - \left(E_j^{(s)}[t]\right)^2$ we obtain the credibility matrix A_j as

$$A_j = \frac{w_{j\bullet}}{D} \begin{bmatrix} w_{j\bullet} \mathrm{Var}_j^{(s)}[t] + \kappa_1 & \kappa_1 E_j^{(s)}[t] \\ \kappa_0 E_j^{(s)}[t] & w_{j\bullet} \mathrm{Var}_j^{(s)}[t] + \kappa_0 E_j^{(s)}[t^2] \end{bmatrix}. \tag{3.30}$$

To compute the credibility matrices A_j for the Hachemeister data we will need to find estimators (Hans Bühlmann and Gisler 2005, 216 and 217) for the three parameters σ^2, τ_0^2, and τ_1^2. They are as follows: consider an estimator of the variance across time for a single state:

$$\hat{\sigma}_j^2 = \frac{1}{n-2} \sum_{t=1}^{N} w_{jt} \left(X_{jt} - \hat{\mu}_{jt}\right)^2,$$

where $\hat{\mu}_{jt}$ are the fitted values from the regression equation for state j. These we can compute via

```
Ys <- cbind(rep(1, 12), 1:12)
mu.jt <- as.vector(Ys %*% Bj)
sigmaj.sq <- tapply(W.jt * (X.jt - mu.jt)^2, S, sum) / (N - 2)

rm(Ys, mu.jt)
```

Then take as an estimator for σ^2 the average of the individual state estimators; that is,

$$\sigma^2 = \frac{1}{J}\sum_{j=1}^{J}\hat{\sigma}_j^2.$$

```
sigma.sq <- mean(sigmaj.sq)
```

For the estimators of the variances of the intercept, τ_0^2, and slope, τ_1^2, we use estimators that are analogous to those in the Bühlmann-Straub model. That is

$$\hat{\tau}_0^2 = c_0\left\{\frac{J}{J-1}\sum_{j=1}^{J}\frac{w_{j\bullet}}{w_{\bullet\bullet}}\left(B_{0j}-\bar{B}_0\right)^2 - \frac{J\hat{\sigma}^2}{w_{\bullet\bullet}}\right\},$$

where

$$c_0 = \frac{J-1}{J}\left\{\sum_{j=1}^{J}\frac{w_{j\bullet}}{w_{\bullet\bullet}}\left(1-\frac{w_{j\bullet}}{w_{\bullet\bullet}}\right)\right\}^{-1},$$

and

$$\bar{B}_0 = \sum_{j=1}^{J}\frac{w_{j\bullet}}{w_{\bullet\bullet}}B_{0j}.$$

We can compute these quantities by starting with $\bar{B}_0$.

```
B0.bar <- sum(W.jb / W.bb * Bj[1,])
```

Then we will need c_0 which we will brake up into smaller terms.

$$\text{term}_1 = \frac{J-1}{J}, \quad \text{term}_2 = \frac{w_{j\bullet}}{w_{\bullet\bullet}}, \quad \text{term}_3 = 1-\frac{w_{j\bullet}}{w_{\bullet\bullet}}, \quad \text{term}_4 = \sum_{j=1}^{J}\frac{w_{j\bullet}}{w_{\bullet\bullet}}\left(1-\frac{w_{j\bullet}}{w_{\bullet\bullet}}\right)$$

and finally calculating c_0 as

$$c_0 = \frac{\text{term}_1}{\text{term}_4}.$$

```
term.1 <- (J - 1) / J
term.2 <- W.jb / W.bb
term.3 <- 1 - term.2
term.4 <- sum(term.2 * term.3)
c0 <- term.1 / term.4

rm(term.1, term.2, term.3, term.4)
```

And for the calculation of τ_0^2 we also brake it up into more manageable pieces:

$$\text{term}_1 = \frac{J}{J-1}, \quad \text{term}_2 = \frac{w_{j\bullet}}{w_{\bullet\bullet}}, \quad \text{term}_3 = \left(B_{0j} - \bar{B}_0\right)^2, \quad \text{term}_4 = \frac{J\sigma^2}{w_{\bullet\bullet}}$$

```
term.1 <- J / (J - 1)
term.2 <- W.jb / W.bb
term.3 <- (Bj[1,] - B0.bar)^2
term.4 <- J * sigma.sq / W.bb
tau0.sq <- c0 * (term.1 * sum(term.2 * term.3) - term.4)

rm(term.1, term.2, term.3, term.4)
```

And similarly for τ_1^2. The formulas are the same with a small change. Instead of using $w_{j\bullet}$ we shall use $w^*_{j\bullet}$ where

$$w^*_{j\bullet} = \text{Var}_j^{(s)}[t] \cdot w_{j\bullet}$$

Therefore, we have

$$\hat{\tau}_1^2 = c_1 \left\{ \frac{J}{J-1} \sum_{j=1}^{J} \frac{w^*_{j\bullet}}{w^*_{\bullet\bullet}} \left(B_{1j} - \bar{B}_1\right)^2 - \frac{J\hat{\sigma}^2}{w^*_{\bullet\bullet}} \right\},$$

where

$$c_1 = \frac{J-1}{J} \left\{ \sum_{j=1}^{J} \frac{w^*_{j\bullet}}{w^*_{\bullet\bullet}} \left(1 - \frac{w^*_{j\bullet}}{w^*_{\bullet\bullet}}\right) \right\}^{-1},$$

and

$$\bar{B}_1 = \sum_{j=1}^{J} \frac{w^*_{j\bullet}}{w^*_{\bullet\bullet}} B_{1j}.$$

The values of B_{0j} and B_{1j} are the intercept and slope for each individual state j and so we have that $\bar{B}_0$ and $\bar{B}_1$ are the weighted averages of the estimated state parameters.

The calculation for τ_1^2 is analogous to τ_0^2. The code to accomplish this follows:

```
B1.bar <- sum(Ws.jb / Ws.bb * Bj[2,])
```

```
term.1 <- (J - 1) / J
term.2 <- Ws.jb / Ws.bb
term.3 <- 1 - term.2
term.4 <- sum(term.2 * term.3)
c1 <- term.1 / term.4

rm(term.1, term.2, term.3, term.4)
```

```
term.1 <- J / (J - 1)
term.2 <- Ws.jb / Ws.bb
term.3 <- (Bj[2,] - B1.bar)^2
term.4 <- J * sigma.sq / Ws.bb
tau1.sq <- c1 * (term.1 * sum(term.2 * term.3) - term.4)

rm(term.1, term.2, term.3, term.4)
```

These calculations yield

$$\hat{\sigma}^2 = 4.9870187 \times 10^7, \quad \hat{\tau}_0^2 = 1.8029435 \times 10^4, \quad \hat{\tau}_1^2 = 665.5618.$$

With these parameter estimates we can now calculate the credibility matrices A_j, the collective intercept and slope, and the credibility weighted estimates for each state. Note that the Hachemeister data had 5 states, but even if it had data for all 50 states, we would still only need to estimate three parameters. Moreover, these parameters are not specific to these five states. We have treated these states as coming from a population of states and these parameters estimate features of the population.

Now that we have estimates for σ^2, τ_0^2, and τ_1^2 we can calculate the credibility weighted estimates of the intercept and slope for our states. The following code sets up the necessary quantities:

```
k0 <- sigma.sq / tau0.sq
k1 <- sigma.sq / tau1.sq

Determ <- (W.jb + k0) * (W.jb * Ej.t2 + k1) - (W.jb * Ej.t)^2
term.11 <- W.jb * Vj.t + k1
term.12 <- k1 * Ej.t
term.21 <- k0 * Ej.t
term.22 <- W.jb * Vj.t + k0 * Ej.t2

#rm(k0, k1)
```

Using Equation 3.30 we can define a function that will return the credibility matrix A_j as:

```
A <- function(j) {
  M <- matrix(c(term.11[j], term.12[j],
                term.21[j], term.22[j]),
              nrow = 2, ncol = 2, byrow = TRUE)
  ans <- W.jb[j] / Determ[j] * M
  dimnames(ans) <- list(c("", ""),
                        c("", ""))
  return(ans)
}
```

We will also need the collective's estimate of the intercept and slope. This estimate is equal to the weighted average of the individual state estimates where the weights are given by the credibility matrices as follows

$$B_{\text{GLS}} = \left(\sum_{j=1}^{J} A_j \right)^{-1} \sum_{j=1}^{J} A_j B_j,$$

where B_j is the estimate of the intercept and slope for state j. We have labelled the estimate with the subscript 'GLS' because it turns out that this estimate is equal to the generalized least squares (GLS) estimate.

```
B.gls <- solve(A(1) + A(2) + A(3) + A(4) + A(5)) %*%
  (A(1) %*% Bj[,1] + A(2) %*% Bj[,2] + A(3) %*% Bj[,3] +
     A(4) %*% Bj[,4] + A(5) %*% Bj[,5])
```

The credibility weighted estimate for state j, which we will label as CW_j, is equal to

$$CW_j = A_j B_j + (I - A_j) B_{\text{GLS}},$$

where I is a 2×2 identity matrix.

```
CW <- function(j){
  I <- diag(1, nrow = 2, ncol = 2)
  ans <- A(j) %*% Bj[,j] + (I - A(j)) %*% B.gls
  dimnames(ans) <- list(c("Intercept", "Slope"),
                        paste("State", j, sep = " "))
  return(ans)
}
```

For state 1 the credibility matrix A_1 is

```
round(A(1), 4)
```

```
 0.8946 0.3389
 0.0125 0.9460
```

and the credibility weighted estimate of the intercept and slope for state 1 are

```
round(CW(1), 2)
```

```
          State 1
Intercept 1652.61
Slope       62.63
```

Assembling the credibility matrices, stand-alone, credibility weighted, and collective estimates we have

```
CM <- rbind(A(1), A(2), A(3), A(4), A(5))
dimnames(CM) <- list(rep(1:5, each = 2),
                     c("Credibility", "Matrix"))
BM <- rbind(Bj[1,1], Bj[2,1],
            Bj[1,2], Bj[2,2],
            Bj[1,3], Bj[2,3],
            Bj[1,4], Bj[2,4],
            Bj[1,5], Bj[2,5])
CR <- rbind(CW(1), CW(2), CW(3), CW(4), CW(5))
BG <- rbind(B.gls,B.gls,B.gls,B.gls,B.gls)
ST <- cbind(CM, BM, CR, BG)
dimnames(ST) <- list(rep(1:5, each = 2),
                     c("Credibility", "Matrix",
                       "Standalone", "Credibility", "Collective"))
round(ST, 4)
```

```
  Credibility Matrix Standalone Credibility Collective
1      0.8946 0.3389  1658.4724   1652.6053  1495.7471
1      0.0125 0.9460    62.3925     62.6304    29.0943
2      0.6583 1.0286  1398.3025   1419.3021  1495.7471
2      0.0380 0.8222    17.1397     15.5656    29.0943
3      0.6029 1.1851  1532.9987   1535.0520  1495.7471
3      0.0437 0.7740    43.3073     41.7250    29.0943
```

```
4       0.3930 1.4753  1176.7041   1368.4757  1495.7471
4       0.0545 0.6122    27.8070     10.9302    29.0943
5       0.7658 0.7245  1521.8993   1503.3002  1495.7471
5       0.0267 0.8812    11.8745     14.6204    29.0943
```

```
rm(CM, BM, CR, BG)
```

The last column labeled 'Collective' repeats its entries for every state because we only have a single estimate for the entire portfolio of states. For the 'Standalone', 'Credibility', and 'Collective' columns we have listed the estimate of the intercept first and the slope second.

Carefully inspecting the credibility estimates for each state reveals some peculiarities that (Hachemeister 1975, 153) noted in his work as follows

> State #4 trend lines clearly point out a distressing aspect of the credibility adjusted trend line. The credibility adjusted trend line has a lower trend than both the country wide and the state trend line.

This is clearly seen in Figure 3.7 where we have one panel for each state and have included the collective estimate of the trend line (dark green line), the stand-alone state estimate (light purple line), and the credibility weighted trend estimate (dark purple line).

Exercise

By carefully inspecting the table of estimates, for both intercept and slope, which of the credibility estimates are not between the collective and stand-alone figures?

Solution

For state 1, both the credibility slope and intercept are outside the intervals defined by the stand-alone and collective estimates and for state 2, only the slope is not between the stand-alone and collective estimates. State 3 has its intercept outside the collective and the stand-alone figures. The slope for state 4, as we have seen, is outside. Only state 5 has both its intercept and slope within the intervals defined by the stand-alone and collective estimates.

```
db2 <- db
db2$SA <- NA   # Stand-alone prediction
db2$CW <- NA   # Credibility weighted prediction
db2$CO <- NA   # Collective prediction
for (j in 1:5) {
  idx <- db2$state == j
  Y <- cbind(rep(1, sum(idx)),
```

```
              T.jt[idx])
  db2$SA[idx] <- Y %*% Bj[,j]
  db2$CW[idx] <- Y %*% CW(j)
  db2$CO[idx] <- Y %*% B.gls
}
rm(j, idx, Y)
```

```
p <- ggplot(data = db2,
       mapping = aes(x = time,
                     y = severity,
                     group = state)) +
  geom_line(mapping = aes(x = time, y = SA),
            color = "#f1b6da") +
  geom_line(mapping = aes(x = time, y = CW),
            color = "#d01c8b") +
  geom_line(mapping = aes(x = time, y = CO),
            color = "#4dac26") +
  labs(x = "Time (in Qs)",
       y = "Severity") +
  scale_x_continuous(breaks = (1:6) * 2,
                     minor_breaks = NULL) +
  coord_cartesian(ylim = c(1204.51, 2809.63)) +
  facet_wrap(vars(state), labeller = label_both)
p
```

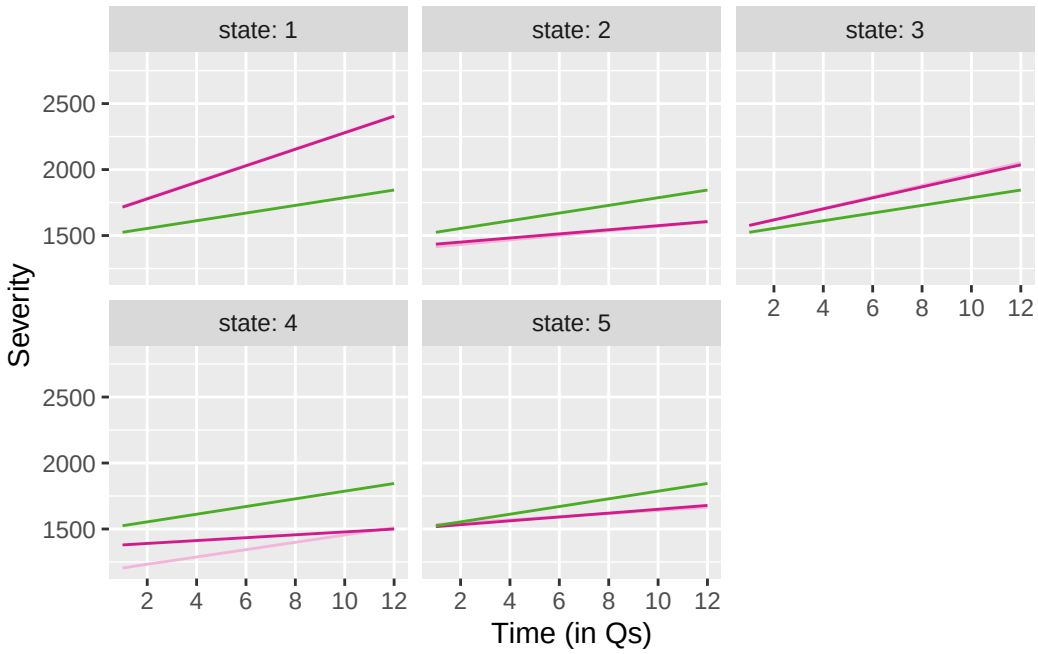


Figure 3.7: Credibility estimates for Hachemeister data. The green colored line corresponds to the collective estimate. The light purple line is the stand-alone estimate for the state and the dark purple line is the credibility weighted estimate.

Several authors (F. De Vylder 1981; F. De Vylder 1985; H. Bühlmann and Gisler 1997) have noticed the strange credibility estimates arrived at by Hachemeister and many actuaries would not apply these methods in practice. Some authors (F. De Vylder 1981; F. De Vylder 1985) tried to impose constraints to resolve the issues and others (Danneburg 1996) have pointed out that these constraints have drawbacks. In 1997, (H. Bühlmann and Gisler 1997) found a simple solution. Recall that the credibility matrix A_j in Equation 3.30 is equal to

$$A_j = \frac{w_{j\bullet}}{D}\begin{bmatrix} w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_1 & \kappa_1 E_j^{(s)}[t] \\ \kappa_0 E_j^{(s)}[t] & w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_0 E_j^{(s)}[t^2] \end{bmatrix}$$

and looking at the off-diagonal elements; namely, $\kappa_1 E_j^{(s)}[t]$ and $\kappa_0 E_j^{(s)}[t]$, we might want to make them equal to zero. If our credibility matrix, A_j, is diagonal then the credibility weighted estimates of the intercept and slope would be split into two individual credibility calculations: one for the intercept and one for the slope. Currently with the credibility matrix we have, the estimate for the intercept involves combining both the intercepts and slopes of the stand-alone and collective estimates. Similarly, the credibility estimate of the slope is a combination of both the intercept and slope estimates of the state and the collective.

So how could those off-diagonal elements be zero? That is, how could we make $E_j^{(s)}[t]$ be zero?

Remembering that

$$E_j^{(s)}[t] = \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} t$$

we could shift our time variable t so that the above expression is equal to zero. In other words, we would like to replace t with $t - t_0$ such that $E_j^{(s)}[t - t_0] = 0$. Namely, we would let

$$t_0 = \sum_{t=1}^{N} \frac{w_{jt}}{w_{j\bullet}} t.$$

Note that t_0 is the weighted average of the time variable for state j. We could also call t_0 the 'center of gravity' for state j. With this translation of the time axis we are putting the intercept of our model at time $t = t_0$ instead of at the traditional origin of time $t = 0$.

In this case, the new credibility matrix A'_j becomes

$$\begin{aligned} A'_j &= \frac{w_{j\bullet}}{D'} \begin{bmatrix} w_{j\bullet}\text{Var}_j^{(s)}[t - t_0] + \kappa_1 & \kappa_1 E_j^{(s)}[t - t_0] \\ \kappa_0 E_j^{(s)}[t - t_0] & w_{j\bullet}\text{Var}_j^{(s)}[t - t_0] + \kappa_0 E_j^{(s)}[(t - t_0)^2] \end{bmatrix} \\ &= \frac{w_{j\bullet}}{D'} \begin{bmatrix} w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_1 & 0 \\ 0 & (w_{j\bullet} + \kappa_0)\text{Var}_j^{(s)}[t] \end{bmatrix} \end{aligned}$$

where

$$D' = (w_{j\bullet} + \kappa_0)(w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_1),$$

and noting that variances are not affected by a linear translation and $E_j^{(s)}[(t - t_0)^2] = \text{Var}_j^{(s)}[t]$. We can simplify to obtain the following credibility matrix

$$A'_j = \begin{bmatrix} \frac{w_{j\bullet}}{w_{j\bullet} + \sigma^2/\tau_0^2} & 0 \\ 0 & \frac{w_{j\bullet}\text{Var}_j^{(s)}[t]}{w_{j\bullet}\text{Var}_j^{(s)}[t] + \sigma^2/\tau_1^2} \end{bmatrix}.$$

The diagonal entries are of the form of the Bühlmann-Straub credibility factors. Hence, the credibility weighted intercept and slope will be strictly between the stand-alone and collective estimates, respectively.

We were able to transform the original credibility matrix A_j into a diagonal credibility matrix A'_j by translating the origin of time to the center of gravity. We did all this for a particular state j and there is no guarantee that the centers of gravity for the states will all coincide with one another. For the Hachemeister data, the individual centers of gravity are

```
CG <- tapply(W.jt * T.jt, S, sum) / W.jb
round(CG, 3)
```

```
    1     2     3     4     5
6.450 6.588 6.300 6.339 6.563
```

and notice that they are all close to each other. The largest difference between any two states is 0.288.

```
j0 <- sum(W.jb * CG) / W.bb
```

Bühlmann and Gisler (Hans Bühlmann and Gisler 2005, 214) have noted that in practice the centers of gravity are usually close to each other and by translating the time variable to the overall center of gravity, the credibility matrices would be nearly diagonal. Table 3.4 shows the credibility estimates when we translate the origin of time to the global center of gravity of 6.475. Note that the credibility matrices are nearly diagonal and the estimated slope for state 4 is now between the standalone and collective values. Also all other credibility estimates are between the stand-alone and collective values.

```
sg <- sig.sq(X.jt, T.jt - j0, W.jt, db$state)$sigma.sq
D <- tau(sg, X.jt, T.jt, W.jt, db$state)$D
CW.one.center <- HBG(sg, D, X.jt, T.jt - j0, W.jt, db$state, use.B.gls = TRUE)

rm(sg, D)
```

```
kbl(CW.one.center$tb,
    digits = c(0, 4, 4, 2, 2, 2),
    row.names = FALSE,
    col.names = c("State", "Col.1", "Col.2", "Standalone",
                  "Credibility", "Collective"),
    align = "crrrrr",
    format.args = list(big.mark = ','),
    booktabs = TRUE,
    linesep = "") %>%
  kable_styling(full_width = FALSE) %>%
  add_header_above(c(" " = 1, "Credibility Matrix" = 2, " " = 3))
rm(CW.one.center)
```

Just as we translated the origin of time to the global center of gravity we could do the time translation on a state-by-state basis. This would then yield diagonal credibility matrices for each state. In Table 3.5 we have done just that and comparing all the estimates to those in Table 3.4 we can see that, in this example, the differences are quite small.

```
j0 <- rep(CG, each = 12)
sg <- sig.sq(X.jt, T.jt - j0, W.jt, db$state)$sigma.sq
D <- tau(sg, X.jt, T.jt, W.jt, db$state)$D
CW.many.centers <- HBG(sg, D, X.jt, T.jt - j0, W.jt, db$state, use.B.gls = TRUE)
```

Table 3.4: Credibility matrices and estimated stand-alone, credibility, and collective intercept and slope for the Hachemeister data when the time variable has been centered at the global center of gravity. Note that the off-diagonal elements of the credibility matrices are nearly zero. For standalone, credibility, and collective estimates the intercept is listed first and the slope second.

	Credibility Matrix				
State	Col.1	Col.2	Standalone	Credibility	Collective
1	0.9731	-0.0014	2,062.46	2,052.54	1,694.98
1	-0.0001	0.9413	62.39	60.69	33.72
2	0.8779	0.0236	1,509.28	1,531.57	1,694.98
2	0.0009	0.7629	17.14	20.91	33.72
3	0.8321	-0.0454	1,813.41	1,793.09	1,694.98
3	-0.0017	0.6881	43.31	40.12	33.72
4	0.6000	-0.0483	1,356.75	1,492.32	1,694.98
4	-0.0018	0.4079	27.81	31.91	33.72
5	0.9288	0.0118	1,598.79	1,605.38	1,694.98
5	0.0004	0.8559	11.87	14.98	33.72

```
rm(j0, sg, D)
```

```
kbl(CW.many.centers$tb,
    digits = c(0, 4, 4, 2, 2, 2),
    row.names = FALSE,
    col.names = c("State", "Col.1", "Col.2", "Standalone",
                  "Credibility", "Collective"),
    align = "crrrrr",
    format.args = list(big.mark = ','),
    booktabs = TRUE,
    linesep = "") %>%
  kable_styling(full_width = FALSE) %>%
  add_header_above(c(" " = 1, "Credibility Matrix" = 2, " " = 3))
```

```
Y <- matrix(c(1,1,1,12), nrow = 2, ncol = 2)
mu.st <- Y %*% reduce(CW.many.centers$B, cbind)
mu.cw <- Y %*% reduce(CW.many.centers$CW, cbind)
mu.co <- Y %*% CW.many.centers$B.col
tb <- tibble(state = rep(1:5, each = 2),
             time = rep(c(1, 12), 5),
```

Table 3.5: Credibility matrices and estimated stand-alone, credibility, and collective intercept and slope for the Hachemeister data when the time variable for each state has been centered at its own center of gravity. Note that the off-diagonal elements of the credibility matrices are exactly zero. For standalone, credibility, and collective estimates the intercept is listed first and the slope second.

	Credibility Matrix				
State	Col.1	Col.2	Standalone	Credibility	Collective
1	0.9731	0.0000	2,060.92	2,051.04	1,693.42
1	0.0000	0.9413	62.39	60.71	33.67
2	0.8779	0.0000	1,511.22	1,533.46	1,693.42
2	0.0000	0.7628	17.14	21.06	33.67
3	0.8324	0.0000	1,805.84	1,787.00	1,693.42
3	0.0000	0.6880	43.31	40.30	33.67
4	0.6002	0.0000	1,352.98	1,489.09	1,693.42
4	0.0000	0.4077	27.81	31.28	33.67
5	0.9288	0.0000	1,599.83	1,606.49	1,693.42
5	0.0000	0.8559	11.87	15.02	33.67

```
              sev.st = as.vector(mu.st),
              sev.cw = as.vector(mu.cw),
              sev.co = rep(as.vector(mu.co), 5))
tc <- tibble(state = 1:5,
              time = CG)

rm(Y, mu.st, mu.cw, mu.co, CG)
```

```
ggplot(data = tb,
       mapping = aes(x = time,
                     group = state)) +
  geom_vline(data = tc,
             mapping = aes(xintercept = time,
                           group = state),
             color = "gray") +
  geom_line(mapping = aes(y = sev.st),
            color = "#f1b6da") +
  geom_line(mapping = aes(y = sev.cw),
            color = "#d01c8b") +
  geom_line(mapping = aes(y = sev.co),
            color = "#4dac26") +
```

```
  scale_x_continuous(breaks = seq(2, 12, by = 2),
                     minor_breaks = NULL) +
  coord_cartesian(ylim = c(1204.51, 2809.63)) +
  labs(x = "Time (in Qs)",
       y = "Severity") +
  facet_wrap(vars(state), labeller = "label_both")

rm(tb, tc)
```

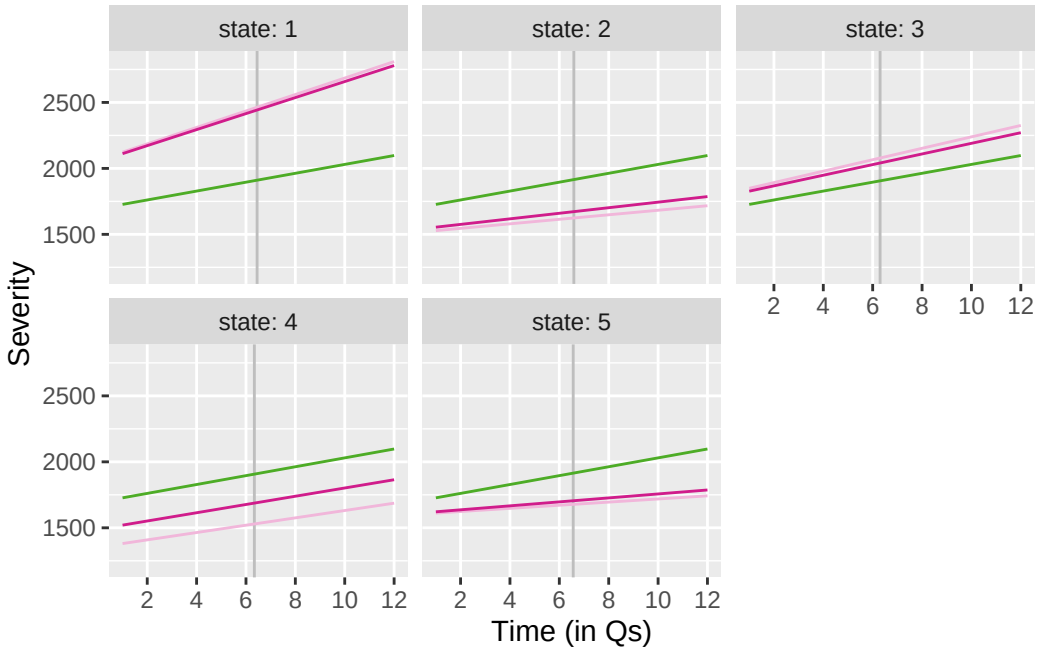


Figure 3.8: Credibility estimates for Hachemeister's data with time translation to the center of gravity for each state. Each state center of gravity is shown as gray vertical line. Note that all the credibility weighted intercepts (at the center of gravity) and slopes (dark purple) are now strictly between the stand-alone estimates (light purple) and the collective estimates (dark green).

Figure 3.8 shows the credibility regression lines when we translate the time variable to the center of gravity (shown as a vertical gray line) for each state. Note how each intercept, at the center of gravity, is strictly between the state stand-alone estimate and the collective estimate. Also, the credibility adjusted slopes are between the stand-alone and the collective estimates. In particular, state 4 now has a very plausible regression line. Compare its panel here (Figure 3.8) with its panel in Figure 3.7.

3.6 Summary

In this chapter we have discussed the idea that the data we have collected on some risks is more credible than the data on other risks. For those risks whose data is credible we can use it with confidence to predict next year. But for those risks whose data is not fully credible we can combine its own data with the data for all risks to come up with a better prediction for next year. Credibility theory tells us how we should put together the collective's data and the own risk data in an optimal way by weighting the two sources together.

In the first section, limited fluctuation credibility, we showed how probabilistic considerations about the aggregate loss model can lead us to establish a criterion based on the expected number of claims necessary to be confident that the results will be within a small margin of error.

The second section was devoted to the balanced Bühlmann model where we established that next year's premium should be a weighted average of a risk's own experience and the experience of the collective of all risks. Namely, the credibility premium has the form

$$z\bar{X}_j + (1-z)\bar{X} \qquad \text{with} \qquad z = \frac{n}{n+\sigma^2/\tau^2}$$

where σ^2 and τ^2 are known, in the actuarial world, as the expected value of the process variance (EVPV) and is the variance of the hypothetical means, respectively. In the statistical literature, these are known as the *within variance* and the *between variance*, respectively. Note that as the expected value of the process variance (within variance), σ^2, increases, the credibility factor z decreases. Similarly, as the variance of the hypothetical means (between variance), τ^2, decreases towards zero, the credibility factor, z, decreases.

The balanced Bühlmann model is critical to our understanding of credibility procedures, but it is not a very useful model in practice as it assumes that all risks have the same exposure and are observed over the same number of periods.

The third section focused on extending the balanced Bühlmann model to a practically useful model known as the Bühlmann-Straub model. In this model, each risk comes with its own exposure weight and not all risks need to be observed over the same time period. With these extensions the credibility premium is of the same form as in the balanced Bühlmann model; namely,

$$Z_j X_{j\bullet} + (1-Z_j)X_z \qquad \text{with} \qquad Z_j = \frac{w_{j\circ}}{w_{j\circ}+\sigma^2/\tau^2}.$$

Again we see that the credibility premium has the same weighted average form as in the balanced Bühlmann model and the credibility factors have also the same form.

In the last section, we explored a credibility regression model first proposed by Hachemeister. Here the basic idea is that we have data on several risks and we would like to estimate a regression line on this data. We could ignore that data came from individual risk and fit one regression line to all of the data. But this approach discards important information. We could

also fit individual regression lines on each risk. For some risks the volume of information would be large enough to give us a 'robust' regression line, but for some risks their volume would be small and we might get some spurious results.

In this situation credibility theory can be applied to estimate the regression coefficients as weighted averages of the individual regression coefficients and the collective regression coefficients. Unfortunately, a naive application of credibility to Hachemeister's data led to implausible results for state 4 where the credibility weighted trend for this state was both lower than its stand-alone and collective estimates.

This implausible result arises from the fact that in this case the credibility factors are 2×2 matrices with non-zero entries in all four positions and thus the credibility weighted intercept and slope are a complex combination of *both* stand-alone and collective intercepts and slopes.

To obtain plausible estimates required two key insights (H. Bühlmann and Gisler 1997):

- Assume that the variance-covariance matrix of the intercept and slope random coefficients is diagonal, and
- Center the time variable at each risks' time center of gravity

With these insights the credibility factors are 2×2 diagonal matrices and so the credibility weighted intercept and slope are each calculated separately yielding estimates that are always between the individual risk and the collective values.

4 Linear Mixed Models

In Chapter 3 we looked at the balanced Bühlmann, the Bühlmann-Straub, and Hachemeister's credibility regression models. These models have also been studied by statisticians under various names: linear mixed models, hierarchical models, longitudinal models, and panel models, to name a few. The statistical literature on these models is extensive and the models are well developed. Actuaries can benefit significantly by using this theory and the tools to assess these models.

We will introduce linear mixed models by reframing these credibility models in the standard statistical notation and exploring the tools that have been developed to fit and assess them. This will allow us to see how the linear mixed model theory can be applied to practical problems in credibility theory.

4.1 Balanced Bühlmann Model Revisited

One way to write the balanced Bühlmann model, see Equation 3.6, is as follows

$$X_{jt} = \mu + \Xi_j + \epsilon_{jt}, \qquad j = 1, 2, \dots, J, \quad t = 1, 2, \dots, T,$$

where X_{jt} is the observation for group j at time t, μ is the overall mean, Ξ_j is a random deviation from the overall mean for group j and ϵ_{jt} is an error term for the j, t observation. This conforms to the actuarial notation but not to the statistician's. So we will switch notation to what is commonly used in statistics.

The response variable in typically named Y (either uppercase or lowercase) and the explanatory variables are usually denoted by X (also either uppercase or lowercase). We can express the balanced Bühlmann model as

$$y_{jt} = \beta + b_j + \epsilon_{jt},$$

where β is the overall mean, b_j is a random variable representing the deviation from the overall mean for the jth group and ϵ_{jt} is the deviation for the j, t observation. Both b_j and ϵ_{jt} are **deviations** and so we know that their means are zero.

Since b_j and ϵ_{jt} are random variables we need to specify their distributions and how they might be related to each other. For this model, we will have both of them to be independent, with constant variance, and normally distributed. The variance of b_j will be denoted by σ_b^2 and denotes the **between** group variability. Actuaries call this the *variance of the hypothetical*

means (VHM). The hypothetical means are $\beta + b_j$. The variance of ϵ_{jt} is known as **within** group variability and is denoted by σ^2. In actuarial groups this is known an the *expected value of the process variance* (EVPV). We can write these specifications as

$$b_j \sim \mathcal{N}(0, \sigma_b^2) \qquad \text{and} \qquad \epsilon_{jt} \sim \mathcal{N}(0, \sigma^2),$$

where $\mathcal{N}(0, \sigma^2)$ represents the normal distribution with mean equal to 0 and variance equal to σ^2.

For the balanced Bühlmann model we did not make the assumption that our random variables were normally distributed. We only assumed that they had a finite first and second moments and we used the method of moments to derive estimates for variances. It turns out that the maximum likelihood estimates of these variances coincides with the method of moments for this simple model. Hence, the balanced Bühlmann model is equivalent to this linear mixed model.

Statisticians would call β a fixed effect and b_i a random effect and since this model has both fixed and random effects it is called a *mixed effects* model. The naming of coefficients as either fixed or random is not without controversy. Gelman and Hill (2007, 245) outline five definitions for these terms and in their work they avoid the use of these terms.

Using the same data as in Table 3.1; namely,

```
dta <- tibble(class = factor(rep(1:3, each = 4),
                             levels = 1:3),
              time = rep(1:4, times = 3),
              value = c(625, 675, 600, 700,
                        750, 800, 650, 800,
                        900, 700, 850, 950))
```

we will illustrate the fitting of a linear mixed effects model and show that we arrive at the same estimates. But rather than jumping straight into that mixed model we want to describe one process of fitting and exploring models to reach that mixed model.

We can start simple by fitting an ordinary least squares model that only includes an intercept term. Such a model is usually called a **null** model. It is the most basic model we can have.

```
BB.null.lm <- lm(value ~ 1,
                 data = dta)
(sBB.null.lm <- summary(BB.null.lm))
```

```
Call:
lm(formula = value ~ 1, data = dta)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-150.00  -81.25  -25.00   62.50  200.00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   750.00      32.13   23.34 1.01e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 111.3 on 11 degrees of freedom
```

From the output, we can see that the overall mean is 750, and the residual standard error is 111.29. Thus we know that $\widehat{\beta}_0 = 750$ and $\widehat{\sigma}^2 = 12,386.36$.

```r
dta$res.null <- resid(BB.null.lm)
```

```r
ggplot(data = dta,
       mapping = aes(x = res.null,
                     y = class)) +
  geom_jitter(width = 0, height = 0.1) +
  labs(x = "Residuals",
       y = "Class")
```

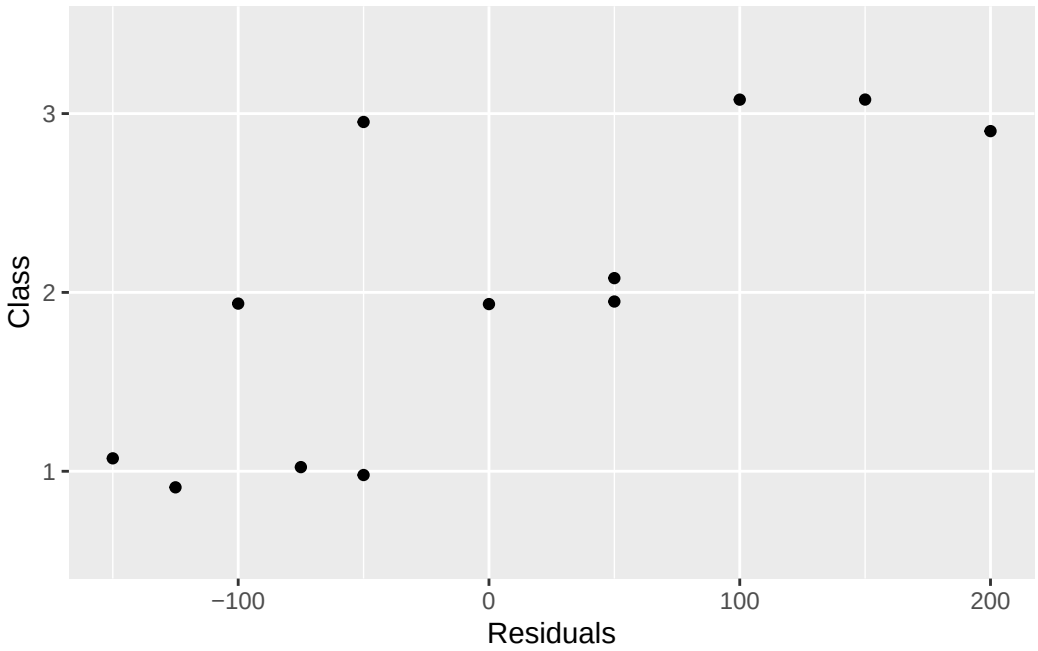


Figure 4.1: Ordinary least squares residuals for the balanced Bühlmann example data. Note that the residuals for class #1 all have the same sign and nearly all points for class #3 also have the same sign. We introduced a slight amount of vertical jittering to avoid over plotting a pair of residuals.

The residuals from this model, see Figure 4.1, show some unsettling patterns. All of the residuals for class #1 are negative and clustered around −100. Similarly, nearly all the residuals for class #3 are positive and also seem to be clustered around 150. Clearly, an intercept only model does not fit the data well and we have an effect from the `class` variable that needs to be incorporated into the model. We can add `class` as a categorical variable to estimate the model.

```
BB.class.lm <- lm(value ~ class - 1,
                  data = dta)
(sBB.class.lm <- summary(BB.class.lm))
```

```
Call:
lm(formula = value ~ class - 1, data = dta)

Residuals:
    Min      1Q  Median      3Q     Max
-150.00  -31.25   12.50   50.00  100.00
```

```
Coefficients:
       Estimate Std. Error t value Pr(>|t|)
class1   650.00      39.53   16.44 5.07e-08 ***
class2   750.00      39.53   18.97 1.44e-08 ***
class3   850.00      39.53   21.50 4.79e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 79.06 on 9 degrees of freedom
Multiple R-squared:  0.9918,    Adjusted R-squared:  0.9891
F-statistic: 364.3 on 3 and 9 DF,  p-value: 1.037e-09
```

In the above model we removed the intercept so that we would estimate a mean value for each class (instead of using one of the classes as an intercept and then estimating deviations from this mean for the other two classes). Note that the residual standard error is now much smaller: 79.06 compared to 111.29. This model fits our data more closely.

```
dta$res.class <- resid(BB.class.lm)
```

```
ggplot(data = dta,
       mapping = aes(x = res.class,
                     y = class)) +
  geom_jitter(width = 0, height = 0.2) +
  labs(x = "Residuals",
       y = "Class")
```

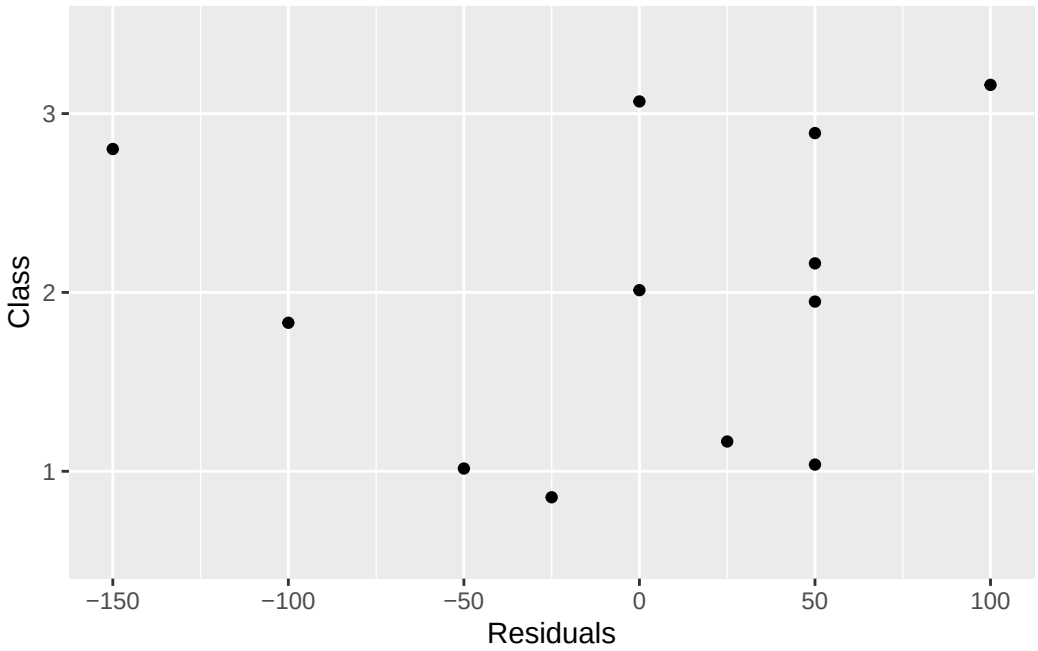


Figure 4.2: Ordinary least squares residuals for the balanced Bühlmann example data with a mean estimate for each class. Note that now all the residuals are centered around zero. We introduced a slight amount of vertical jittering to avoid over plotting a pair of residuals.

Figure 4.2 shows residuals that are much better behaved. They are all centered around zero with both positive and negative values for each class. We might be quite happy with this model if we are interested in just these three classes. But consider that these three classes might have been just a sample from hundreds of classes (say, workers compensation occupational classes). If we were to include all possible classes into a model we may not be able to estimate all of the parameters accurately. Some classes may have lots of data, but others may have very little. More troublesome is the fact that as we add classes, the number of parameters that need to be estimated increases.

So we want to think of the three classes we have as being a sample from a population of classes. Thus, we want to estimate the following mixed model:

$$y_{jt} = \beta + b_j + \epsilon_{jt},$$

where $j = 1, 2, 3$ and $t = 1, 2, 3, 4$. This model has a fixed effect β which is constant across classes and a random deviation from the overall mean, b_j, for each class. We can fit this model with the `lmer()` function from the `lme4` package as follows:

```
BB.mx <- lmer(value ~ 1 + (1 | class),
              data = dta)
```

The response variable is `value` and the first 1 after the '~' says we want a fixed effect intercept. We can specify other fixed effects in this part of the formula as we would in fitting a regular regression model. The component in parenthesis after the plus sign is for the random effects. Here we have `1 | class` because we want a random intercept for each level of the `class` variable.

The parameters to be estimated for this model are: β the fixed effect and σ_b^2 the between-class variance (variance of the hypothetical means), and σ^2 the within-class variance (expected value of the process variance) also known as the residual variance.

```
(sBB.mx <- summary(BB.mx))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: value ~ 1 + (1 | class)
   Data: dta

REML criterion at convergence: 133.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-1.6997 -0.5929  0.1581  0.6325  1.4626

Random effects:
 Groups   Name        Variance Std.Dev.
 class    (Intercept) 8438     91.86
 Residual             6250     79.06
Number of obs: 12, groups:  class, 3

Fixed effects:
            Estimate Std. Error t value
(Intercept)   750.00      57.74   12.99
```

From the output we have $\hat{\beta} = 750$, $\hat{\sigma}_b^2 = 8438$, and $\hat{\sigma}^2 = 6250$. Thus, we can see that the variance between classes is bigger than the variance within a class. The random effects are

```
round(ranef(BB.mx)$class, 3)
```

```
  (Intercept)
1    -84.375
2      0.000
3     84.375
```

which tells us that our estimated mean for class #1 is $750 - 84.375 = 665.625$, for class #2 is $750 + 0 = 750$, and for class #3 is $750 + 84.375 = 834.375$. These are the hypothetical means for each class. Note that these are exactly the same estimates as the credibility estimates we calculated in Chapter 3. The credibility factor can also be easily derived from the above output:

$$Z = \frac{T}{T + \hat{\sigma}^2/\hat{\sigma}_b^2} = \frac{4}{4 + 6250/8437.5} = 0.84375.$$

The value of T represents the number of observations in each group and since we are in the *balanced* Bühlmann model, we know that each group has the same number of observations. The above output tells us that there were 12 observations and 3 groups; hence, $T = 12/3 = 4$.

In the mixed model we have assumed that the residual variance, σ^2, is constant. By plotting the fitted values against the standardized residuals we can check this assumption. Figure 4.3 shows these residuals and even though we have a small sample size the residuals are well behaved.

```
dta$mu.mx <- fitted(BB.mx)
dta$sres.mx <- resid(BB.mx, type = "pearson", scaled = TRUE)
```

```
ggplot(data = dta,
       mapping = aes(x = mu.mx,
                     y = sres.mx)) +
  geom_jitter(width = 3, height = 0) +
  labs(x = "Fitted Values",
       y = "Standardized Residuals")
```

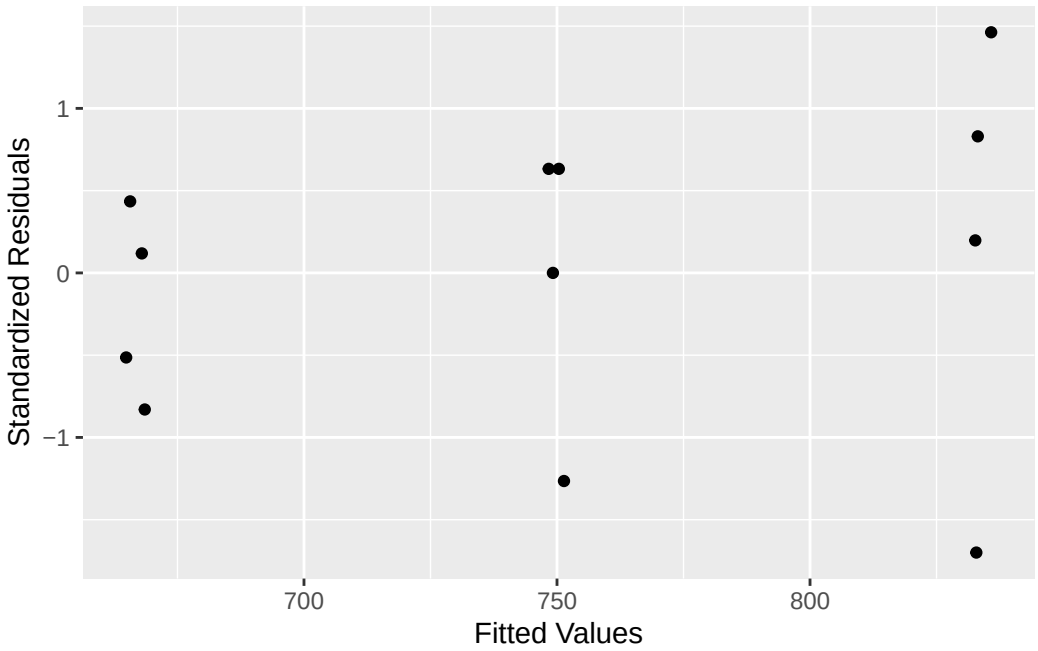


Figure 4.3: Standardized residuals for the balanced Bühlmann data fitted with the mixed model. Note that all residuals are within 1.5 standard deviations from zero. We added a bit of horizontal jittering to avoid over plotting a pair of residuals.

4.2 Bühlmann-Straub Model Revisited

The Bühlmann-Straub model is nearly the same as the balanced Bühlmann model. There are two key differences:

1. We do not assume that all risks have been observed for the same number of periods (no longer have a balanced data set), and
2. We introduce weights so that the residual variance is proportional to the inverse of the weights.

Therefore, the Bühlmann-Straub model can be written as

$$y_{jt} = \beta + b_j + \epsilon_{jt} \qquad \text{with} \quad b_j \sim \mathcal{N}(0, \sigma_b^2) \quad \text{and} \quad \epsilon_{jt} \sim \mathcal{N}\left(0, \frac{\sigma^2}{w_{jt}}\right),$$

where w_{jt} are the weights associated with the observation from risk j and time t.

In Section 3.4 we illustrated the standard actuarial calculations for this model on a simulated data set that had 100 different risk classes and 5 time periods of observations. The parameters

used in the simulation were

$$\beta = 80, \qquad \sigma_b^2 = 64, \qquad \sigma^2 = 100.$$

Note that in the Bühlmann-Straub model we denoted the between risk variance (aka variance of the hypothetical means) by the symbol τ^2, but here we use σ_b^2.

We will use the same simulated data to illustrate how to estimate these parameters via a linear mixed model and so we load the data set we created in the previous chapter.

```
bs.dta <- read_csv("BS-simulated-data.csv",
                   col_types = "fdd")
```

We estimate the Bühlmann-Straub model using the linear mixed effects regression function `lmer()` from the R package `lme4` as follows:

```
BS.mx <- lmer(X.jt ~ 1 + (1 | risk),
              data = bs.dta,
              weights = W.jt)
```

Note that the data set uses the actuarial notation, `X.jt`, for the response variable and `risk` for the name of the class variable. Where the formula in the first argument, $\text{X.jt} \sim 1 + (1 \,|\, \text{risk})$, says that we have a fixed effects intercept, the first 1, and the expression inside parenthesis denotes the random component of the model. In this case, the random component is just an intercept that varies by the classification variable `risk`. We can obtain summary information about the fit via the `summary()` and we have saved the information in an object, `sBS.mx`, to be able to extract some of that information later on.

```
(sBS.mx <- summary(BS.mx))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: X.jt ~ 1 + (1 | risk)
   Data: bs.dta
Weights: W.jt

REML criterion at convergence: 3901.4

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.37735 -0.67636 -0.01581  0.64974  2.76651

Random effects:
```

```
 Groups   Name        Variance Std.Dev.
 risk     (Intercept)  61.14    7.819
 Residual             104.64   10.229
Number of obs: 500, groups:  risk, 100

Fixed effects:
            Estimate Std. Error t value
(Intercept)  78.5384     0.9083   86.46
```

The following table shows the true value of the model parameters, their estimated values from the mixed model, and the estimates from Section 3.4. Note that the mixed model estimates and the credibility estimates are very close to each other and both are not far away from true values we used to simulate the data.

Parameter	True Values	Mixed Model Estimates	Credibility Estimates
β	80	78.5384	78.4363
σ_b^2	64	61.1411	60.9652
σ^2	100	104.6386	104.5239

For the Bühlmann-Straub model the credibility factors differ by risk group j and from the above model output they would be equal to

$$\hat{Z}_j = \frac{w_{j\bullet}}{w_{j\bullet} + \hat{\sigma}^2/\hat{\sigma}_b^2} = \frac{w_{j\bullet}}{w_{j\bullet} + 104.64/61.14},$$

where $w_{j\bullet}$ is the sum of all the weights across time for risk j.

From the linear mixed model, `BS.mx`, we can also obtain values for the deviations, b_j, from the overall mean β. These values are $\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_{100}$. The first 20 of them are

```
round(ranef(BS.mx)$risk[1:20,1], 3)
```

```
 [1] -11.126  -3.886   3.838   3.244  -3.004   5.729   1.625  -6.815  16.494
[10]  -7.125   7.894   1.822  11.837  -5.196  -3.828   0.172  -6.792  14.001
[19]  -6.224  -9.247
```

These values together with the estimate of the fixed effect, $\hat{\beta}$, yields the credibility weighted estimate for each risk; that is, $\hat{\beta} + \hat{b}_j$ is our estimate for risk j. For the first 20 risks we have

```
round(fixef(BS.mx) + ranef(BS.mx)$risk[1:20, 1], 3)
```

```
 [1] 67.412 74.652 82.376 81.782 75.534 84.268 80.164 71.724 95.032 71.414
[11] 86.433 80.360 90.375 73.343 74.710 78.710 71.746 92.539 72.314 69.291
```

We add fitted values to our data set and compute the standardized residuals from our model.

```
bs.dta$mu.mx <- fitted(BS.mx)
bs.dta$sres.mx <- resid(BS.mx, type = "pearson", scaled = TRUE)
```

Figure 4.4 displays the fitted values versus the standardized residuals. The scatter plot of points looks appears like a random cloud of points centered about the line $y = 0$. But if you look closely you might be able to discern a upward sloping pattern.

```
ggplot(data = bs.dta,
       mapping = aes(x = mu.mx,
                     y = sres.mx)) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "Standardized Residuals")
```

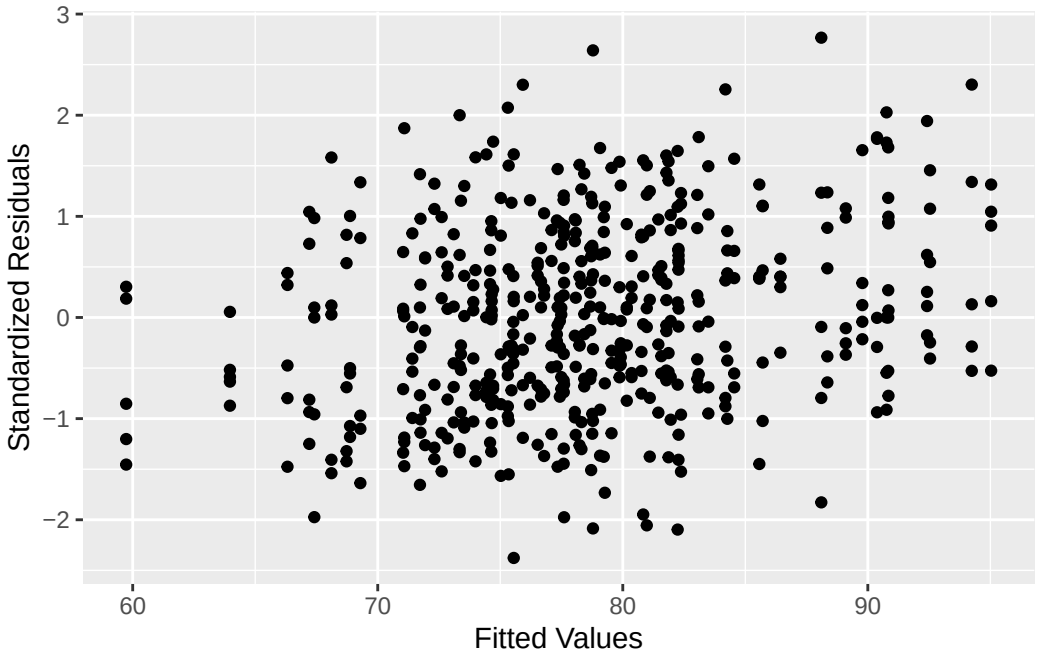


Figure 4.4: Diagnostic plot for the linear mixed model with random intercepts fitted to the simulated Bühlmann-Straub data. All the standardized residuals are within 2.5 standard deviations from the origin. The overall impression is that of a random cloud of points.

Exercise

Fit a linear regression line to the points shown in Figure 4.4 to show that there is as upward sloping pattern in the residuals.

Solution

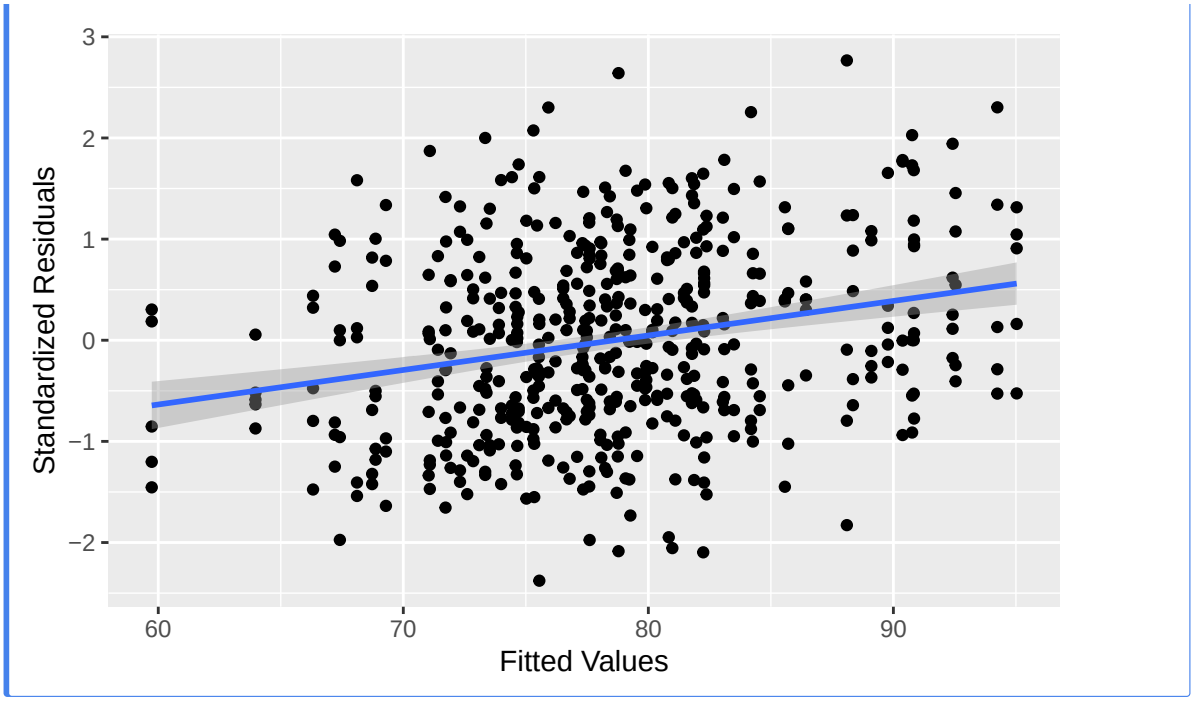
We fit a linear model to the points shown in Figure 4.4 as follows:

```
BS.lm <- lm(sres.mx ~ mu.mx,
            data = bs.dta)
(sBS.lm <- summary(BS.lm))
```

```
Call:
lm(formula = sres.mx ~ mu.mx, data = bs.dta)

Residuals:
     Min       1Q   Median       3Q      Max
-2.27282 -0.66309 -0.05755  0.62719  2.63522

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.683488   0.471425  -5.692 2.14e-08 ***
mu.mx        0.034135   0.005981   5.707 1.97e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8957 on 498 degrees of freedom
Multiple R-squared:  0.0614,    Adjusted R-squared:  0.05951
F-statistic: 32.58 on 1 and 498 DF,  p-value: 1.971e-08
```

The value of the coefficient of `mu.mx` is 0.0341 and from the summary information we can see that it is significant.
We can also show the diagnostic plot with the linear regression line.

```
ggplot(data = bs.dta,
       mapping = aes(x = mu.mx,
                     y = sres.mx)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(x = "Fitted Values",
       y = "Standardized Residuals")
```



A pattern in the residuals would normally suggest that our model does not fit the data well. In an ordinary least squares situation this would be correct and we would conclude that our model does not capture the underlying pattern in the data. But our situation is more complex than ordinary least squares and the upward sloping pattern we are seeing in Figure 4.4 is what we should expect.

The pattern we see is the result of shrinking our estimates towards the overall mean. To see the effect Figure 4.5 shows four panels. The top-left panel displays the response variable on the y-axis and the risk group on the x-axis. The risk groups have been ordered from smallest fitted value based on the linear mixed model to largest. The fitted values from the mixed model are the credibility weighted values given by

$$\hat{y}_{jt} = Z_j\,\bar{y}_j + (1 - Z_j)\,\bar{y},$$

where $\bar{y}_j$ is the average response value for group j, $\bar{y}$ is the collective average, and Z_j is the credibility factor given by

$$Z_j = \frac{w_{j\bullet}}{w_{j\bullet} + \hat{\sigma}^2/\hat{\sigma}_b^2}.$$

```
p1 <- ggplot(data = dtb,
             mapping = aes(x = mu.rnk,
                           y = X.jt)) +
  geom_point(color = "gray", pch = 1) +
  labs(x = "Risk Group",
```

```
       y = "Response Value")

p2 <- p1 + geom_point(data = rsk,
                      mapping = aes(x = mu.rnk,
                                    y = X.bar.j),
                      color = "lightblue")

p3 <- p1 + geom_point(data = rsk,
                      mapping = aes(x = mu.rnk,
                                    y = mu.j),
                      color = "pink")

p4 <- p2 + geom_point(data = rsk,
                      mapping = aes(x = mu.rnk,
                                    y = mu.j),
                      color = "pink")

(p1 + p2) / (p3 + p4)
```

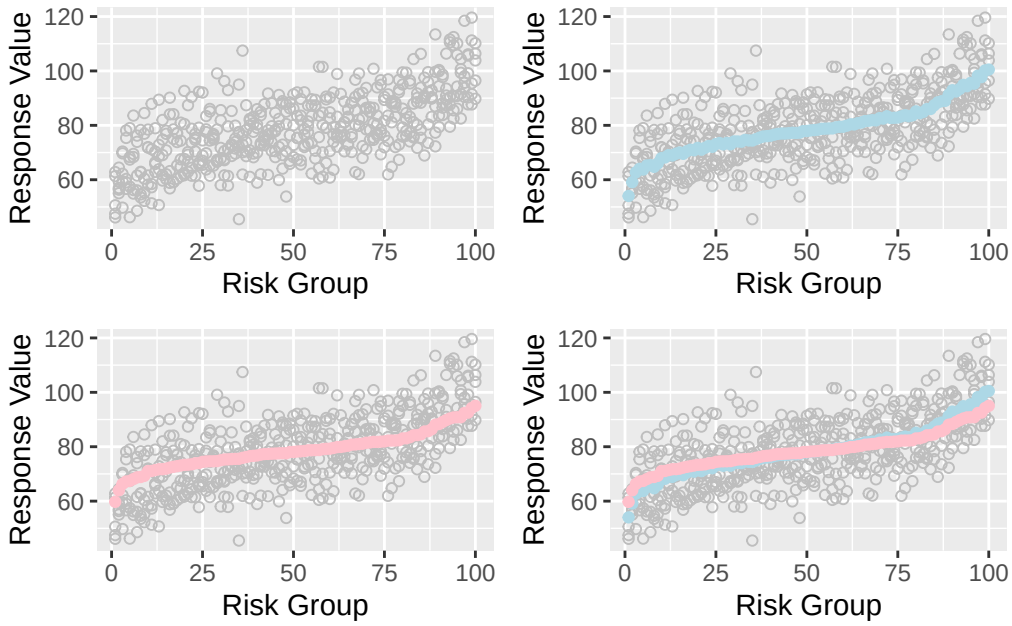


Figure 4.5: The top-left panel shows the simulated Bühlmann-Straub data where the risk groups have been ordered from smallest fitted value to largest. The fitted values come from the linear mixed model and coincide with the credibility weighted values. The top-right panel shows the fitted values and the bottom-left panel shows the average response value for each risk group. The bottom-right panel combines all three panels. Note that the fitted values have been shrunk towards the overall mean value of approximately 80.

For this simulated data each risk group has five observations and from the plot you can see that the *average for each risk group* ranges from below 60 to a bit more than 100. The top-right hand panel includes this average, $\bar{y}_j$, for each risk group.

The bottom-left panel shows the fitted values from the mixed model. These values increase steadily from left to right. Finally, in the bottom-right panel we have superimposed all three panels and we can clearly see that on both ends of the graph the fitted values are closer to the collective mean. These fitted values have been shrunk from the individual group average, $\bar{y}_j$, to the collective average $\bar{y}$.

When we compute the response residuals from this model we are taking the difference between actual values (shown as light gray circles) and the risk group's fitted value (shown in pink). These differences are not centered at the mean value for each risk group's (shown in light blue) and as we approach both extremes the discrepancy increases. On the left-hand side they are more negative and on the right-hand side they are more positive. Therefore, a residuals versus fitted values plot shows a positive trend line.

Exercise

There are three quantities that affect the amount of shrinkage that will occur; they are the weights, w_{jt}, the between risk variance, σ_b^2 (aka variance of the hypothetical means), and the within risk variance, σ^2 (aka expected value of the process variance).
Take each one in turn, and using the credibility factors Z_j, determine the effect on shrinkage that increasing or decreasing each quantity will have.

Solution

The amount of shrinkage is controlled by the size of the credibility factor and for the Bühlmann-Straub model we know that it is given by

$$Z_j = \frac{w_{j\bullet}}{w_{j\bullet} + \sigma^2/\sigma_b^2}.$$

If the value of Z_j is close to 1, then there will be very little shrinkage and the fitted values (credibility estimates) will be close to the average of the group, $\bar{y}_j$.
If we increase the weights, w_{jt}, then Z_j will approach 1. If the within group variance, σ^2, decreases towards zero, then the credibility factor, Z_j, will approach 1. Also, if the between group variance, σ_b^2, increases towards infinity, then again Z_j will approach 1 and the amount of shrinkage will decrease.

In Appendix A we have written a function to simulate data sets that conform to the Bühlmann-Straub model. We can use this function to explore how different values for the number of risks per group and within/between group variances affect the amount of shrinkage.

4.3 Some Linear Mixed Model Theory

In the previous section we estimated the linear mixed models corresponding to the balanced Bühlmann and the Bühlmann-Straub models. We could have gone straight into Hachemeister's regression model to illustrate that too, but it would be better to understand some of the key constructions needed for these model to appreciate more complex situations. Therefore, we will start by laying down some standard notation and constructions for the linear mixed model.

For a single level of grouping, we follow the discussion in Edward W. Frees, Young, and Luo (1999) closely and we can write down the linear mixed effects model as

$$\begin{aligned} y_j &= X_j\beta + Z_j b_j + \epsilon_j, \qquad j = 1, 2, \ldots, J \\ b_j &\sim \mathcal{N}(0, D), \qquad \epsilon_j \sim \mathcal{N}\left(0, R_j\right), \end{aligned}$$

where index j denotes the grouping factor, J is the total number of groups, y_j is a vector of response values for group j with dimension $n_j \times 1$, X_j is the fixed-effects design matrix with

dimensions $n_j \times p$, Z_j is the random-effects design matrix with dimensions $n_j \times q$. The fixed-effects linear predictor consists of p explanatory variables and so we have $\beta = (\beta_1, \beta_2, \ldots, \beta_p)$ as fixed-effects coefficients. We also have q random-effects explanatory variables.

We assume that the responses between groups are independent, but allow for serial correlation and weighting by assuming that the variance-covariance matrix for the error terms, ϵ_j, is an $n_j \times n_j$ matrix which we write as R_j. We also assume that the expected value of the error terms is zero; that is, $\mathbb{E}[\epsilon_j] = 0$. We also assume that the group specific effects, b_j, are independent and identically distributed with $\mathbb{E}[b_j] = 0$ and variance-covariance matrix D with dimensions $q \times q$. Note that the variance-covariance matrix D does not depend on the group. And we assume that the group-specific effects and the error terms are independent; that is, we have that their covariance, $\text{Cov}(b_{ju}, \epsilon_{kv})$ is zero for all combinations of j, u, k, and v. Hence, the variance-covariance matrix for response vector y_j is

$$\begin{aligned}
\text{Var}(y_j) &= \text{Var}(X_j\beta + Z_j b_j + \epsilon_j) \\
&= \text{Var}(Z_j b_j + \epsilon_j) \\
&= \text{Var}(Z_j b_j) + \text{Var}(\epsilon_j) + 2\text{Cov}(Z_j b_j, \epsilon_j) \\
&= Z_j D Z_j^t + R_j \\
&= V_j,
\end{aligned}$$

where a superscript 't' denotes the transpose operation. So we have that the variance-covariance matrix V_j has dimension $n_j \times n_j$ and we further assume that this matrix is invertible. We also know that this matrix is symmetric and if we let $N = \max(n_1, n_2, \ldots, n_J)$ be the maximum number of observations we have across all groups, then this matrix, V_j, has at most $N(N+1)/2$ unknown values. So let τ be the vector of unknown values and we can denote the dependence of V_j on this vector via $V_j(\tau)$.

For the balanced Bühlmann example we discussed in Section 3.3 we have $J = 3$ groups observed over $N = 4$ periods and each group had the same number of observations; that is, $n_j = 4$ for all $j = 1, 2, 3$. The vectors of observations were

$$\begin{aligned}
y_1^t &= (625, 675, 600, 700) \\
y_2^t &= (750, 800, 650, 800) \\
y_3^t &= (900, 700, 850, 950).
\end{aligned}$$

The design matrices X_j and Z_j are all of dimension 4×1 and have only an intercept as explanatory variable; namely $p = q = 1$, thus

$$X_j = Z_j = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

The variance-covariance matrix of the group effects, D, is of dimension 1×1 and we labeled it as σ_b^2 in this chapter and as τ^2 in Section 3.3. We also assumed that error terms, ϵ_j, were independent of each other and so we have

$$R_j = \sigma^2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

And the variance-covariance matrix of the responses for group j, $\text{Var}(y_j) = V_j$, is equal to

$$\begin{aligned} V_j(\tau) &= Z_j D Z_j^t + R_j \\ &= \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} D \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} + \sigma^2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \sigma_b^2 + \sigma^2 & \sigma_b^2 & \sigma_b^2 & \sigma_b^2 \\ \sigma_b^2 & \sigma_b^2 + \sigma^2 & \sigma_b^2 & \sigma_b^2 \\ \sigma_b^2 & \sigma_b^2 & \sigma_b^2 + \sigma^2 & \sigma_b^2 \\ \sigma_b^2 & \sigma_b^2 & \sigma_b^2 & \sigma_b^2 + \sigma^2 \end{bmatrix}; \end{aligned}$$

therefore, the vector of variance components is $\tau = (\sigma_b^2, \sigma^2)$.

The entire model for all observations would be assembled by stacking the response vectors y_1, y_2 and y_3 into a single 12×1 column vector. The grand design matrices X and Z are of dimension 12×3 where the first column has four 1's and zeros after, the second column has four 0's, then four 1's, and then zeroes, and the final column starts with zeroes and ends with four 1's.

$$\begin{bmatrix} 625 \\ 675 \\ 600 \\ 700 \\ 750 \\ 800 \\ 650 \\ 800 \\ 900 \\ 700 \\ 850 \\ 950 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} + \begin{bmatrix} \epsilon_{11} \\ \epsilon_{12} \\ \epsilon_{13} \\ \epsilon_{14} \\ \epsilon_{21} \\ \epsilon_{22} \\ \epsilon_{23} \\ \epsilon_{24} \\ \epsilon_{31} \\ \epsilon_{32} \\ \epsilon_{33} \\ \epsilon_{34} \end{bmatrix}$$

with the variance of the response vector equal to a block diagonal matrix of dimension 12×12 where the first 4×4 block is V_1, the second 4×4 diagonal block is V_2, and the final 4×4

diagonal block is V_3. All other entries are zero.

$$V = \begin{bmatrix} V_1 & & \\ & V_2 & \\ & & V_3 \end{bmatrix}.$$

The generalized least squares (GLS) estimator of the fixed-effects β assumes that the variance components τ are known and is given by

$$\beta_{\text{GLS}} = \left(\sum_{j=1}^{J} X_j^t V_j^{-1} X_j \right)^{-1} \left(\sum_{j=1}^{J} X_j^t V_j^{-1} y_j \right). \tag{4.1}$$

In the balanced Bühlmann model we have $\beta_{\text{GLS}} = \bar{y}$ where $\bar{y}$ is the average of all the response values. To see this, first note that the variance-covariance matrix V_j has dimension $n_j \times n_j$ and is of the form (4×4 example)

$$V_j = \begin{bmatrix} a & b & b & b \\ b & a & b & b \\ b & b & a & b \\ b & b & b & a \end{bmatrix}$$

where $a = \sigma_b^2 + \sigma^2$ and $b = \sigma_b^2$. Because this matrix has a lot of structure its inverse is relatively easy to figure it out by looking at small cases. In our example, we have

$$V_j^{-1} = \frac{1}{a(a+2b) - 3b^2} \begin{bmatrix} a+2b & -b & -b & -b \\ -b & a+2b & -b & -b \\ -b & -b & a+2b & -b \\ -b & -b & -b & a+2b \end{bmatrix},$$

which we can quickly verify by calculating a couple of entries for the matrix product $V_j^{-1} V_j$. For an $n \times n$ matrix the diagonal in the inverse matrix has the form $a + (n-2)b$ and the off-diagonal elements are all $-b$. The multiplicative constant in front of the matrix is equal to $\left[a(a+(n-2)b) - (n-1)b^2\right]^{-1}$.

Exercise

Verify that the matrix product $V_j^{-1} V_j$ is the identity matrix by calculating the $(1,1)$ and $(2,1)$ entries of this matrix and noting that all other entries would be equal to one of these two calculations.

Solution

For the $(1,1)$ entry we need to take the dot product of the first row of V_j^{-1} and the first column of V_j. Ignoring the scalar multiplier in front of V_j^{-1} we have

$$\begin{bmatrix} a+2b & -b & -b & -b \end{bmatrix} \begin{bmatrix} a \\ b \\ b \\ b \end{bmatrix} = a(a+2b) - 3b^2.$$

This is equal to the scalar multiplier in front of V_j^{-1} and so the $(1,1)$ entry of the product $V_j^{-1}V_j$ is equal to 1.
For the $(2,1)$ entry we need to calculate the dot product of the second row of V_j^{-1} and the first column of V_j;

$$\begin{bmatrix} -b & a+2b & -b & -b \end{bmatrix} \begin{bmatrix} a \\ b \\ b \\ b \end{bmatrix} = -ab + ab + 2b^2 - 2b^2 = 0.$$

Hence, the $(2,1)$ entry is zero.

Since the design matrix X_j is just a column of 1's in the balanced Bühlmann model the matrix product $X_j^t V_j^{-1}$ is equal to the $1 \times N$ matrix where each entry is the sum of a column of V_j^{-1}; that is, we have

$$X_j^t V_j^{-1} = \frac{\begin{bmatrix} a-b & a-b & \cdots & a-b \end{bmatrix}}{a(a+(N-2)b) - (N-1)b^2}$$

where the numerator is a vector of length N and the denominator is a scalar. Multiplying the above vector by X_j on the right, that is, summing up all the entries in the vector, we obtain the 1×1 matrix

$$X_j^t V_j^{-1} X_j = \frac{N(a-b)}{a(a+(N-2)b) - (N-1)b^2}.$$

Similarly, for the second term in Equation 4.1 we have the 1×1 matrix

$$X_j^t V_j^{-1} y_j = \frac{(a-b)(y_{j1} + y_{j2} + \cdots + y_{jN})}{a(a+(N-2)b) - (N-1)b^2}.$$

Finally, noting that the denominators are all the same we can sum these expressions across

$j = 1, 2, \ldots, J$ resulting in

$$\left(\sum_{j=1}^{J} X_j^t V_j^{-1} X_j\right)^{-1} \left(\sum_{j=1}^{J} X_j^t V_j^{-1} y_j\right) = \frac{a(a+(N-2)b) - (N-1)b^2}{JN(a-b)} \cdot \frac{(a-b)(y_{11} + y_{12} + \cdots + y_{JN})}{a(a+(N-2)b) - (N-1)b^2}$$
$$= \frac{y_{11} + y_{12} + \cdots + y_{JN}}{JN} = \bar{y}_{\cdot}$$

Therefore, we have that the generalized least squares estimator of β in the balanced Bühlmann model is equal the sample mean.

For the Bühlmann-Straub model we only have to make some minor changes to the specification in the linear mixed model we used in the balanced Bühlmann case. We allow an unequal number of observations, n_j, for each group and we need to incorporate weights, w_{jt}, for each observation so that larger weights result in a smaller within group variance; hence, we will set the $n_j \times n_j$ variance-covariance matrix R_j to be equal to

$$R_j = \begin{bmatrix} \frac{\sigma^2}{w_{jt}} & 0 & \cdots & 0 \\ 0 & \frac{\sigma^2}{w_{jt}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\sigma^2}{w_{jt}} \end{bmatrix}.$$

This is the same construction we would use when specifying a weighted least squares regression.

The design matrices X_j and Z_j both have dimension $n_j \times 1$ and all entries are equal to 1. The variance-covariance matrix D has dimension 1×1 and we write its single entry as σ_b^2. The parameters to be estimated are β and $\tau = (\sigma_b^2, \sigma^2)$.

To summarize, the linear mixed effects model with one level of grouping can be written as

$$Y_j = X_j\beta + Z_j b_j + \epsilon_j, \qquad j = 1, 2, \ldots, J,$$

where j is the grouping variable and n_j is the number of observations for the jth group. The fixed effects vector β has p components because the design matrices, X_j have p columns representing the explanatory variables. The matrices Z_j have dimension $n_j \times q$ as we have q explanatory variables for the random effects. The random vectors b_j and ϵ_j have dimension n_j and follow normal distributions $\mathcal{N}(0, D)$ and $\mathcal{N}(0, R_j)$, where the matrices D and R_j are the variance-covariance matrices with dimensions $q \times q$ and $n_j \times n_j$, respectively. These matrices must by symmetric and positive definite (otherwise they are not valid variance-covariance matrices).

4.4 Hachemeister's Regression Model Revisited

For the Hachemeister data we have the severity of claims over 12 quarters from a sample of 5 states. We have seen from Figure 3.5 that different intercepts and slopes for these states are a reasonable starting point and we would like to make inferences from the larger population of states that these 5 came from. Therefore, we will specify a linear mixed model where the linear predictor for the fixed effects has an intercept and the variable `time` and we use the same linear predictor for the random effects. This way we will get a random intercept and a random slope for each state.

```
hm.dta <- read_csv("hachemeister-data.csv",
                   col_types = "fidd")
```

A natural way to specify the linear mixed model in the `lmer()` function is as follows:

```
hm.mixed.1 <- lmer(severity ~ time + (time | state),
                   data = hm.dta,
                   weights = claims/1000)
```

```
boundary (singular) fit: see help('isSingular')
```

The message displayed regarding the boundary fit will be discussed after we present the results of the fit.

The first part of the right-hand side of the formula, in this case just `time`, represents the fixed effects and the second part, within parenthesis, that is `(time | state)`, is the random component. For both components we did not specify an intercept because R includes one by default. The vertical bar within the random component separates the specification for the linear predictor and the grouping variable, for this example, `state`. For the weights, we have divided the number of claims by 1,000 to keep the numbers in the calculations from getting too big and causing numerical difficulties in the estimation algorithm.

Notice that we have not provided any instructions on what kind of variance-covariance matrices D or R_j we want to use. The matrix D is of dimension 2×2 (the Z_j matrices have two columns: intercept and `time`) and the matrices R_j are of dimension 12×12 because for each `state` we have 12 quarterly observations. The default behavior for `lmer()` is to have R_j be a diagonal matrix with entries equal to σ^2/w_{jt}, since we specified a `weights` argument in the call. The matrix D will be a general 2×2 variance-covariance matrix. The $(1,1)$ position is the variance of the intercept, the $(2,2)$ position is the variance of the slope and the $(2,1)$ or $(1,2)$ positions is the covariance between intercept and slope. Thus for this model we will be estimating 2 fixed effects, β_0 and β_1, 3 entries for the matrix D, and the residual variance σ^2.

The summary of the above fitted model is provided below where you will see two sections; one labeled 'Random effects' and the other one 'Fixed effects.'

```
summary(hm.mixed.1)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: severity ~ time + (time | state)
   Data: hm.dta
Weights: claims/1000

REML criterion at convergence: 782.4

Scaled residuals:
    Min      1Q  Median      3Q     Max
-1.9751 -0.6266 -0.2336  0.6918  2.6742

Random effects:
 Groups   Name        Variance Std.Dev. Corr
 state    (Intercept) 11990.2  109.50
          time          553.2   23.52   1.00
 Residual             47599.0  218.17
Number of obs: 60, groups:  state, 5

Fixed effects:
            Estimate Std. Error t value
(Intercept)  1501.29      60.76  24.707
time           27.75      11.69   2.374

Correlation of Fixed Effects:
     (Intr)
time 0.541
optimizer (nloptwrap) convergence code: 0 (OK)
boundary (singular) fit: see help('isSingular')
```

The fixed effects section tells us, for this example, that the population average severity at time $t = 0$ is \$1,501.29 and for each additional quarter, average severity will increase by \$27.75. Also we have the standard errors of these coefficients and their t-values; that is, the estimate divided by its standard error. We have evidence in our data that the fixed effects are different from zero.

From the random effects section we have almost all the information for the remaining parameters. The column labeled `Variance` has the variances for the intercept, for the variable `time`,

and the residual variance $\hat{\sigma}^2 = 4.7599 \times 10^4$. Note that the column labeled `Std.Dev.` is *not the standard error of the estimated variances*. This column is just the square root of the entries in the `Variance` column. We do not have the estimated covariance between the intercept and `time`, but we can get this information by extracting the entire variance-covariance matrix D and the residual variance too.

```
print(VarCorr(hm.mixed.1), comp = "Variance")
```

```
 Groups   Name        Variance Cov
 state    (Intercept) 11990.16
          time          553.16  2575.368
 Residual             47598.96
```

The very last line of the summary output shows the following message: `boundary (singular) fit: see help('isSingular')`. This is tells us that during the optimization one of our parameters has reached its boundary.

We can compute the eigenvalues of the variance-covariance matrix D via

```
eigen(VarCorr(hm.mixed.1)$state)$value
```

```
[1] 1.254332e+04 1.136868e-13
```

and see that the second eigenvalue is essentially zero; hence, our matrix D is super close to not being positive definite. A positive definite matrix must have all of its eigenvalues positive. If at least one of them is zero, then the matrix is positive semi-definite.

This is an indication that our default choice of matrix D with three free parameters is not ideal. Therefore, for our next mixed model we will restrict the variance-covariance matrix D to be diagonal. To specify such a model we need to tell the `lmer()` function that we want independent random components for the intercept and the slope parameters even though both use the same grouping variable. We do this by specifying **two** random effects in the formula for `lmer()`. The first one gives us the random intercept and the second one the random slope. Note that we need to tell R explicitly not to include an intercept in the second random effect by using `0 + time`.

```
hm.mixed.2 <- lmer(severity ~ time + (1 | state) + (0 + time | state),
                   data = hm.dta,
                   weights = claims/1000)
summary(hm.mixed.2)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: severity ~ time + (1 | state) + (0 + time | state)
   Data: hm.dta
Weights: claims/1000

REML criterion at convergence: 785.5

Scaled residuals:
    Min      1Q  Median      3Q     Max
-1.8951 -0.6462 -0.2441  0.5449  2.6713

Random effects:
 Groups   Name        Variance Std.Dev.
 state    (Intercept) 19909.0  141.1
 state.1  time          605.1   24.6
 Residual             48723.8  220.7
Number of obs: 60, groups:  state, 5

Fixed effects:
            Estimate Std. Error t value
(Intercept)  1491.99      78.46  19.015
time           29.55      12.76   2.315

Correlation of Fixed Effects:
     (Intr)
time -0.248
```

From the random effects section of the summary output we see only estimates for the diagonal elements of D and the residual variance, σ^2. Note that the fixed effects are slightly different than those in our previous model, but for all practical purposes are the same. The estimated random effects, deviations from the fixed effects, for each state are

```
ranef(hm.mixed.2)$state
```

```
  (Intercept)      time
1  162.868302  32.78440
2  -78.554481 -13.24185
3   44.108233  12.01649
4 -137.982018 -16.88416
5    9.559964 -14.67488
```

and putting these together with the fixed effects we obtain the following credibility weighted values:

```
round(fixef(hm.mixed.2) + t(as.matrix(ranef(hm.mixed.2)$state)), 2)
```

```
                  1       2       3       4       5
(Intercept) 1654.86 1413.44 1536.10 1354.01 1501.55
time          62.34   16.31   41.57   12.67   14.88
```

These are not very different from the estimates we got in Chapter 3 and they suffer from the same issues that we noted there. Many of the intercepts and slopes are not between the individual states and collective estimates.

We cannot easily extract the credibility matrices from the fitted model, but (Edward W. Frees, Young, and Luo 1999, Table 1) provides an explicit formula to calculate these matrices from information we do readily have from the model. The formula is

$$A_j = \frac{\det(DW_j)I_2 + \hat{\sigma}^2 DW_j}{\det(DW_j) + \hat{\sigma}^2 \text{trace}(DW_j) + \hat{\sigma}^4}, \tag{4.2}$$

where I_2 is a 2×2 identity matrix, W_j is given by

$$W_j = \begin{bmatrix} \sum_{k=1}^{n_j} w_{jk} & \sum_{k=1}^{n_j} t_{jk} w_{jk} \\ \sum_{k=1}^{n_j} t_{jk} w_{jk} & \sum_{k=1}^{n_j} t_{jk}^2 w_{jk} \end{bmatrix},$$

'det' is the determinant of a matrix, and 'trace' is the sum of the diagonal elements of a square matrix. This formula for the credibility matrix A_j matches the formula we developed in Chapter 3. And as we saw in that chapter, the reason for credibility weighted estimates not to lie between the standalone and collective estimates is that the matrix A_j is not diagonal.

Exercise

Show that Equation 4.2 matches Equation 3.30 in the case where the variance-covariance matrix D is diagonal with entries τ_0^2 and τ_1^2.

Solution

Before looking at Appendix B for a derivation try it yourself. Start with the denominator in Equation 4.2 and then work on the numerator. The denominator is a just a number and the numerator is a 2×2 matrix. The calculations are straightforward but tedious.

We also saw in Chapter 3 that to achieve a diagonal credibility matrix we can center the time variable at each group's center of gravity. We can add a centered time variable, `ctime`, to our data set via

```
CG <- with(hm.dta,
           tapply(time * claims, state, sum) /
             tapply(claims, state, sum))
hm.dta$ctime <- hm.dta$time - CG[hm.dta$state]
rm(CG)
```

and fit the same linear mixed model by replacing `time` with `ctime`.

```
hm.mixed.3 <- lmer(severity ~ ctime + (1 | state) + (0 + ctime | state),
                   data = hm.dta,
                   weights = claims/1000)
summary(hm.mixed.3)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: severity ~ ctime + (1 | state) + (0 + ctime | state)
   Data: hm.dta
Weights: claims/1000

REML criterion at convergence: 788.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.0491 -0.7028 -0.1559  0.5252  2.6324

Random effects:
 Groups   Name        Variance Std.Dev.
 state    (Intercept) 70838.8  266.16
 state.1  ctime         446.4   21.13
 Residual             49019.8  221.40
Number of obs: 60, groups:  state, 5

Fixed effects:
            Estimate Std. Error t value
(Intercept)  1674.96     122.12  13.715
ctime          34.09      11.59   2.942

Correlation of Fixed Effects:
      (Intr)
ctime 0.000
```

Note that the fixed effect for the intercept has changed significantly because now we are measuring the mean severity at time approximately $t = 6.5$ instead of at time $t = 0$. Also

worth noting is the correlation of the fixed effects shown at the very bottom of the summary output. Now the intercept and the slope have a zero correlation whereas in our previous model it was -0.248. We expected this result because by centering the time variable we have made the intercept and the centered time variable orthogonal to each other.

The credibility weighted estimates from this model are

```
round(fixef(hm.mixed.3) + t(as.matrix(ranef(hm.mixed.3)$state)), 2)
```

```
                  1       2       3       4       5
(Intercept) 2058.27 1516.73 1799.56 1398.97 1601.24
ctime         60.02   22.45   39.63   32.08   16.27
```

which now lie between the stand alone and the collective estimates.

Now that we have an initial model, `hm.mixed.3`, we should check whether the distributional assumptions that we have made are valid for the data. There are two basic assumptions:

1. the within-group errors are independent and identically normally distributed with mean zero and variance σ^2/w_{jt}, and are independent of the random effects, and
2. the random effects are normally distributed with mean zero and variance-covariance matrix D that does not depend on the group and are independent for different groups.

Checking Within-Group Errors Assumptions

To assess the within-group residuals we can use a boxplot of standardized residuals by state. Figure 4.6 shows such a plot for the `hm.mixed.3` model where we have also included the residual points and a larger blue circle at the mean value of the residuals. The states have been arranged in order of the size of their interquartile range and we can see that there is an increasing spread. Even though all five groups of residuals are centered about zero their spread is not constant. This suggests that a single within-group variance, σ^2, parameter may not be correct and we might need to select a more general model where each group has its own parameter.

```
hm.dta$mu <- fitted(hm.mixed.3)
hm.dta$rPS <- resid(hm.mixed.3,
                    type = "pearson",
                    scaled = TRUE)
iqr <- tapply(hm.dta$rPS, hm.dta$state,
              function(x) diff(quantile(x, c(0.25, 0.75))))
o.iqr <- order(iqr, decreasing = TRUE)
hm.dta$state.iqr <- fct_relevel(hm.dta$state,
```

```
                                levels(hm.dta$state)[o.iqr])
rm(iqr, o.iqr)
```

```
ggplot(data = hm.dta,
       mapping = aes(x = rPS,
                     y = state.iqr)) +
  geom_vline(xintercept = 0, color = "gray") +
  geom_boxplot() +
  geom_jitter(height = 0.2,
              alpha = 0.3) +
  stat_summary(fun = mean,
               geom = "point", pch = 1,
               color = "blue", size = 2) +
  labs(x = "Standardized Pearson Residuals",
       y = "State")
```

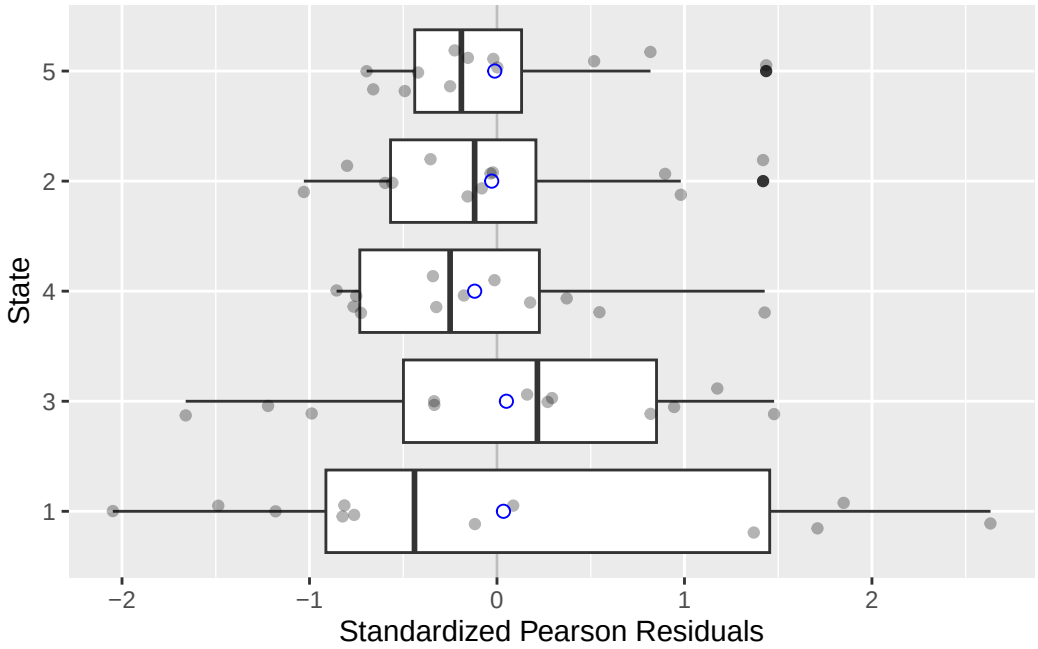


Figure 4.6: Boxplots and underlying data of standardized residuals for the `hm.mixed.3` model. The large blue circle is the average value of the residuals. The states have been ordered by the size of the interquartile range of their residuals.

```
p1 <- ggplot(data = hm.dta,
       mapping = aes(x = mu,
```

```
                      y = rPS,
       color = state)) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "Std. Residuals") +
  theme(legend.position = "none")

p2 <- ggplot(data = hm.dta,
       mapping = aes(x = mu,
                     y = abs(rPS))) +
  geom_point() +
  geom_smooth(method = "lm",
              se = FALSE) +
  labs(x = "Fitted Severity Values",
       y = "|Std. Residuals|") +
  theme(legend.position = "none")

p3 <- ggplot(data = hm.dta,
       mapping = aes(x = mu,
                     y = severity,
                     color = state)) +
  geom_abline(intercept = 0, slope = 1,
              color = "gray") +
  geom_point() +
  coord_cartesian(xlim = c(1000, 2500),
                  ylim = c(1000, 2500)) +
  labs(x = "Fitted Values",
       y = "Actual Values") +
  theme(legend.position = "none")

p4 <- ggplot(data = hm.dta,
       mapping = aes(sample = rPS)) +
  geom_qq() +
  geom_abline(slope = 1, intercept = 0) +
  labs(x = "Theoretical Quantiles",
       y = "Std. Residuals") +
  theme(legend.position = "none")

(p1 + p2) / (p3 + p4)
rm(list = ls(pattern = "p+"))
```

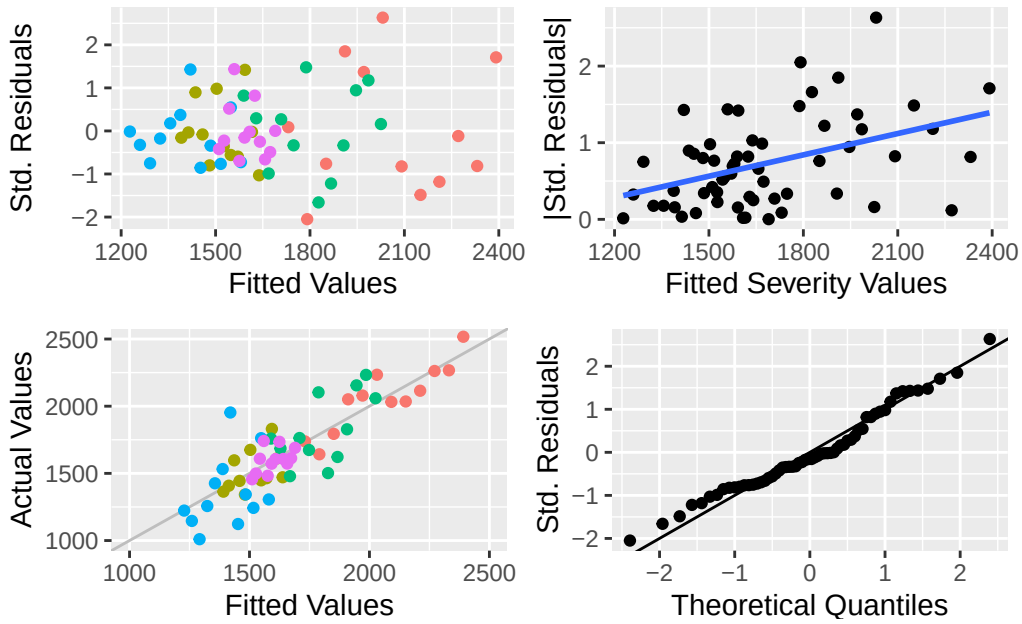


Figure 4.7: Diagnostic plots for the `hm.mixed.3` model.

Figure 4.7 shows other diagnostic plots to assess the assumptions about the within group residuals for the `hm.mixed.3` model. The top row panels confirm that the residuals do not have constant variability by state. The top-left plot shows a fanning out the residuals as the fitted values increase in size. The top-right panel displays the absolute value of the residuals against the fitted values and we have superimposed a least squares estimated trend line that clearly shows our residuals spreading out as the fitted values increase.

The bottom-left panel is an actual versus expected plot together with the line $y = x$. It looks like we have most points scattered around the line $y = x$. There is one possible outlier: the point with coordinates close to $(1400, 2000)$. In the bottom-right hand we have displayed a QQ-plot. We would like the points to be on the line $y = x$ and most of them do follow this pattern. But on the lower left corner, as the theoretical quantiles increase towards negative infinity, all the points are above the line $y = x$. This tells us that our data has a thinner left-hand tail compared to the normal distribution.

Checking Random Effects Assumptions

The second assumption we need to check is about the random effects. They should be normally distributed with a mean of zero and variance-covariance matrix D. For Hachemeister's data we only have five observations and so it will be difficult to draw definitive conclusions.

```
p1 <- ggplot(data = filter(tbl, term == "(Intercept)"),
             mapping = aes(x = estimate,
                           y = qn)) +
  geom_point() +
  labs(x = "Intercept Random Effects",
       y = "Std. Normal Quantiles")
p2 <- ggplot(data = filter(tbl, term == "ctime"),
             mapping = aes(x = estimate,
                           y = qn)) +
  geom_point() +
  labs(x = "ctime Random Effects",
       y = "Std. Normal Quantiles")
p1 + p2
rm(tbl, p1, p2)
```

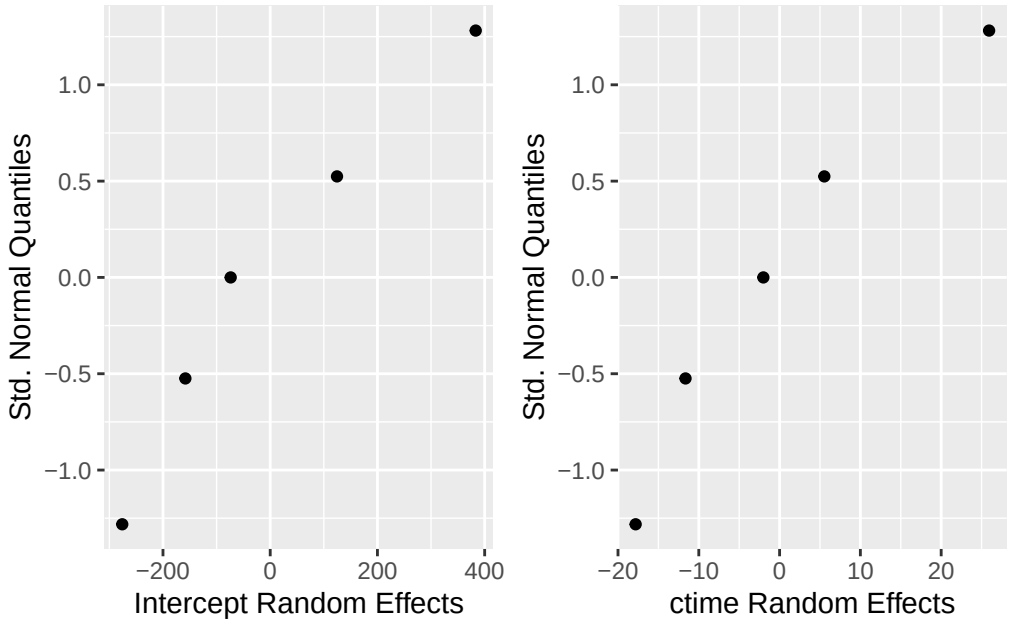


Figure 4.8: QQ-plots of the random effects for the Hachemeister `hm.mixed.3` model.

Figure 4.8 displays the QQ-plot for the random effects from model `hm.mixed.3`. We expect the points in these plots to lie along a straight line. In this case the assumption of normality seems reasonable. While the points in both panels are not perfectly on a line their departure is not excessive.

We should also check that the random effects follow a multivariate normal distribution with mean $\mu = (0, 0)$ and variance-covariance matrix D. The probability density function for our

two-dimensional example is

$$f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \frac{1}{2\pi\sqrt{\det(D)}} \exp\left(-\frac{1}{2}\begin{bmatrix} x_1 & x_2 \end{bmatrix} D^{-1} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right).$$

Note that the value of the density function f depends on x_1 and x_2 only through the value of the expression inside the exponential function; namely, through what is called the squared Mahalanobis distance:

$$d^2 = \begin{bmatrix} x_1 & x_2 \end{bmatrix} D^{-1} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = ax_1^2 + 2bx_1x_2 + cx_2^2,$$

if D^{-1} has diagonal entries equal to a and c and off-diagonal entries equal to b. The set of points (x_1, x_2) with Mahalanobis distance d^2 all have the same value of the density function f; that is, these points create a contour line in the 3-dimensional $(x_1, x_2, f([x_1, x_2]^t))$ surface of the density function.

In our example, the matrix D has been estimated as

$$D = \begin{bmatrix} 70{,}838.77 & 0 \\ 0 & 446.39 \end{bmatrix}.$$

Its inverse, D^{-1}, is also diagonal; hence, $b = 0$ and so we can see that the set of points (x_1, x_2) that have constant Mahalanobis distance form an ellipse whose major and minor axes fall along the x and y axes.

Figure 4.9 shows a scatter plot of the random effects for model `hm.mixed.3` along with ellipses at Mahalanobis distance of 1 and 2 standard deviations away from the origin. Note that the five points we have available are all within two standard deviations.

```
ggplot(data = tbs,
       mapping = aes(x = x,
                     group = gp)) +
  geom_line(mapping = aes(y = y.up), color = "gray") +
  geom_line(mapping = aes(y = y.dn), color = "gray") +
  geom_point(data = bind_cols(gp = "z",
                              ranef(hm.mixed.3)$state),
             mapping = aes(x = `(Intercept)`,
                           y = ctime,
                           gp = NULL)) +
  labs(x = "(Intercept)",
       y = "Centered Time")
rm(tbs)
```

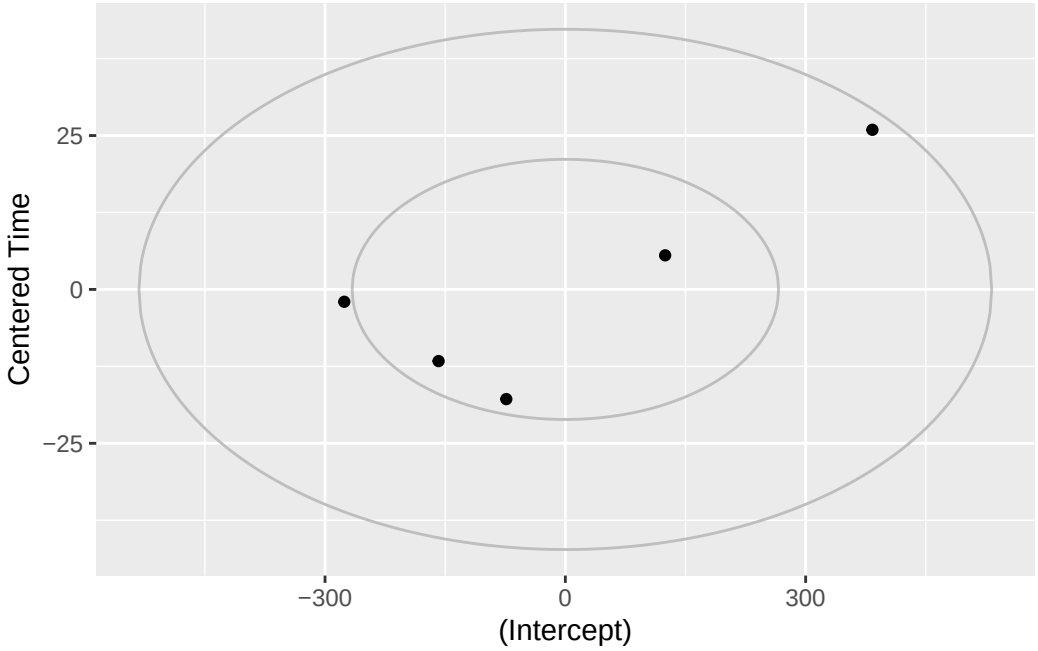


Figure 4.9: Scatter plot of estimated random effects for model `hm.mixed.3` along with contour lines that are 1 and 2 standard deviations away from the origin.

If the variance-covariance matrix D had not been diagonal, then we would still have elliptical contours, but they would have been rotated around the origin (the mean of our bivariate normal distribution is $(0, 0)$).

Note that in Figure 4.9 we chose to display the set of points that are Mahalanobis distance of 1 and 2 away from the origin because we are all very familiar that, in the 1-dimensional standard normal distribution, within these distances we have about 68% and 95% of the total density. For a 2-dimensional normal distribution these values do not give us the same proportion of the total density. To find the appropriate values we need to know that the squared Mahalanobis distance, d^2, has a chi-squared distribution with p degrees of freedom, where p is the dimension of the multivariate normal distribution.

In our case, $p = 2$ and if we are looking to obtain the same 68% and 95% coverage, then we must choose the Mahalanobis distance equal to the following values

```
crit.points <- sqrt(qchisq(c(0.68, 0.95), df = 2))
names(crit.points) <- c("68%", "95%")
crit.points
```

```
     68%      95%
1.509592 2.447747
```

Exercise

Using the `mvtnorm` package generate 2,000 multivariate random points with mean $\mu = (0, 0)$ and variance-covariance matrix

$$D = \begin{bmatrix} 70838 & 0 \\ 0 & 446 \end{bmatrix}.$$

Plot the points using three different colors depending on whether the points are within Mahalanobis distance 1, between 1 and 2, and beyond 2. Check that the proportion of points are not 68% or 95% for being within Mahalanobis distance 1 or 2 of the origin.

Solution

Let us set up the mean vector $\mu = (0, 0)$ and the variance-covariance matrix D.

```
N <- 2000
mu <- c(0,0)
D <- matrix(c(70838, 0, 0, 446),
            nrow = 2, ncol = 2)
```

Using the `rmvnorm()` function we can simulate the points via

```
set.seed(12837)
z <- rmvnorm(N, mean = mu, sigma = D)
```

The object `z` will be a $50{,}000 \times 2$ matrix where each row is one simulated point. Next we want to compute the Mahalanobis distance for each row, z_i, using the formula

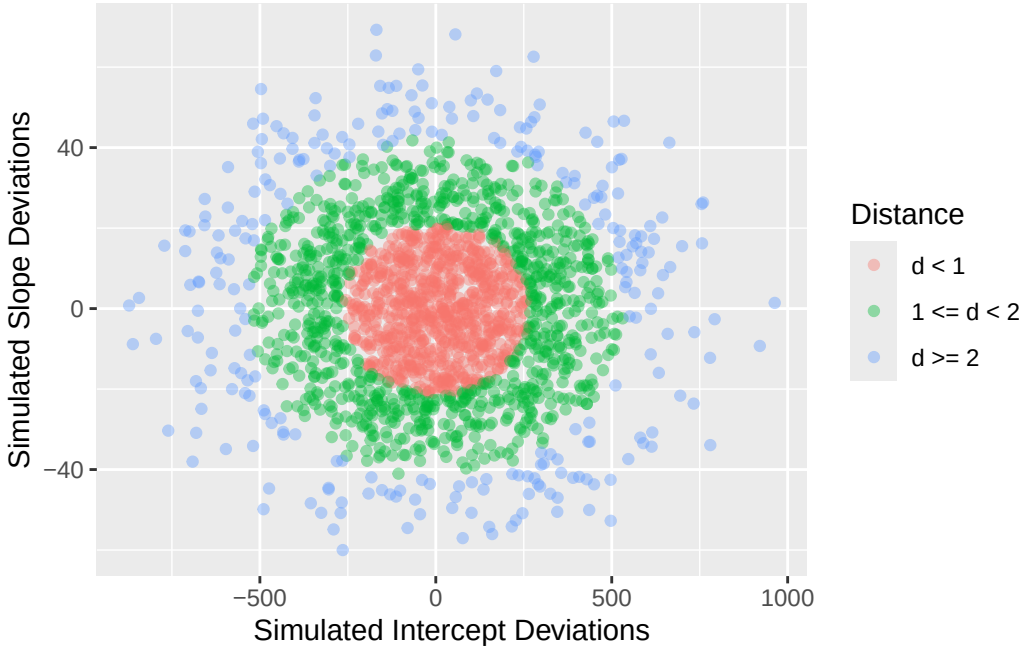
$$\sqrt{(z_i - \mu)^t D^{-1} (z_i - \mu)}.$$

We can do this by using the `apply()` function and we also need to create a categorical variable to distinguish which points are at different Mahalanobis distances. We will put all of this into a data frame

```
tb <- as.data.frame(z)
names(tb) <- c("x", "y")
tb$md <- apply(z, 1, function(z) sqrt(t(z - mu) %*% solve(D) %*% (z - mu)))
tb$md.bin.1 <- cut(tb$md,
                   breaks = c(-Inf, 1, 2, Inf),
                   labels = c("d < 1", "1 <= d < 2", "d >= 2"))
tb$md.bin.2 <- cut(tb$md,
                   breaks = c(-Inf, crit.points, Inf),
                   labels =c("d < 1.509", "1.509 <= d < 2.448", "d >= 2.448"))
```

and create the scatter plot.

```
ggplot(data = tb,
       mapping = aes(x = x,
                     y = y,
                     color = md.bin.1)) +
  geom_point(alpha = 0.4) +
  labs(x = "Simulated Intercept Deviations",
       y = "Simulated Slope Deviations",
       color = "Distance")
```



The proportion of points in each colored region is given by

```
xtabs( ~ md.bin.1, data = tb) / N
```

```
md.bin.1
     d < 1 1 <= d < 2     d >= 2
     0.395      0.471      0.134
```

Note how we only have about 40% of the points within Mahalanobis distance of 1 and about 86.6% of the points within distance 2. If we use the correct thresholds given by the chi-squared distribution; namely, 1.509 and 2.448, then we would have the appropriate coverage as shown next.

```
xtabs( ~ md.bin.2, data = tb) / N
```

```
md.bin.2
         d < 1.509 1.509 <= d < 2.448          d >= 2.448
            0.6940              0.2495              0.0565
```

4.5 Summary

In this chapter we revisited the three classic credibility models

- Balanced Bühlmann model,
- Bühlmann-Straub model, and
- Hachemeister's credibility regression model

and expressed them in terms of the well developed branch of statistics known as linear mixed models. For the practicing actuary embracing linear mixed models to implement credibility techniques brings substantial benefits. By using this theory can bring all the machinery that statisticians have developed to bear on our applications and apply standard software to carry out the necessary computations. We also have at our disposal inference techniques and model checking procedures. More importantly linear mixed models allow us to capture the correlation that exists in many of our data sets and we have many models to choose from.

We looked closely at linear mixed models with one level of grouping and introduced the concepts of fixed effects and random effects. One way of thinking about these effects, but perhaps not a very good one (as pointed out in (Gelman and Hill 2007, 245)), is that fixed effects estimate features of the population from which our sample was taken and random effects for those variables whose values are just a sample of the possible values the population has.

One clear disadvantage of the linear mixed model is that the random effects and the response variable must be normally distributed. This restriction is a serious one for actuarial work but in the next chapter we will introduce generalized linear mixed models. This class of statistical models expands the well known framework of generalized linear models that many actuaries use to include random effects with distributions from the exponential family. With this expanded set of models actuaries can significantly increase their modeling capabilities.

5 Generalized Linear Mixed Models

In the previous chapter we introduced linear mixed models and saw how three classical credibility models are special cases of the linear mixed model theory. Linear mixed models are characterized by having a response variable that is normally distributed and by having random effects that are also normally distributed. This model is an extension of the classical ordinary least squares model and applicable in many situations. But we know that real-world data is richer and more complex than the normal distribution can accommodate. The extension from the classical linear model to the generalized linear model, where the response distribution is a member of the exponential family, opened up a new area of techniques and tools well suited to the data and problems that actuaries encounter in practice.

Over the last two decades, actuaries have made good use of the generalized linear model. The next step in expanding these model to more complex data structures is to introduce random effects into the framework of generalized linear models and allow these random effects to have other distributions besides the normal.

Also another important extension is focused on the dispersion parameter. Generalized linear model theory keeps the dispersion parameter fixed across all observations. From experience we know that such a fixed parameter is not always ideal. It would be helpful to link the dispersion parameter to some explanatory variables.

In this chapter we introduce an extension of GLMs known as *hierarchical generalized linear models* that will allow us to have random effects whose distributions come from a broader family and to model the dispersion parameter via explanatory variables. These models are based on the theory of h-likelihood which brings together both Bayesian and Frequentist perspectives.

In the next section we give a brief conceptual introduction to the hierarchical generalized linear model without delving too much in the theory. Then we will present several examples on how to use these new models. Our discussion follows closely the work of Youngjo Lee, Nelder, and Pawitan (2021) and LEE (2020).

5.1 Hierarchical Generalized Linear Models

Hierarchical generalized linear models were introduced in 1996 (Y. Lee and Nelder 1996). These models use a generalization of the likelihood function called *hierarchical* or h-likelihood.

The maximization of this extended likelihood, under appropriate conditions, gives estimates of both fixed as well as random effects and the dispersion parameter.

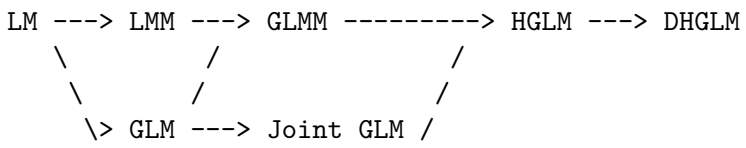
Starting with the linear model researchers worked in two directions to expand it. The first path created the linear mixed effects model (LMM) where the linear predictor can have random terms which are normally distributed. The second path worked on introducing an expanded list of response distributions giving us the generalized linear model (GLM). The combination of these two yields the generalized linear mixed model (GLMM) with distributions for the response variable from GLM and random effects in the linear predictor from LMM.

Also from the GLM framework, practitioners and researchers worked at enhancing the capabilities in modeling the variance of the response variable. GLMs use a single multiplier, ϕ, the dispersion parameter to scale the variance function. In many situations, this single parameter is not adequate to capture the volatility of the response and so a link function and a linear predictor were introduced for the dispersion parameter giving rise to *joint GLMs*.

Combining generalized linear mixed models (GLMM) and joint GLMs and expanding the distributional assumptions for the random effects brings us to the *hierarchical generalized linear model* (HGLM). These models have a response variable whose distribution comes from the exponential family. The linear predictor has fixed and random effects and these random effects are not constrained to be normally distributed. The dispersion parameter can be modeled via a separate link-function tied to a different linear predictor with fixed effects.

And finally we have *double hierarchical generalized linear models* (DHGLM) where we take a HGLM and allow random effects in the dispersion model and can also introduce explanatory variables via a link-function and a linear predictor into the variance of the random effects.

The following diagram gives a crude representation of how these various models are interconnected (adapted from (LEE 2020, 3)).



To help us translate between the mathematical description of a model and the R code necessary to implement the model consider the following mixed model

$$g(\mathbb{E}[y]) = X\beta + Zv,$$

where y is the response variable, $g()$ is the link function for the mean, β represents the fixed effects and v are the random effects. The matrices X and Z are the design matrices for the fixed and random effects, respectively. We also need to specify the distribution of the response variable (a member of the exponential family) and the distribution of the random effects; that

is, $v \sim D(\lambda)$, where D just stands for a distribution such as the Gaussian or gamma with parameter vector λ.

So far we have only described a model for the mean. If we are also modeling the dispersion, ϕ, parameter then we would also have

$$h(\mathbb{E}[\phi]) = W\gamma + Mu,$$

where h is a link function, γ are fixed effects, u are random effects, and W and M are design matrices. Since the u are random we also have to specify their distribution which will come with some parameters.

The full specification of a model can be complex, but using the notation introduced in by LEE (2020) in chapter 6 makes things more manageable. A double hierarchical generalized linear model (DHGLM) model is represented by a pair

$$\{\text{model}(\mu), \text{model}(\phi)\},$$

where the first entry is the model for the *mean* and the second entry represents the model for the *dispersion*.

For example, the usual generalized linear model would be written an $\{\text{GLM}(\mu), \text{constant}\}$. A joint model would we written as $\{\text{GLM}(\mu), \text{GLM}(\phi)\}$, where we have two regular GLM models that are interlinked to form the joint model. If we want to include a random effect into the model for the mean, we can write it as $\{\text{HGLM}(\mu), \text{constant}\}$ and if we also want to have the dispersion parameter modeled then we would say $\{\text{HGLM}(\mu), \text{GLM}(\phi)\}$.

There are two R packages to fit these models; `hglm` and `dhglm`. We'll use the first one briefly when we revisit the Hachemeister data because it allows us to use nearly the same calling code as we did in the previous chapter. But we will mostly use the `dhglm` package.

The `dhglm` package uses two functions to fit a model. The first one, `DHGLMMODELING`, creates the appropriate structures for the mean and dispersion models. The second one, `dhglmfit` does the actual computations. As an example of their use, a standard GLM model $\{\text{GLM}(\mu), \text{constant}\}$ would be specified and fitted as follows:

```
model.mu <-  DHGLMMODELING(Model = "mean",
                           Link = "log",
                           LinPred = count ~ age + gender)
model.phi <- DHGLMMODELING(Model = "dispersion")

fit <- dhglmfit(RespDist = "poisson",
                DataMain = accidents,
                MeanModel = model.mu,
                DispersionModel = model.phi)
```

Where we are fitting a Poisson model to a response variable `count` using explanatory variables `age` and `gender`. These variables live in a dataset called `accidents`.

5.2 Examples

In this section we present four examples to get familiar with specifying and fitting models with the `hglm` and `dhglm` packages:

1. Hachemeister
2. Fabric Faults
3. Train Accidents
4. Diabetes Progression

Quick Revisit with the Hachemeister Data

In the last chapter we fitted several linear mixed models to the Hachemeister data. Here we will re-fit the last model `hm.mixed.3` using the machinery from hierarchical generalized linear models and compare results. The main function to fit HGLMs is `hglm2()` and can be found in the `hglm` package.

Model `hm.mixed.3` used a centered version of time called `ctime`; that is, a weighted average of time where the weights are the number of claims. Let's load our data and compute `ctime`.

```
hm.dta <- read_csv("hachemeister-data.csv",
                   col_types = "fidd")
CG <- with(hm.dta,
           tapply(time * claims, state, sum) /
             tapply(claims, state, sum))
hm.dta$ctime <- hm.dta$time - CG[hm.dta$state]
rm(CG)
```

Using the `hglm2()` we specify the model in the same way as before. The response variable is `severity` and we have fixed effects for the intercept and the time variable `ctime`. We also include uncorrelated random effects for the intercept via `(1 | state)` and time `(0 + ctime | state)`. Both of these random effects vary by state. The response distribution is specified to be normally distributed with an identity link function through the `family` parameter. The random effects are also normally distributed with an identity link function via the `rand.family` parameter.

```
hm.hglm.3 <- hglm2(severity ~ ctime + (1 | state) + (0 + ctime | state),
                   data = hm.dta,
                   family = gaussian(link = "identity"),
                   rand.family = gaussian(link = "identity"),
                   weights = claims)
```

The summary output from the fitting process contains two major sections: one for the mean model and the other for the dispersion model. Our model did non specify any structure for the dispersion model and so it is taken to be a single parameter.

```
summary(hm.hglm.3)
```

```
Call:
hglm2.formula(meanmodel = severity ~ ctime + (1 | state) + (0 +
    ctime | state), data = hm.dta, family = gaussian(link = "identity"),
    rand.family = gaussian(link = "identity"), weights = claims)

----------
MEAN MODEL
----------

Summary of the fixed effects estimates:

            Estimate Std. Error t-value Pr(>|t|)
(Intercept)  1674.93     122.28  13.697  < 2e-16 ***
ctime          34.09      11.58   2.944  0.00484 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Note: P-values are based on 52 degrees of freedom

Summary of the random effects estimates:

                     Estimate Std. Error
(Intercept)| state:1  383.3472   123.4313
(Intercept)| state:2 -158.2215   127.8617
(Intercept)| state:3  124.6477   130.2105
(Intercept)| state:4 -276.0770   145.3201
(Intercept)| state:5  -73.6963   125.4177

Summary of the random effects estimates:

               Estimate Std. Error
ctime| state:1  25.9279    12.2453
ctime| state:2 -11.6410    14.2447
ctime| state:3   5.5344    15.0492
ctime| state:4  -2.0069    17.8084
ctime| state:5 -17.8144    13.2125
```

```
----------------
DISPERSION MODEL
----------------

NOTE: h-likelihood estimates through EQL can be biased.

Dispersion parameter for the mean model:
[1] 49017713

Model estimates for the dispersion term:

Link = log

Effects:
  Estimate Std. Error
   17.7077     0.1969

Dispersion = 1 is used in Gamma model on deviances to calculate the standard error(s).

Dispersion parameter for the random effects:
[1] 70873.4   446.5

Dispersion model for the random effects:

Link = log

Effects:
.|Random1
  Estimate Std. Error
   11.1687     0.7257

.|Random2
  Estimate Std. Error
    6.1013     0.8774

Dispersion = 1 is used in Gamma model on deviances to calculate the standard error(s).

EQL estimation converged in 3 iterations.
```

The fixed effects estimates are shown first, followed by the random effects for the intercept and then the random effects for the predictor variable `ctime`.

Putting together the estimated intercepts and slopes (fixed effects plus random effects) from

the model above, `hm.hglm.3`, we obtain

```
ans <- hm.hglm.3$fixef + matrix(hm.hglm.3$ranef, nrow = 2, ncol = 5, byrow = TRUE)
dimnames(ans) <- list(c("(Intercept)", "ctime"),
                      1:5)
round(ans, 3)
```

```
                  1        2        3        4        5
(Intercept) 2058.280 1516.712 1799.581 1398.856 1601.237
ctime         60.019   22.450   39.625   32.084   16.277
```

While the estimates we obtained in the previous chapter based on the generalized linear mixed model, `hm.mixed.3`, are

```
round(fixef(hm.mixed.3) + t(as.matrix(ranef(hm.mixed.3)$state)), 3)
```

```
                  1        2        3        4        5
(Intercept) 2058.273 1516.728 1799.565 1398.973 1601.241
ctime         60.021   22.445   39.627   32.082   16.272
```

Comparing them they are virtually identical. Also other estimated quantities such as the variances for the random effects are very close to each other. The residual variance for model `hm.mixed.3` is equal to 49019.8 and for our hierarchical model `hm.hglm.3` it is 49018.3. In model `hm.mixed.3` the variance for the intercept and `ctime` is 70838.8 and 446.4, respectively. For the hierarchical model,`hm.hglm.3`, we have 70873.4 and 446.5. Again close to each other.

Textile Fabric Defects

In this example we consider data set `fabric` from (Bissell 1972) where we will be investigating the number of faults in rolls of textile fabric. This data set is available in the `mdhglm` package and it has three variables and 32 observations.

This data set has also been analyzed in (LEE 2020). The response variable in the number of faults and so a natural choice would be to use the Poisson distribution. The only predictor variable is the length of the roll of fabric. Figure 5.1 shows that there is a relationship between the response and our predictor variable and that this relationship is not linear. As the length of a roll of fabric increases we see an increasing number of defects. If we transform both the response and predictor variables with a logarithmic function (not shown here), then the relationship between them seems linear.

```
ggplot(data = fabric,
       mapping = aes(x = x,
                     y = y)) +
  geom_point() +
  labs(x = "Length of Roll (in m)",
       y = "Number of Faults")
```

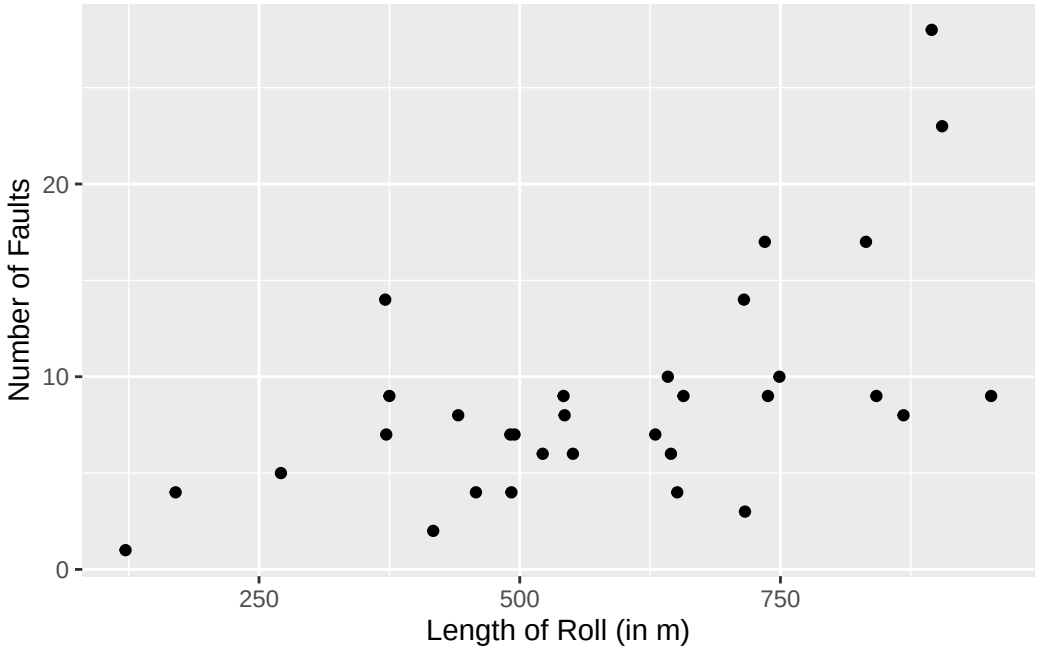


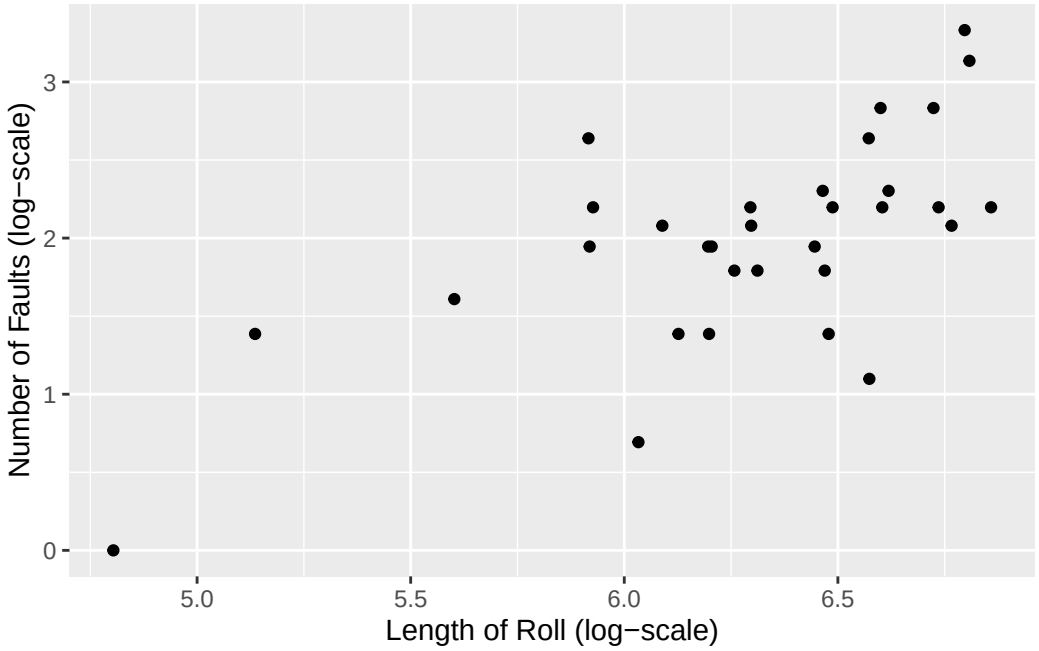
Figure 5.1: Number of faults in a roll of fabric.

Exercise

Transform both response and predictor variables by applying a logarithmic function and plot them. Does the relationship seem linear?

Solution

```
ggplot(data = fabric,
       mapping = aes(x = log(x),
                     y = log(y))) +
  geom_point() +
  labs(x = "Length of Roll (log-scale)",
       y = "Number of Faults (log-scale)")
```



While there are several points that do not fall close to a straight line pattern, the overall impression is that these points are indeed closer to a linear pattern than the original data.

Therefore, a reasonable starting point would be to use a Poisson generalized linear model with a log-link and the logarithm of the length of a roll (`x.lg`) as our predictor variable; that is, we want to fit the following model:

$$\log\left(\mathbb{E}[y_i]\right) = \beta_0 + \beta_1 \log(x_i),$$

where y_i is the number of faults and x_i is the length of the roll of fabric. Fitting such a model yields:

```
fab.poi.glm <- glm(y ~ x.lg,
                   data = fabric,
                   family = poisson(link = "log"))
summary(fab.poi.glm)
```

```
Call:
glm(formula = y ~ x.lg, family = poisson(link = "log"), data = fabric)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)  -4.1730     1.1352  -3.676 0.000237 ***
x.lg          0.9969     0.1759   5.668 1.45e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 103.714  on 31  degrees of freedom
Residual deviance:  64.537  on 30  degrees of freedom
AIC: 191.84

Number of Fisher Scoring iterations: 4
```

Even though both the intercept and the coefficient for the logarithm of the length of a roll of fabric are statistically significant, the model fits very poorly. The residual deviance of 64.5 is extremely large compared with residual degrees of freedom of 30 and we have a clear indication of overdispersion. Perhaps we have misspecified the linear predictor, but given that we only have one variable to work with, there is not much we can do about it. Another reason for the lack of fit could be that our choice of link function (logarithm) is not correct. But we do have some evidence that a log-link function is suitable. Hence, we conclude that the Poisson distribution is not adequate for this data and we move on to considering the negative binomial distribution.

We know that a negative binomial distribution arises as a mixture of the Poisson and gamma distributions as follows: let u be an unobserved gamma random variable with mean equal to 1 and variance equal to $1/\theta$ and conditionally on u, let Y be a Poisson random variable with mean equal to $\lambda \cdot u$. Then the marginal distribution of Y will be negative binomial. Fitting a negative binomial distribution GLM yields

```r
fab.nb.glm <- glm.nb(y ~ x.lg,
                     data = fabric)
summary(fab.nb.glm)
```

```
Call:
glm.nb(formula = y ~ x.lg, data = fabric, init.theta = 8.667407437,
    link = log)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -3.7951     1.4577  -2.603  0.00923 **
x.lg          0.9378     0.2280   4.114 3.89e-05 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(8.6674) family taken to be 1)

    Null deviance: 50.28  on 31  degrees of freedom
Residual deviance: 30.67  on 30  degrees of freedom
AIC: 181.39

Number of Fisher Scoring iterations: 1

              Theta:  8.67
          Std. Err.:  4.17

 2 x log-likelihood:  -175.387
```

Note that the estimated coefficients of this model do not differ significantly from those in the Poisson model and for this negative binomial model we do not have any evidence of lack of fit. The residual deviance is very close to the residual degrees of freedom. Of course, other diagnostics are needed to fully check the adequacy of this model.

Exercise

Use the following diagnostic plots to assess the adequacy of the negative binomial model.

1. Quantile Residuals vs. Fitted Values
2. Absolute value of Quantile Residuals vs. Fitted Values
3. Quantile Residuals vs. Predictor Variable
4. Linear Predictor vs. Working Responses

Solution

Let us compute the quantities we need for the diagnostic plots:

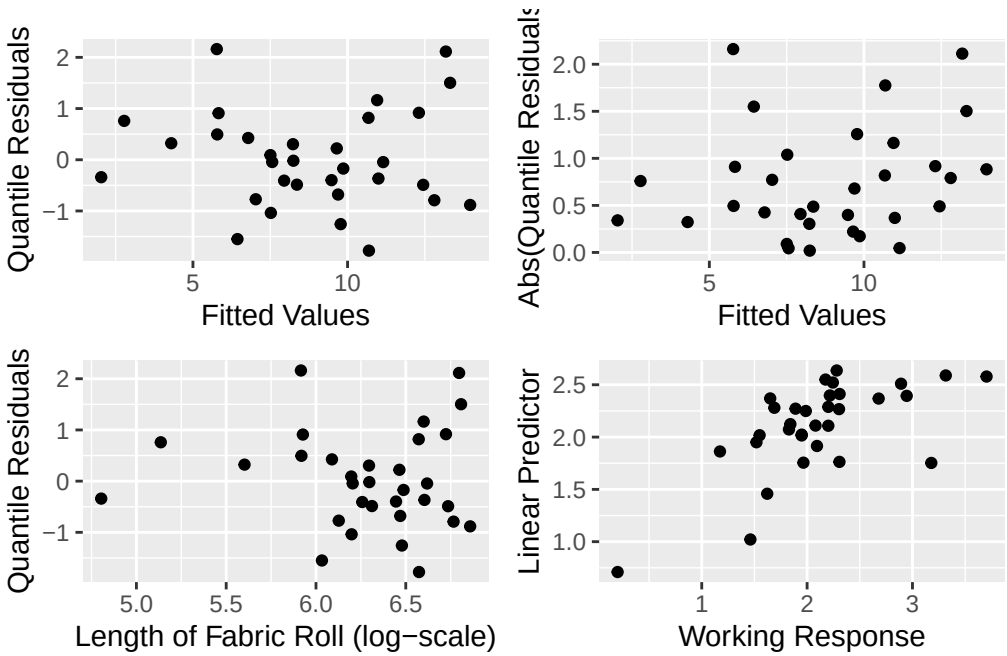
```
fabric.res <- fabric |>
  mutate(eta = predict(fab.nb.glm, type = "link"),
         mu  = predict(fab.nb.glm, type = "response"),
         rQ  = qresid(fab.nb.glm),
         rW  = resid(fab.nb.glm, type = "working"),
         wR  = rW + eta)
```

Compute the individual plots.

```
p1 <- ggplot(data = fabric.res,
             mapping = aes(x = mu,
                           y = rQ)) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "Quantile Residuals")
p2 <- ggplot(data = fabric.res,
             mapping = aes(x = mu,
                           y = abs(rQ))) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "Abs(Quantile Residuals)")
p3 <- ggplot(data = fabric.res,
             mapping = aes(x = x.lg,
                           y = rQ)) +
  geom_point() +
  labs(x = "Length of Fabric Roll (log-scale)",
       y = "Quantile Residuals")
p4 <- ggplot(data = fabric.res,
             mapping = aes(x = wR,
                           y = eta)) +
  geom_point() +
  labs(x = "Working Response",
       y = "Linear Predictor")
```

And arrange them in a 2×2 grid.

```
(p1 + p2) / (p3 + p4)
```



Both left-hand panels should display a random cloud of points. If there are any patterns in these plots, then our model would not be adequate. For the upper right panel, we would like to see a constant even spread of points across the y-axis. Any systematic increase or decrease would be an indication that our variance function is not correct. For the last panel in the bottom right-hand corner, the ideal pattern would be to have all points line up along the line $y = x$. Departures from this pattern would be an informal indication that the link function is not correct (or that we have misspecified the linear predictor).

Since the estimated value of θ in the negative binomial model is 8.67 we know that the variance of the gamma distribution is 1/8.67. Figure 5.2 displays what the estimated density function for the random effect looks like.

```
ggplot(data = tibble(x = seq(0.1, 2.2, length = 500),
                     y = dgamma(x, shape = 8.67, scale = 1/8.67)),
       mapping = aes(x = x,
                     y = y)) +
  geom_line() +
  labs(x = "Unobserved Random Variable",
       y = "Density")
```

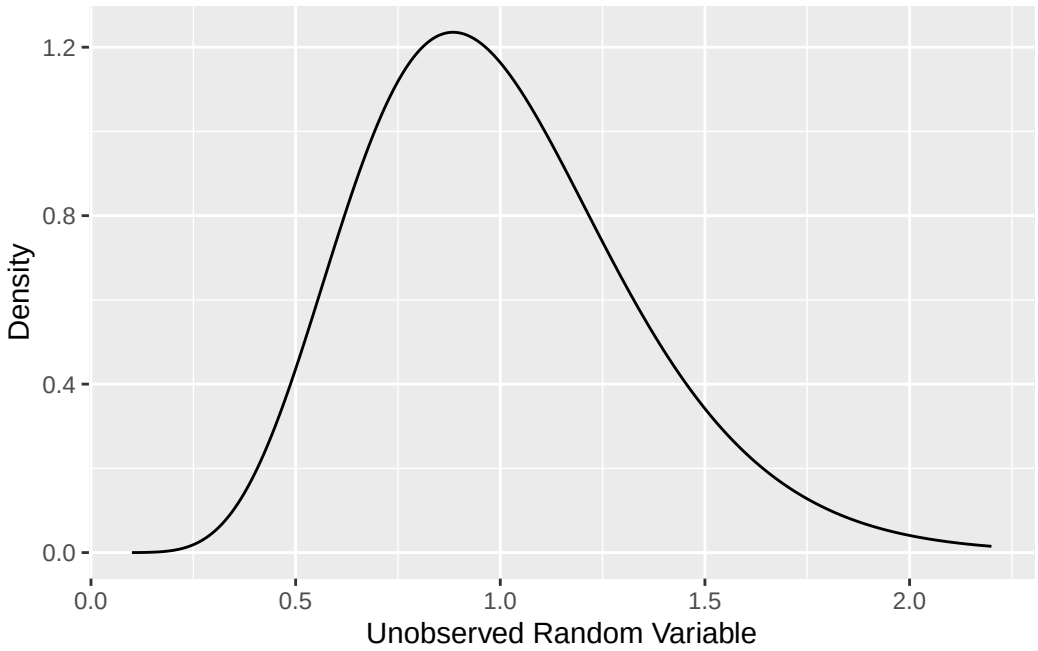


Figure 5.2: The density function for the random effect u.

We can also view the above model as linear mixed model. The mean of the Poisson distribution is $\lambda \cdot u$ and we would like to introduce explanatory variables. Hence, using a logarithmic link function we set

$$\log(\lambda \cdot u) = \log(\lambda) + \log(u) = X\beta + v,$$

where $X\beta$ incorporates all of our fixed effects explanatory variables and $v = log(u)$ is the unobserved random effect.

For the fabric data we can fit such a model by specifying the structure of:

1. the mean model
2. the dispersion model

For now we will keep the dispersion model to be a single constant (just like we always do when we fit a GLM).

The mean model is

```
model.mu <- DHGLMMODELING(Model = "mean",
                          Link = "log",
                          LinPred = y ~ x.lg + (1 | rf),
                          RandDist = "gamma")
```

Note that we have chosen a gamma distribution for the random effect.

The dispersion model is just a constant so we do not specify any components:

```
model.phi <- DHGLMMODELING(Model = "dispersion")
```

We fit this model via

```
fab.hglm.nb <- dhglmfit(RespDist = "poisson",
                        DataMain = fabric,
                        MeanModel = model.mu,
                        DispersionModel = model.phi)
```

```
Distribution of Main Response :
                        "poisson"
[1] "Estimates from the model(mu)"
y ~ x.lg + (1 | rf)
[1] "log"
            Estimate Std. Error t-value
(Intercept)  -3.9195     1.4442  -2.714
x.lg          0.9624     0.2259   4.261
[1] "Estimates for logarithm of lambda=var(u_mu)"
[1] "gamma"
   Estimate Std. Error t-value
rf   -2.074     0.3623  -5.726
[1] "========== Likelihood Function Values and Condition AIC =========="
                                        [,1]
-2ML (-2 p_v(mu) (h))          :  175.72501
-2RL (-2 p_beta(mu),v(mu) (h)) :  179.88494
cAIC                           :  172.77209
Scaled Deviance                :   14.28605
df                             :   14.40607
```

Here the estimated fixed effects are close to those reported for the negative binomial model. The variance of the random effect, -2.074 is reported on a logarithmic scale and exponentiating its value we arrive at a similarly sized parameter (0.125682).

Figure 5.3 reveals that while our model may be adequate there are some areas of concern. The display shows the studentized deviance residuals. The top left panel shows that for fitted values greater than about 12.5, we have a group of observations with positive and increasing residuals only. And below 7.5 there seems to be a larger number of observations with negative residuals than positive ones. The top-right panel shows the absolute value of studentized

residuals versus fitted values. Ideally, there would be no underlying trend in the residuals, but the graph shows that as fitted values increase, residuals first decrease and then increase. The QQ-plot, in the bottom left-panel, shows the expected pattern and the right-hand bottom panel also shows a reasonable histogram for the residuals.

```
df <- tibble(mu = as.numeric(fab.hglm.nb[["mu"]]),
             srD = as.numeric(fab.hglm.nb[["mean_residual"]]))
p1 <- ggplot(data = df,
             mapping = aes(x = mu,
                           y = srD)) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "Stu. Dev. Residuals")
p2 <- ggplot(data = df,
             mapping = aes(x = mu,
                           y = abs(srD))) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "|Stu. Dev. Residuals|")
p3 <- ggplot(data = df,
             mapping = aes(sample = srD)) +
  geom_qq() +
  labs(x = "Theoretical Quantiles",
       y = "Sample Quantiles")
p4 <- ggplot(data = df,
             mapping = aes(srD)) +
  geom_histogram(bins = 5) +
  labs(x = "Stu. Dev. Residuals",
       y = "Frequency")
```

```
(p1 + p2) / (p3 + p4)
```

Figure 5.3: Diagnostic plots from a Poisson-gamma hierarchical generalized linear model fitted to the fabric data. The top panels show that the model has some deficiencies beacuase there are discernible patterns. The bottom panels show the expected patterns.

Train Accident

In this example we analyze a dataset from (Agresti 2002) regarding the number of collisions involving British Rail passenger trains and road vehicles between 1975 and 2003. The available variables are the number of annual collisions (`y`) between trains and road vehicles, the distance traveled per year (`t`) in millions of kilometers, the number of years (`x`) since 1975, and an identification (`id`) label for each row of data. The data has also been analyzed in (LEE 2020) and we follow their discussion closely. The data is available under the name `train` from the `mdhglm` package.

We are interested in understanding the rate of accidents per million kilometers traveled. A scatterplot (not displayed here) shows a non-linear decreasing trend for the rate of collisions as time increases, but a logarithmic transformation of the response variable shows (see Figure 5.4) a linear trend with substantial variability around it.

```
ggplot(data = train,
       mapping = aes(x = x + 1975,
                     y = log(y/t))) +
  geom_point() +
```

```
  labs(x = "Calendar Year",
       y = "Rate of Collission (per M km, log-scale)")
```

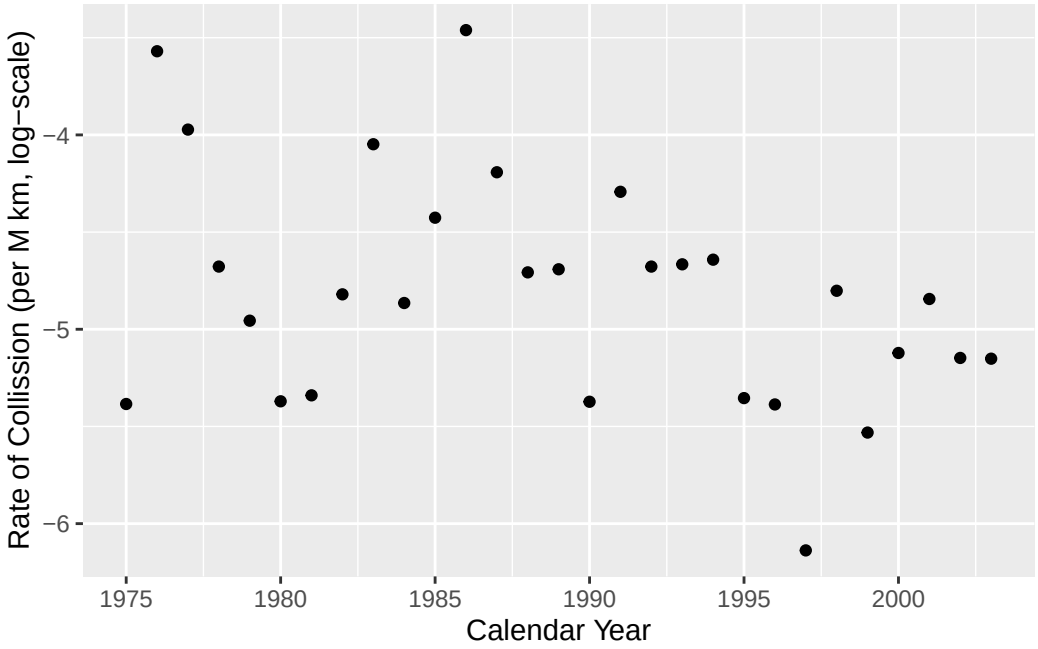


Figure 5.4: The rate of collisions, per million kilometers traveled (log-scale), between British passenger trains and road vehicles from 1975 to 2003.

A Poisson GLM might be our first choice for modeling the rate, but again such as model does not fit the data adequately. Overdispersion is clearly present.

Exercise

Fit a Poisson model to the rate of collisions and show that the fit is not adequate by plotting the absolute value of the quantile residuals against fitted values.

Solution

To fit a Poisson model with a logarithmic link function to the rate of collisions we would specify the following model

$$\log\left(\mathbb{E}\left[\frac{y}{t}\right]\right) = \beta_0 + \beta_1 x.$$

This model can be re-written as

$$\log(\mathbb{E}[y]) = \beta_0 + \beta_1 x + \log(t),$$

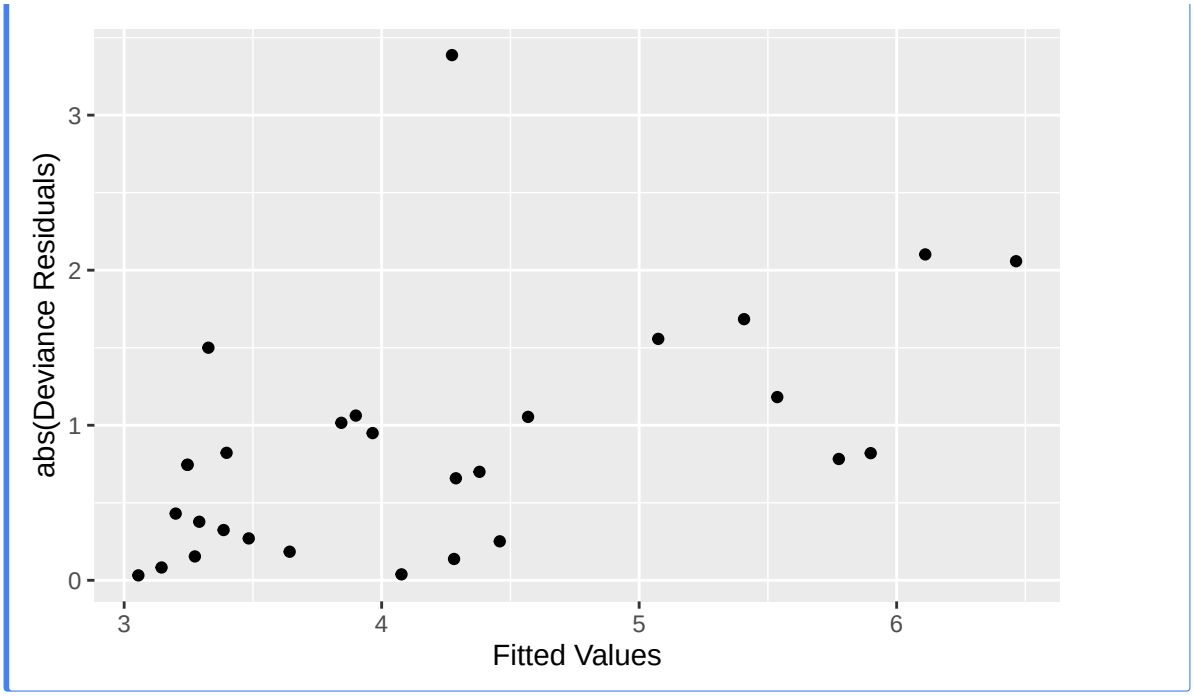
where the last term, $\log(t)$, is an offset term for the amount of exposure. The code to fit the above model is

```
train.poi <- glm(y ~ x + offset(log(t)),
                 data = train,
                 family = poisson(link = "log"))
summary(train.poi)
```

```
Call:
glm(formula = y ~ x + offset(log(t)), family = poisson(link = "log"),
    data = train)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.21142    0.15892  -26.50  < 2e-16 ***
x           -0.03292    0.01076   -3.06  0.00222 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 47.376  on 28  degrees of freedom
Residual deviance: 37.853  on 27  degrees of freedom
AIC: 133.52

Number of Fisher Scoring iterations: 5
```

Overdispersion is likely since the residual deviance is larger than the degrees of freedom. The plot of absolute value quantile residuals show a clear increasing trend as fitted values increase. This tells us that the variance function we have selected (in this case it is linear because we are using the Poisson distribution) is not increasing fast enough and so our assumption that the number of collisions is Poisson distributed is not correct.

```
ggplot(data = tibble(mu = predict(train.poi, type = "response"),
                     rD = resid(train.poi, type = "deviance")),
       mapping = aes(x = (mu),
                     y = abs(rD))) +
  geom_point() +
  labs(x = "Fitted Values",
       y = "abs(Deviance Residuals)")
```



Fitting a Poisson–gamma HGLM should be our next choice. We will model the dispersion parameter as a constant. The mean will include a random effect with a gamma distribution and since our response variable is a rate and we are using a log-link function, we will include an offset in our model.

```
model_mu <- DHGLMMODELING(Model = "mean",
                          Link = "log",
                          LinPred = y ~ x + (1 | id),
                          Offset = log(train$t),
                          RandDist = "gamma")
model_phi <- DHGLMMODELING(Model = "dispersion")
train.hglm <- dhglmfit(RespDist = "poisson",
                           DataMain = train,
                           MeanModel = model_mu,
                           DispersionModel = model_phi)
```

```
Distribution of Main Response :
                           "poisson"
[1] "Estimates from the model(mu)"
y ~ x + (1 | id)
[1] "log"
             Estimate Std. Error t-value
(Intercept) -4.13359    0.22304 -18.533
```

```
x           -0.03633    0.01452  -2.502
[1] "Estimates for logarithm of lambda=var(u_mu)"
[1] "gamma"
   Estimate Std. Error t-value
id   -1.752     0.4235  -4.137
[1] "========== Likelihood Function Values and Condition AIC =========="
                                        [,1]
-2ML (-2 p_v(mu) (h))          :  127.79514
-2RL (-2 p_beta(mu),v(mu) (h)) :  136.82822
cAIC                           :  129.86560
Scaled Deviance                :   11.31993
df                             :   15.56043
```

The fixed effects are both significant at the 5% level. The coefficient for year is negative and shows that as each year goes by we can expect the number of collisions to decrease by about 3.5%.

The variance of the random effect (on a log-scale) is -1.752 with a standard error equal to 0.423 and so the variance is statistically different from zero. The density for our random effect is shown in Figure 5.5.

```
theta <- exp(train.hglm[["lambda_coeff"]][1])
df <- tibble(x = train.hglm[["v_h"]][,1],
             y = 0)
```

```
ggplot(data = tibble(x = seq(0.01, 2.5, length = 200),
                     y = dgamma(x, scale = theta, shape = 1/theta)),
       mapping = aes(x = x,
                     y = y)) +
  geom_line() +
  geom_point(data = df,
             mapping = aes(x = exp(x),
                           y = y)) +
  labs(x = "Unobservable Random Effect",
       y = "Density")
```

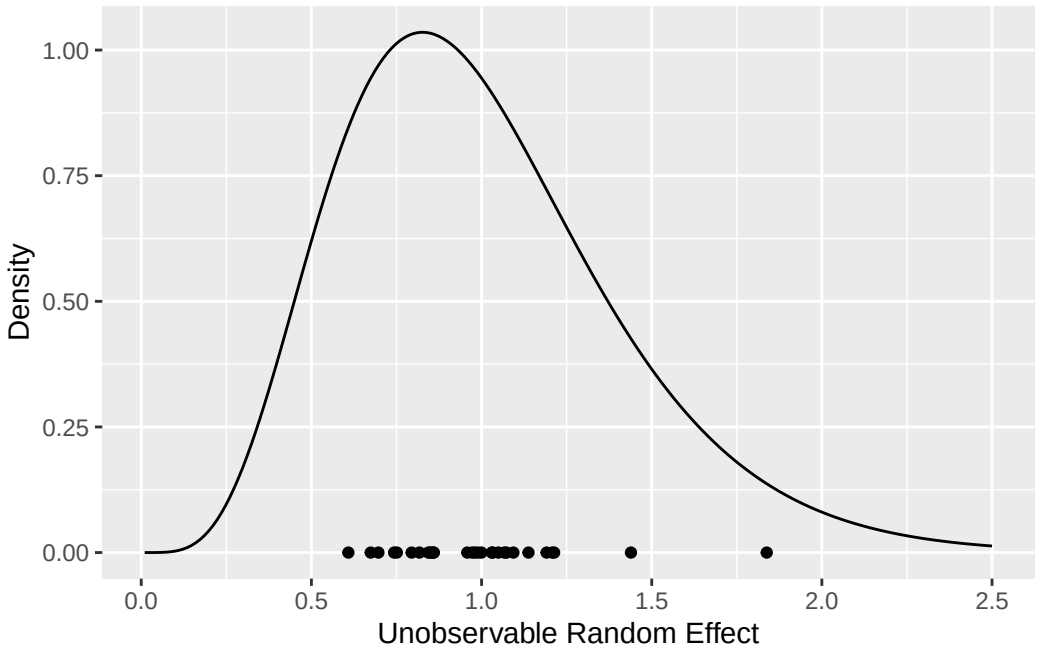


Figure 5.5: Estimated density function for train dataset along with the estimated random effects (points on the x-axis).

Figure 5.6 displays QQ-plots for the Poisson and the Poisson–gamma HGLM models. Note that the Poisson model shows that the distribution of the studentized deviance residuals have a fatter tail than the normal distribution. The two points in the upper-right hand corner are too large. Whereas for the Poisson–gamma HGLM model those two points are much closer to the theoretical line.

```
df <- tibble(model = factor(rep(c("Poisson Model", "HGLM Model"), each = 29),
                            levels = c("Poisson Model", "HGLM Model")),
             value = NA)
df$value[1:29] <- rstudent(train.poi)
df$value[30:58] <- train.hglm[["mean_residual"]][,1]
```

```
ggplot(data = df,
       mapping = aes(sample = value)) +
  facet_wrap(vars(model)) +
  geom_qq(aes()) + geom_qq_line() +
  labs(x = "Theoretical Quantiles",
       y = "Sample Quantiles")
```

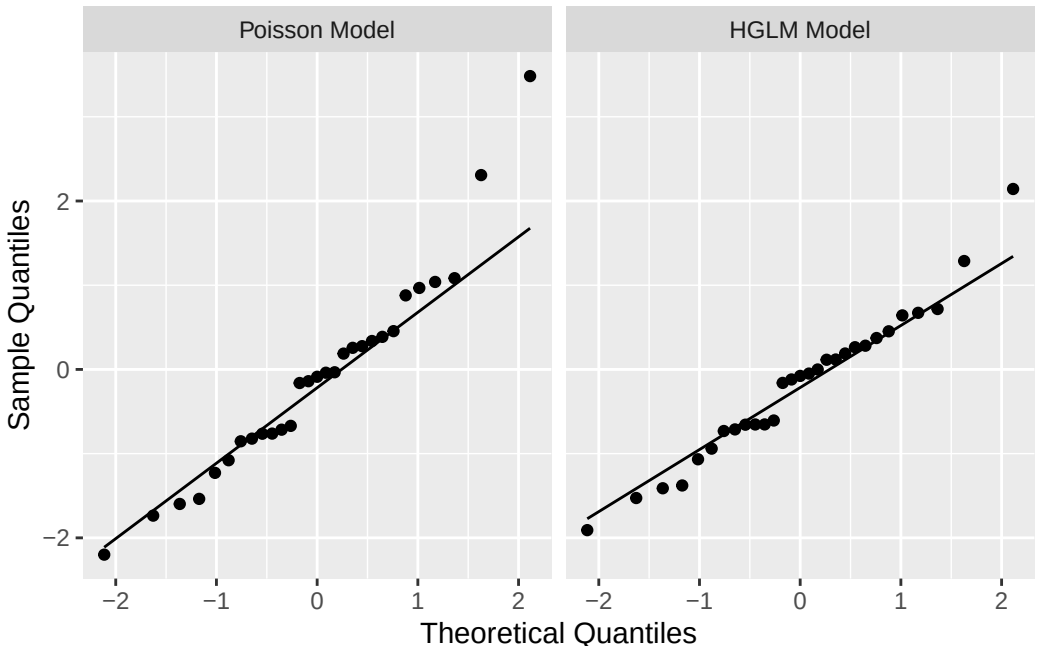


Figure 5.6: Quantile-quantile plot for the studentized deviance residuals from the Poisson and the Poisson-gamma (HGLM) models.

Diabetes Progression

For this example we use a data set of diabetes patients to illustrate the fitting of a joint GLM. The data was analyzed in (Efron et al. 2004; Antoniadis et al. 2016) and it involves 442 diabetic patients. Ten baseline variables for these patients were recorded and a year later a measure of disease progression was also collected. A model was sought to predict disease progression based on the baseline variables: age, sex, body mass index, average blood pressure, and six blood serum measurements. Table 5.1 displays the first six rows of the data as shown in Table 1 from (Efron et al. 2004, 408). The data is available in the R package `lars` under the name `diabetes`. Note that the explanatory variables in the dataset `diabetes` have been scaled to have mean zero and unit variance, but Table 5.1 shows the unscaled values for the first six rows.

```
tb |>
  kbl(booktabs = TRUE,
      align = "crrrrrrrrrrr",
      col.names = c("Patient", "age", "sex", "bmi", "abp", "tc",
                    "ldl", "hdl", "tch", "ltg", "glu", "y"),
      linesep = c("", "", "\\addlinespace")) |>
  add_header_above(c(" " = 1, " " = 1, " " = 1, " " = 1, " " = 1,
```

Table 5.1: First six rows of diabetes data.

					Serum measurements						Response
Patient	age	sex	bmi	abp	tc	ldl	hdl	tch	ltg	glu	y
1	59	2	32.1	101	157	93.2	38	4	4.9	87	151
2	48	1	21.6	87	183	103.2	70	3	3.9	69	75
3	72	2	30.5	93	156	93.6	41	4	4.7	85	141
4	24	1	25.3	84	198	131.4	40	5	4.9	89	206
5	50	1	23.0	101	192	125.4	52	4	4.3	80	135
6	23	1	22.6	89	139	64.8	61	2	4.2	68	97

Table 1 in (Efron et al. 2004).

```
                    "Serum measurements" =  6, "Response" = 1),
                  align = c("c", "r", "r", "r", "r", "c", "r")) |>
  kable_styling(latex_options = "scale_down") |>
  add_footnote(label = "Table 1 in (Efron et al. 2004).",
               notation = "none") |>
  kable_classic()
rm(tb)
```

```
p1 <- ggplot(data = diab,
             mapping = aes(x = age,
                           y = y)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Scaled Values of Age",
       y = "Disease Progression")

p2 <- ggplot(data = diab,
             mapping = aes(x = bmi,
                           y = y)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Scaled Values of BMI",
       y = "Disease Progression")

p3 <- ggplot(data = diab,
             mapping = aes(x = hdl,
                           y = y)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Scaled Values of HDL",
       y = "Disease Progression")
```

```
p4 <- ggplot(data = diab,
             mapping = aes(x = glu,
                           y = y)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Scaled Values of Glucose",
       y = "Disease Progression")
```

```
(p1 + p2) / (p3 + p4)
rm(p1, p2, p3, p4)
```

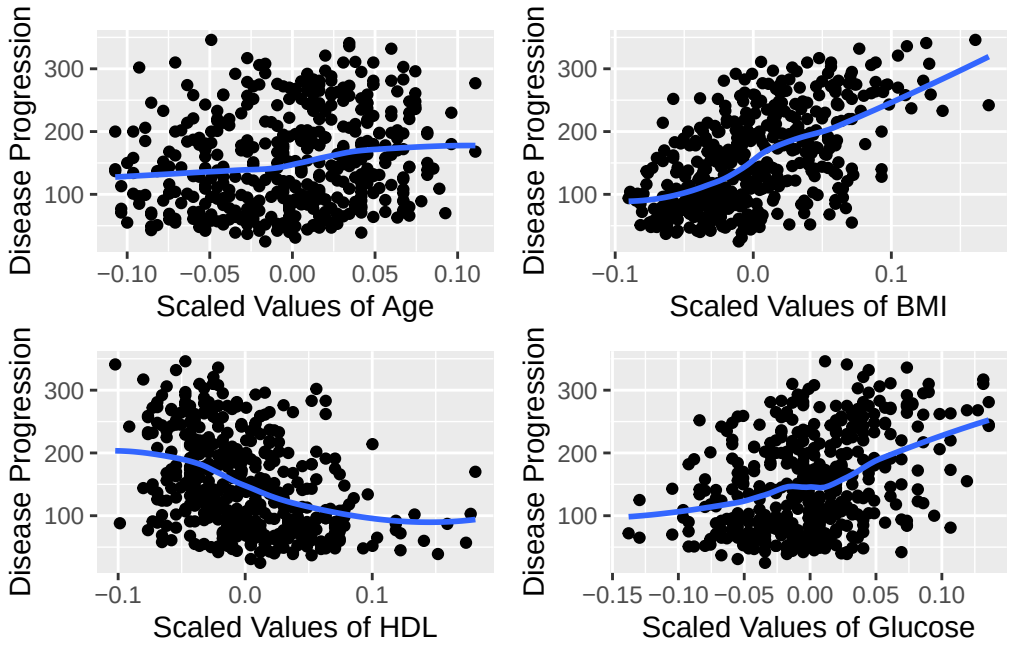


Figure 5.7: Exploratory graphs of disease progression versus explanatory variables age, body mass index (BMI), high density lipoprotein (HDL), and glucose. A scatterplot smooth has been added to aid in detecting the overall pattern. Note that in several of the panels, the variance in disease progression is not constant across the values of the explanatory variables.

Figure 5.7 displays four exploratory graphs of the response variable, disease progression, versus some explanatory variables. Body mass index (BMI) and glucose level show a strong relationship to the response whereas age and high density lipoprotein (HDL) show a weaker relationship. We can also see that the variability in the response is not constant across the range of values in the explanatory variables.

Some of the blood serum measurements (six variables) may be correlated to each other. Figure 5.8 displays a scatterplot matrix of these measurements where we can see that `tc` is highly positively linearly correlated with `ldl` (positions $(2,1)$ and $(1,2)$ in the plot matrix). And variable `tch` is highly negatively linearly correlated with `hdl` (positions $(4,3)$ and $(3,4)$ in the plot matrix). Variable `ldl` is also linearly correlated with `tch`, as is, `ltg` with `tch`.

```
ggpairs(diab[,5:10],
        axisLabels = "none",
        progress = FALSE)
```

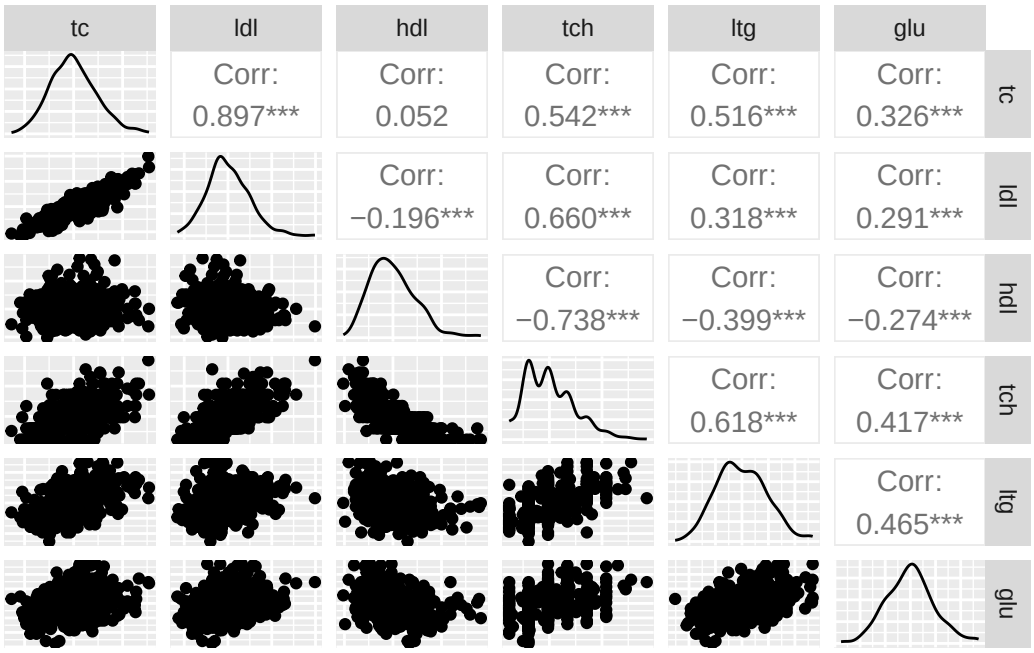


Figure 5.8: Scatterplot matrix for the blood serum variables. The diagonal entries show non-parametric estimates of the density function for each variable. The upper triangular entries are the pairwise linear correlation coefficents and the bottom triangular entries are the pairwise scatterplot for the variables.

Based on the observations from Figure 5.7 and Figure 5.8, we suspect that some of the variables will not be significant in predicting the mean response and some will help us model the variance of the response. Hence, we would like to fit a joint GLM model; that is, we want to have a generalized linear model for the response and also a generalized linear model for the dispersion parameter. We specify such a structure as follows:

```
model.mu <-  DHGLMMODELING(Model = "mean",
                           Link = "identity",
                           LinPred = y ~ age + sex + bmi + abp + tc +
                             ldl + hdl + tch + ltg + glu)

model.phi <- DHGLMMODELING(Model = "dispersion",
                           Link = "log",
                           LinPred = y ~ age + sex + bmi + abp + tc +
                             ldl + hdl + tch + ltg + glu)
```

Assuming that the response variable, disease progression, is adequately represented as a normal distribution we can fit the joint model via:

```
diab.model <- dhglmfit(RespDist = "gaussian",
                       DataMain = diab,
                       MeanModel = model.mu,
                       DispersionModel = model.phi)
```

```
Distribution of Main Response :
                      "gaussian"
[1] "Estimates from the model(mu)"
y ~ age + sex + bmi + abp + tc + ldl + hdl + tch + ltg + glu
[1] "identity"
            Estimate Std. Error   t-value     p_val       LL     UL
(Intercept)  151.721      2.583 58.732782 0.000e+00   146.66  156.8
age           12.249     54.655  0.224114 8.227e-01   -94.87  119.4
sex         -241.175     56.167 -4.293894 1.756e-05  -351.26 -131.1
bmi          475.325     67.734  7.017552 2.258e-12   342.57  608.1
abp          344.574     62.717  5.494131 3.926e-08   221.65  467.5
tc          -555.372    341.346 -1.627006 1.037e-01 -1224.41  113.7
ldl          277.509    276.958  1.001990 3.163e-01  -265.33  820.3
hdl            1.285    167.371  0.007678 9.939e-01  -326.76  329.3
tch          150.522    142.311  1.057694 2.902e-01  -128.41  429.5
ltg          651.564    135.923  4.793637 1.638e-06   385.16  918.0
glu           60.692     60.032  1.010989 3.120e-01   -56.97  178.4
[1] "Estimates from the model(phi)"
y ~ age + sex + bmi + abp + tc + ldl + hdl + tch + ltg + glu
[1] "log"
            Estimate Std. Error  t-value
(Intercept)    7.905    0.06813 116.0213
age           -2.710    1.58187  -1.7130
```

```
sex           -4.613    1.62120  -2.8454
bmi            1.861    1.76060   1.0571
abp            4.357    1.73308   2.5142
tc           -15.799   11.28435  -1.4001
ldl           16.515    9.20588   1.7939
hdl           -3.058    5.73781  -0.5329
tch           -6.654    4.31590  -1.5417
ltg            8.042    4.62341   1.7393
glu            2.310    1.74794   1.3214
[1] "========== Likelihood Function Values and Condition AIC =========="
                             [,1]
-2ML (-2 h)           :  4737.276
-2RL (-2 p_beta (h)) :  4629.960
cAIC                  :  4759.276
Scaled Deviance     :    431.000
df                    :   431.000
```

The top section of the output gives the estimated coefficients for the model of the response variable and we can see that variables `sex`, `bmi`, `abp`, and `ltg` are all significant at the 5% level. The bottom section shows the estimated coefficients for the dispersion model. Here the coefficients for `sex` and average blood pressure `abp` are significant.

Re-estimating the model with only the significant variables gives us the following estimated coefficients.

```
model.mu <- DHGLMMODELING(Model = "mean",
                          Link = "identity",
                          LinPred = y ~ sex + bmi + abp + ltg)
model.phi <- DHGLMMODELING(Model = "dispersion",
                           Link = "log",
                           LinPred = y ~ sex + abp)
diab.final <- dhglmfit(RespDist = "gaussian",
                       DataMain = diab,
                       MeanModel = model.mu,
                       DispersionModel = model.phi)
```

```
Distribution of Main Response :
                      "gaussian"
[1] "Estimates from the model(mu)"
y ~ sex + bmi + abp + ltg
[1] "identity"
            Estimate Std. Error t-value     p_val     LL     UL
```

```
(Intercept)    152.2      2.641  57.638 0.000e+00  147.0 157.39
sex           -157.0     56.783  -2.765 5.699e-03 -268.3 -45.69
bmi            585.8     64.444   9.090 9.929e-20  459.5 712.09
abp            312.6     64.841   4.821 1.426e-06  185.5 439.71
ltg            551.0     63.952   8.616 6.930e-18  425.7 676.36
[1] "Estimates from the model(phi)"
y ~ sex + abp
[1] "log"
            Estimate Std. Error t-value
(Intercept)    8.020    0.06765 118.543
sex           -3.031    1.46589  -2.068
abp            2.880    1.46782   1.962
[1] "========== Likelihood Function Values and Condition AIC =========="
                             [,1]
-2ML (-2 h)          :  4793.995
-2RL (-2 p_beta (h)) :  4750.245
cAIC                 :  4803.995
Scaled Deviance      :   437.000
df                   :   437.000
```

Figure 5.9 shows the studentized deviance residuals against the fitted values for both the mean and dispersion models. For the mean model, the overall shape of the points looks random with a slight increase on the lower end of the fitted values. For the dispersion model, we have slight curvature of the residuals but it is minimal.

```
p1 <- ggplot(data = diab.sm,
       mapping = aes(x = mean.mu,
                     y = mean.sr)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Scaled Fitted Values",
       y = "Studentized Residuals",
       title = "Mean Model")
p2 <- ggplot(data = diab.sm,
       mapping = aes(x = phi.mu,
                     y = phi.sr)) +
  geom_point() + geom_smooth(se = FALSE) +
  labs(x = "Scaled Fitted Values",
       y = "Studentized Residuals",
       title = "Dispersion Model")
p1 + p2
```

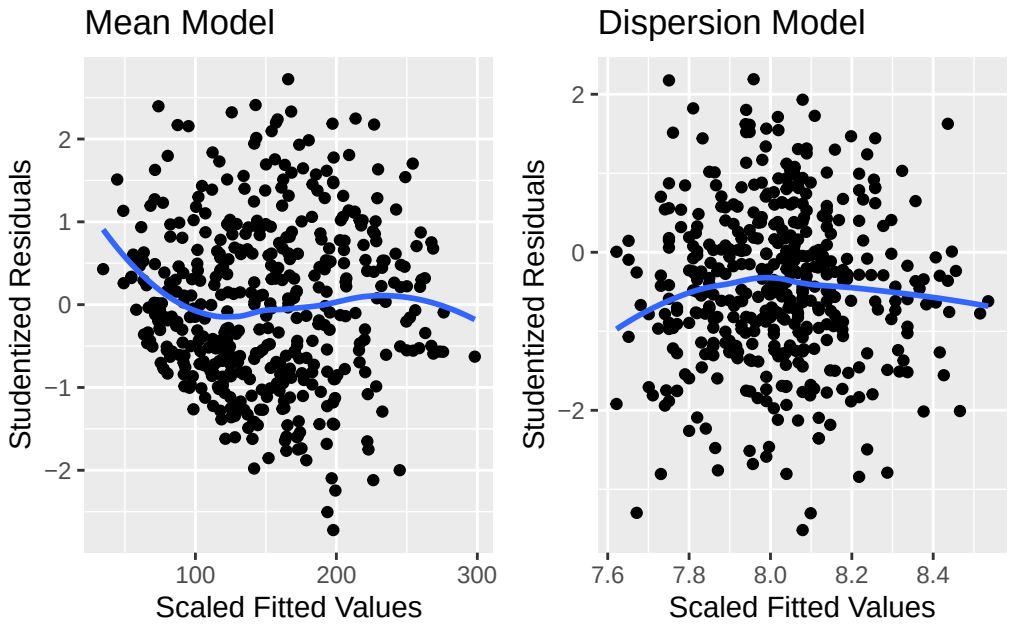


Figure 5.9

5.3 Summary

In this chapter we introduced a class of hierarchical generalized linear mixed models (Youngjo Lee, Nelder, and Pawitan 2021) that extend the generalized linear model by allowing random effects with normal and non-normal distributions and modeling the dispersion parameter via explanatory variables with both fixed and random effects.

We can think of these models as a pair of {mean, dispersion}-models. The standard generalized linear model would be specified as $\{\text{GLM}(\mu), \text{constant}\}$; meaning that we have a GLM for the mean of the response variable and a constant dispersion model.

Other models in the following crude diagram

```
LM ---> LMM ---> GLMM ---------> HGLM ---> DHGLM
   \          /               /
    \        /               /
     \> GLM ---> Joint GLM /
```

can be specified as

1. **Linear Mixed Model** (LMM): the mean model would be a hierarchical GLM with normally distributed random effects, a Gaussian distribution for the response together with the identity link function.

2. **Joint GLM** (JGLM): Both the mean model and the dispersion model are generalized linear models.
3. **Generalized Linear Mixed Model** (GLMM): the dispersion model is constant. The model for the mean response is a generalized linear model with normally distributed random effects.
4. **Hierarchical Generalized Linear Model** (HGLM): Both the mean and dispersion are modeled. The mean model is a generalized linear model with random effects that are not restricted to being normally distributed. The dispersion parameter is modeled via a GLM with fixed effects only.
5. **Double Hierarchical Generalized Linear Model** (DHGLM): extends the HGLM model by allowing random effects in the model for the dispersion parameter and allowing the modeling of the variance of the random effects via explanatory variables.

We revisited the Hachemeister dataset to show how the same model (essentially) can be fitted to the data based on the new class of hierarchical generalized linear models. We also presented two new examples with gamma random effects: fabric fault data and train collisions with road vehicles. For both of these examples, the response variable was a count for which we used a Poisson distribution. But the Poisson model was not adequate for these data because of overdispersion. Hence we introduced a gamma random effect yielding the negative binomial distribution.

In the final example we analyzed a dataset quantifying the disease progression of diabetic patients. Here we noticed that the variance of the response variable was not constant and so we wanted to introduce explanatory variables to model it. Therefore, we fitted a joint GLM to these data where we specified a linear predictor for the mean disease progression and also introduced another linear predictor for the dispersion model.

In the next chapter, we will present several examples that make use of random effects for categorical variables that have a large number of levels. This will bring us back to incorporating credibility into our modeling.

6 Applications

```
library(patchwork)
library(statmod)
library(gamlss)
library(MASS)
library(dhglm)
library(kableExtra)
library(tidyverse)
```

6.1 Massachusetts Auto Bodily Injury Claims

For this example, we use a data set of automobile bodily injury claims from the Commonwealth of Massachusetts. This data has been used to illustrate several different types of analyses, such as,

1. modeling hidden exposures (Rempala and Derrig 2005),
2. credibility using copulas (Edward W. Frees and Wang 2005), and
3. multivariate credibility (Edward W. Frees 2003).

The data is available in the R package `CASdatasets` (Dutang and Charpentier 2020) under the name `usmassBI2`. The data is longitudinal and describes the claims experience for 29 randomly chosen towns (out of more than 300) in Massachusetts for the years 1993 to 1998. The variables available are `TOWNCODE`, `YEAR`, `AC` (average claims per unit of exposure), `PCI` (per-capita income of the town), and `PPSM` (population per square mile).

As described in (Edward W. Frees 2003; and Edward W. Frees and Wang 2005) the average claim amounts have already been restated in 1991 dollars using the consumer price index (CPI) in order to mitigate any time trends due to inflation. This data has also been analyzed in (Charpentier 2015, chap. 15) and we follow their discussion.

Data Exploration

Table 6.1: Descriptive statistics for average claims per unit of exposure for a random sample of 29 towns in Massachusetts. Dollar amounts have been restated to 1991 using the consumer price index.

	Average Claim Amount					
	1993	1994	1995	1996	1997	1998
Mean	133.00	129.03	143.38	141.17	142.94	134.37
Median	131.57	131.45	138.76	149.00	144.73	131.96
Std. Deviation	31.59	32.63	38.28	39.28	36.22	32.85
Minimum	80.03	42.74	61.04	66.20	61.68	74.89
Maximum	212.46	209.52	238.22	201.99	248.75	191.05

```
data("usmassBI2", package = "CASdatasets")
```

```
tb <- usmassBI2 |>
  group_by(YEAR) |>
  summarize("Mean" = mean(AC),
            "Median" = median(AC),
            "Std. Deviation" = sd(AC),
            "Minimum" = min(AC),
            "Maximum" = max(AC)) |>
  pivot_longer(cols = 2:6,
               names_to = "measure",
               values_to = "value") |>
  pivot_wider(names_from = YEAR,
              values_from = value)
```

```
tb |>
  kbl(booktabs = TRUE,
      col.names = c("", 1993:1998),
      digits = c(0, rep(2, 6))) |>
  add_header_above(c(" ", "Average Claim Amount" = 6)) |>
  kable_classic()
rm(tb)
```

Table 6.1 displays the descriptive statistics for average claim size by calendar year. Note that the means and medians look reasonably stable across calendar years. Also the standard deviation seems to hover around 35% and the maximums and minimums do not seem to fluctuate heavily; therefore, it seems like the distributions are stable across years.

In Figure 6.1 we have a multiple time series plot (aka spaghetti plot) where each line represents the observations, across time, for one town. Two towns have been highlighted: 35 and 53. Town code 35 has large average claims in the first couple of years and town code 53 has some of the lowest claims across all years. Figure 6.2 displays average claim cost against per capita income and population per square mile. Towns 35 and 53 have also been highlighted and note that town 53 is sparsely populated and has a high per capita income whereas town 35 has the highest population density and fairly large average claim costs. Also note that for each town, the per capita income and population per square mile do not fluctuate much by year, but the average claims do. Based on this observation, when modeling the average claim cost, an intercept for each town would make sense.

```
ggplot(data = filter(usmassBI2,
                     TOWNCODE != 35 & TOWNCODE != 53),
       mapping = aes(x = YEAR,
                     y = AC,
                     group = TOWNCODE)) +
  geom_line(color = "gray") +
  geom_point(color = "darkgray") +
  geom_line(data = filter(usmassBI2,
                          TOWNCODE == 35),
            mapping = aes(x = YEAR,
                          y = AC),
            color = "pink") +
  geom_point(data = filter(usmassBI2,
                           TOWNCODE == 35),
             mapping = aes(x = YEAR,
                           y = AC),
             color = "red") +
  geom_line(data = filter(usmassBI2,
                          TOWNCODE == 53),
            mapping = aes(x = YEAR,
                          y = AC),
            color = "lightblue") +
  geom_point(data = filter(usmassBI2,
                           TOWNCODE == 53),
             mapping = aes(x = YEAR,
                           y = AC),
             color = "blue") +
  labs(x = "Calendar Year",
       y = "Average Claim")
```

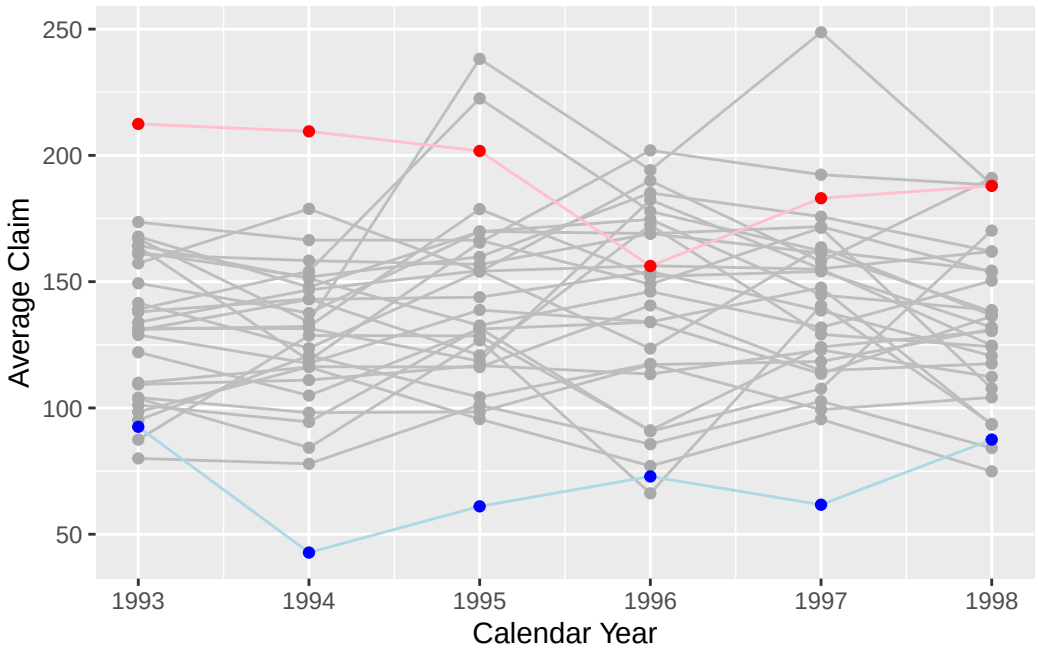


Figure 6.1: Multiple time series plot of aveargе claim per unit of exposure. Each time series corresponds to one of the 29 towns in the data. The red dots joined by pink lines correspond to town code 35, which has some of the highest average claims during the first couple of years. The blue points joined by light blue lines correspond to town code 53, which has some of the lowest average claims.

```
p <- ggplot(data = filter(usmassBI2,
                          TOWNCODE != 35 & TOWNCODE != 53),
            mapping = aes(x = PCI / 1000,
                          y = AC)) +
  geom_point(color = "darkgray") +
  geom_point(data = filter(usmassBI2,
                           TOWNCODE == 35),
             mapping = aes(x = PCI / 1000,
                           y = AC),
             color = "red") +
  geom_point(data = filter(usmassBI2,
                           TOWNCODE == 53),
             mapping = aes(x = PCI / 1000,
                           y = AC),
             color = "blue") +
  labs(x = "Per Capita Income ($000)",
       y = "Average Claim Cost") +
  theme(legend.position = "none")
```

```
q <- ggplot(data = filter(usmassBI2,
                          TOWNCODE != 35),
            mapping = aes(x = log(PPSM),
                          y = AC)) +
  geom_point(color = "darkgray") +
  geom_point(data = filter(usmassBI2,
                           TOWNCODE == 35),
             mapping = aes(x = log(PPSM),
                           y = AC),
             color = "red") +
  geom_point(data = filter(usmassBI2,
                           TOWNCODE == 53),
             mapping = aes(x = log(PPSM),
                           y = AC),
             color = "blue") +
  labs(x = "Population per Sq. Mile (log-scale)",
       y = "Average Claim Cost") +
  theme(legend.position = "none")
p + q
rm(p, q)
```

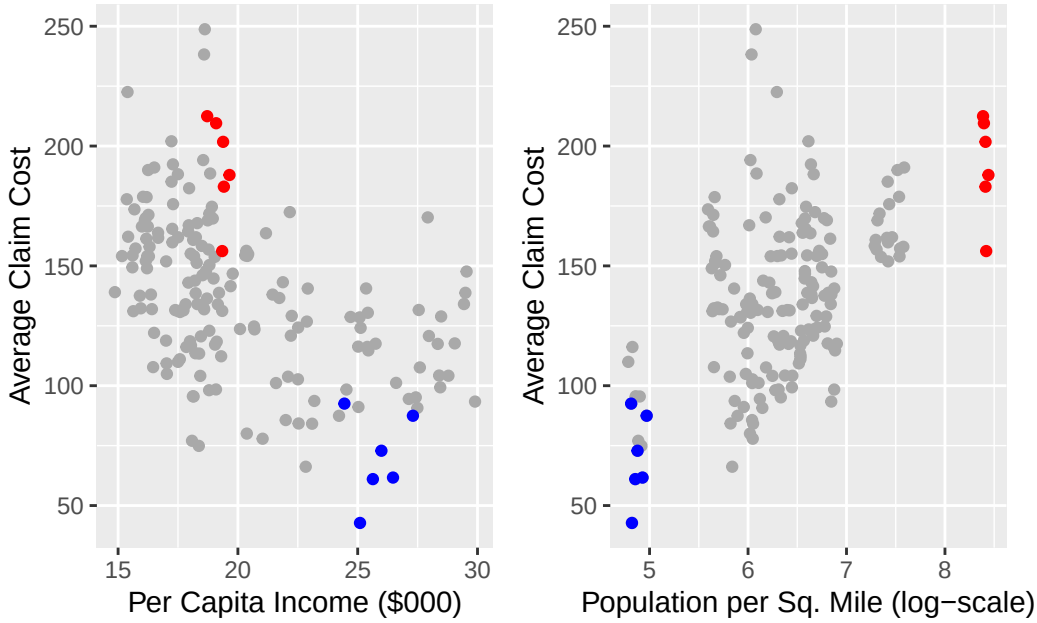


Figure 6.2: Average claim sizes by per capita income (in thousands of dollars) and population per square mile (in log base 10 scale). The red colored points correspond to town code 35 and the blue colored points correspond to town code 53.

Figure 6.3 displays the histogram of average claim size across all towns and years along with a non-parametric estimate of the density (solid line) and a normal distribution (dashed line). Overall, the normal distribution fits this data well.

```
ggplot(data = usmassBI2,
       mapping = aes(x = AC)) +
  geom_histogram(aes(y = after_stat(density)), bins = 30, fill = "grey") +
  geom_density(linewidth = 0.75) +
  geom_line(data = tibble(x = seq(25, 275, length = 100),
                          y = dnorm(x, mean = 137.32, sd = 35.18)),
            mapping = aes(x = x, y = y),
            color = "blue",
            linetype = "dashed",
            linewidth = 0.75) +
  labs(x = "Average Claim Cost",
       y = "Density")
```

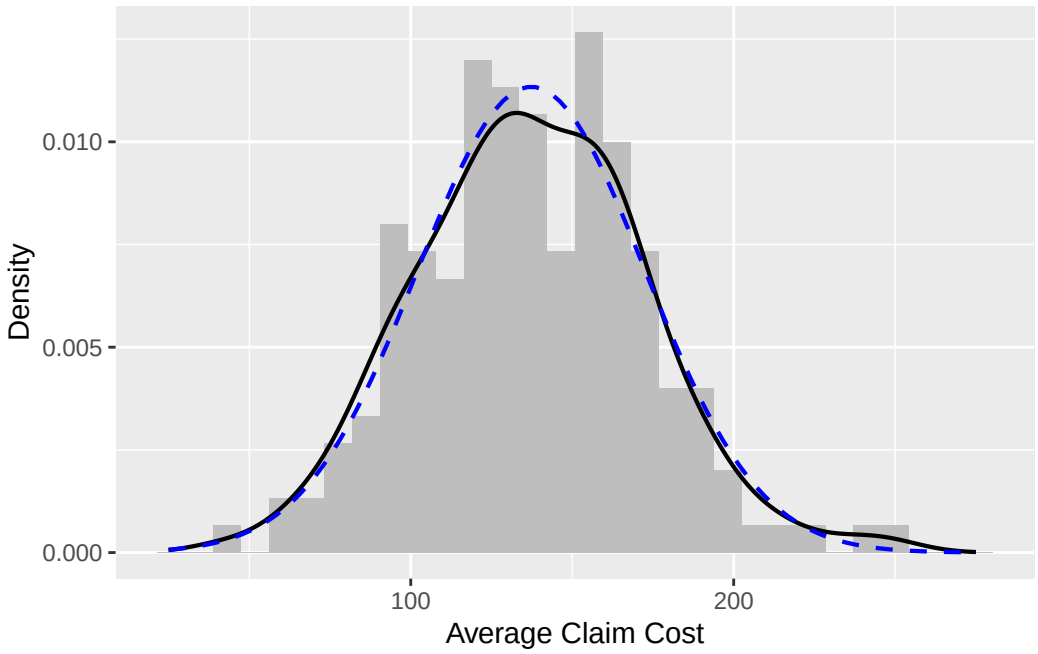


Figure 6.3: Histogram of average claim costs along with a non-parametric estimate of the density function (solid line) and a normal density function (dashed line) chosen to match the empirical mean and standard deviation across all towns and all years.

Modeling Average Claim Size

From our exploratory analysis of the data we can start our modeling of average claim cost by using a normal distribution for the response variable and applying the following transformations to the explanatory variables

1. shift the origin for `YEAR` to 1992,
2. scale the per capita income to measure it in thousands of dollars, and
3. compute the logarithm of population per square mile

Also we will take calendar years 1993 to 1997 to train our models and keep 1998 for validation purposes.

```
usmassBI2 <- usmassBI2 |>
  mutate(YR = YEAR - 1992,
         lnPPSM = log(PPSM),
         PCI.k = PCI / 1000)

db.train <- usmassBI2 |>
  filter(YEAR < 1998)
```

```r
db.test <- usmassBI2 |>
  filter(YEAR == 1998)
```

Complete Pooling

First we will fit a model ignoring the `TOWNCODE` variable and so we are pooling all of our data together, implicitly assuming that all the towns in Massachusetts form a single homogeneous group. This is clearly not a reasonable assumption, but it is a good starting point.

```r
bi.all <- lm(AC ~ PCI.k + lnPPSM + YR,
             data = db.train)
summary(bi.all)
```

```
Call:
lm(formula = AC ~ PCI.k + lnPPSM + YR, data = db.train)

Residuals:
    Min      1Q  Median      3Q     Max
-51.661 -16.846  -0.419  12.680 103.850

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  68.8695    23.1298   2.978  0.00342 **
PCI.k        -4.2410     0.5604  -7.568 4.47e-12 ***
lnPPSM       22.3442     2.9603   7.548 5.00e-12 ***
YR            3.8353     1.5324   2.503  0.01346 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 25.99 on 141 degrees of freedom
Multiple R-squared:  0.4812,    Adjusted R-squared:  0.4701
F-statistic: 43.59 on 3 and 141 DF,  p-value: < 2.2e-16
```

The estimated value of the intercept, 68.87, is the average claim cost in 1992 for a town that has a population density of one person per square mile and a per capita income of zero. Such a town in Massachusetts does not exist. The average per capita income (in thousands) and the logarithm of the population per square mile across all towns in our data is 20.06 and 6.38, respectively. Therefore, using our current model, we would estimate the expected claim costs in 1993 to be

$$68.87 - 4.24 \cdot 20.06 + 22.34 \cdot 6.38 + 3.84 \cdot 1 = 130.18,$$

close to the middle of the data for 1993 shown in Figure 6.1.

The coefficient for calendar year of 3.84 tells us that as we move from one year to the next, the average claim cost will increase by this dollar amount. Keeping in mind that the data had already been adjusted to account for inflation we must attribute this increase to other sources. As per capita income increases by $1,000, we see a decline in the average claim cost of 4.24. And if we had a 10% increase in population density, then the average claim costs would increase by about 2.13.

```
db.train <- db.train |>
  mutate(bi.all.mu = predict(bi.all),
         bi.all.rS = rstandard(bi.all))
```

```
p1 <- ggplot(data = db.train,
             mapping = aes(x = bi.all.mu,
                           y = bi.all.rS)) +
  geom_point() +
  geom_smooth(se = TRUE) +
  labs(x = "Fitted Values",
       y = "Standardized Residuals")
p2 <- ggplot(data = db.train,
             mapping = aes(x = bi.all.mu,
                           y = abs(bi.all.rS))) +
  geom_point() +
  geom_smooth(se = TRUE) +
  labs(x = "Fitted Values",
       y = "|Standardized Residuals|")
(p1 + p2)
```

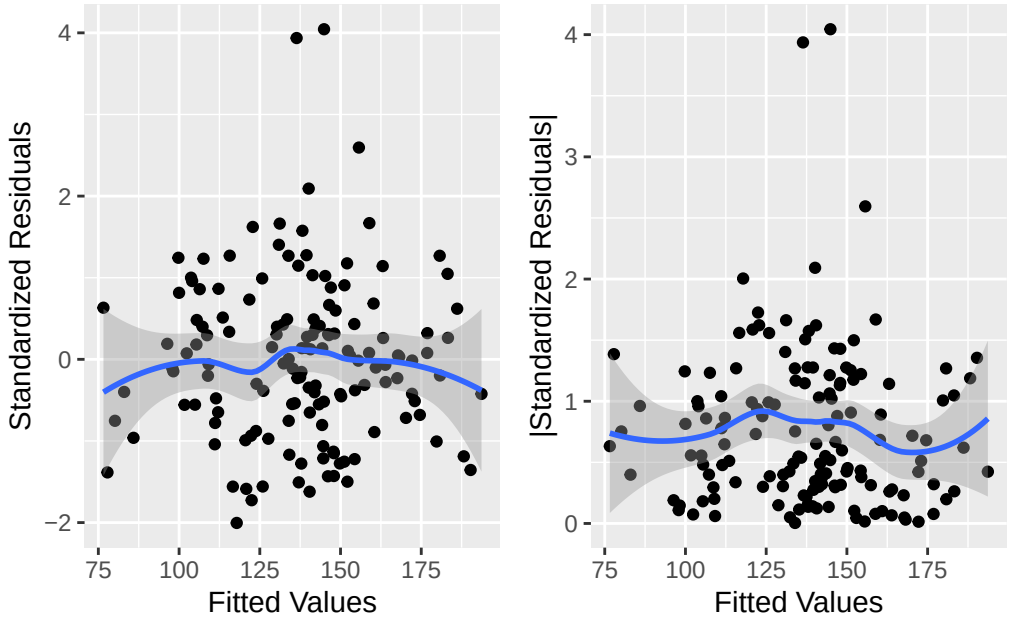


Figure 6.4: Diagnostic plots for model `bi.all`. Left-hand panel shows stadardized residuals versus fitted values and the right-hand panel is the absolute value of the standardized residuals against the fitted values. The blue line in each panel is a scatterplot smooth showing the overall trend of the points.

```
p1 <- ggplot(data = db.train,
             mapping = aes(x = YEAR,
                           y = bi.all.rS)) +
  geom_point() +
  geom_smooth(se = TRUE) +
  labs(x = "Calendar Year",
       y = "Standardized Residuals")
p2 <- ggplot(data = db.train,
             mapping = aes(x = PCI.k,
                           y = bi.all.rS)) +
  geom_point() +
  geom_smooth(se = TRUE) +
  labs(x = "Per Capita Income ($000)",
       y = "Standardized Residuals")

p3 <- ggplot(data = db.train,
             mapping = aes(x = lnPPSM,
                           y = bi.all.rS)) +
  geom_point() +
```

```
  geom_smooth(se = TRUE) +
  labs(x = "Log Population per Square Mile",
       y = "Standardized Residuals")
(p1 + p2) / (p3 + (plot_spacer() +
                     theme(plot.margin = unit(c(0,0,0,33), "pt"))))
rm(p1, p2, p3)
```

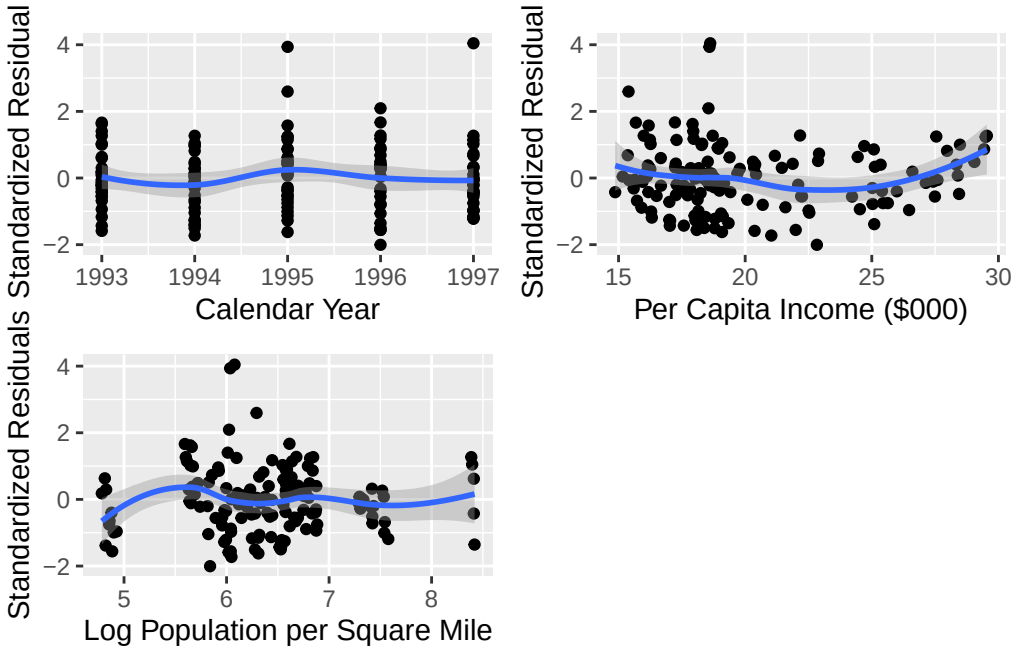


Figure 6.5: Diagnostic plots for model `bi.all`. Each panel shows the standardized residuals against a predictor variable. The blue line in each panel shows the overall trend of points.

Figure 6.4 and Figure 6.5 show diagnostic plots for the model `bi.all`. All five plots show that the model seems adequate. The left-hand panel of Figure 6.4 shows the expected pattern of a random cloud of points centered about $y = 0$. There are four observations with residuals greater than 2. Three of these observations come from `TOWNCODE` 45 and one from `TOWNCODE` 16. Looking at calendar year 1995 in Figure 6.1, town code 45 corresponds to the upper most point and town code 16 is the second highest point. Tracing the line for town code 16 we can see that in 1995 they had a large average claim cost but all other years it remained stable. In contrast, town code 45 had the highest average claim cost in 1995, then in 1996 it had the second highest, and in 1997 the highest claim cost again (well above the second highest).

The right-hand side panel of Figure 6.4 shows a fairly uniform spread of points as the fitted values increase; thus, our assumption that the distribution of the response variable, average

claim size, is normal seems appropriate. Note that for the largest fitted values we see an upward trend letting us know that for these values we have more variability in our data than our model provides.

In Figure 6.5 we have plotted the three explanatory variables against the standardized residuals. For calendar year we see no meaningful patterns. For the per capita income, we observe that in the range from 20,000 to 25,000, the model tends to overpredict, and above 25,000, the model systematically underpredicts. While the overall pattern is flat for the logarithm of population per square mile, there are a few isolated places where the model tends to overpredict (below 5) and underpredict (slightly above 5.5).

No Pooling

In the previous section, we pooled all of our data assuming that all 29 sampled towns in Massachusetts would create a homogeneous group and fitted the linear model

$$\mathbb{E}[\text{AC}] = \beta_0 + \beta_1 \cdot \text{PCI.k} + \beta_2 \cdot \log(\text{PPSM}) + \beta_3 \cdot \text{YR}.$$

We can take a diametrically opposite stance and assume that no two towns are similar in any way. With this view, we would fit the above linear model to each town separately by

1. creating a list where each item is the data frame for one town
2. fitting the linear model to each item in the list

as follows:

```
twns <- unique(as.character(db.train$TOWNCODE))
twn.db <- map(twns,
              \(x) filter(db.train, TOWNCODE == x) |>
                select(AC, PCI.k, lnPPSM, YR, YEAR, TOWNCODE))
names(twn.db) <- twns
```

```
twns.lm <- map(twn.db,
               \(d) lm(AC ~ PCI.k + lnPPSM + YR,
                       data = d))
```

We now have 29 regression models. Some of them fit well, while others do not. For example, we can collect for each model, the R^2 measure as an indicator of model fit via

```
map_dbl(twns.lm,
        \(x) summary(x)$r.squared) |>
  sort() |>
  round(3)
```

```
   50    39    42    40    31    41    45    33    44    14    53    30    43
0.028 0.073 0.089 0.424 0.451 0.534 0.552 0.580 0.600 0.645 0.645 0.720 0.737
   17    52    13    51    12    38    16    35    37    21    10    34    15
0.742 0.790 0.811 0.844 0.857 0.902 0.915 0.936 0.952 0.959 0.979 0.988 0.993
   32    11    36
0.995 0.998 0.999
```

For town codes 50, 39, and 42, the adjusted R^2 measure is extremely low (less than 9%), and for town codes 17, 52, 13, (middle of the list) the R^2 measure is 74.2%, 79.0%, and 81.1%, respectively. Whereas for town codes 32, 11, and 36, the measure is above 99.5%. Figure 6.6 shows an actual versus expected plot for these six towns arranged from low R^2 values to high. The gray line represents the line of perfect fit; that is, $y = x$ in each panel.

```
tb <- bind_rows(bind_cols(twn.db[["50"]], mu = predict(twns.lm[["50"]])),
                bind_cols(twn.db[["39"]], mu = predict(twns.lm[["39"]])),
                bind_cols(twn.db[["42"]], mu = predict(twns.lm[["42"]])),

                bind_cols(twn.db[["17"]], mu = predict(twns.lm[["17"]])),
                bind_cols(twn.db[["52"]], mu = predict(twns.lm[["52"]])),
                bind_cols(twn.db[["13"]], mu = predict(twns.lm[["13"]])),

                bind_cols(twn.db[["32"]], mu = predict(twns.lm[["32"]])),
                bind_cols(twn.db[["11"]], mu = predict(twns.lm[["11"]])),
                bind_cols(twn.db[["36"]], mu = predict(twns.lm[["36"]])))
tb$TOWNCODE <- fct_drop(tb$TOWNCODE)
tb$TOWNCODE <- fct_relevel(tb$TOWNCODE,
                           c("32","11","36","17","52","13","50","39","42"))
```

```
ggplot(data = tb,
       mapping = aes(x = AC,
                     y = mu)) +
  geom_point() +
  geom_abline(slope = 1,
              color = "gray") +
  labs(x = "Actual Average Claim Cost",
       y = "Expected Average Claim Cost") +
  facet_wrap(vars(TOWNCODE),
             scales = "free")
rm(tb)
```

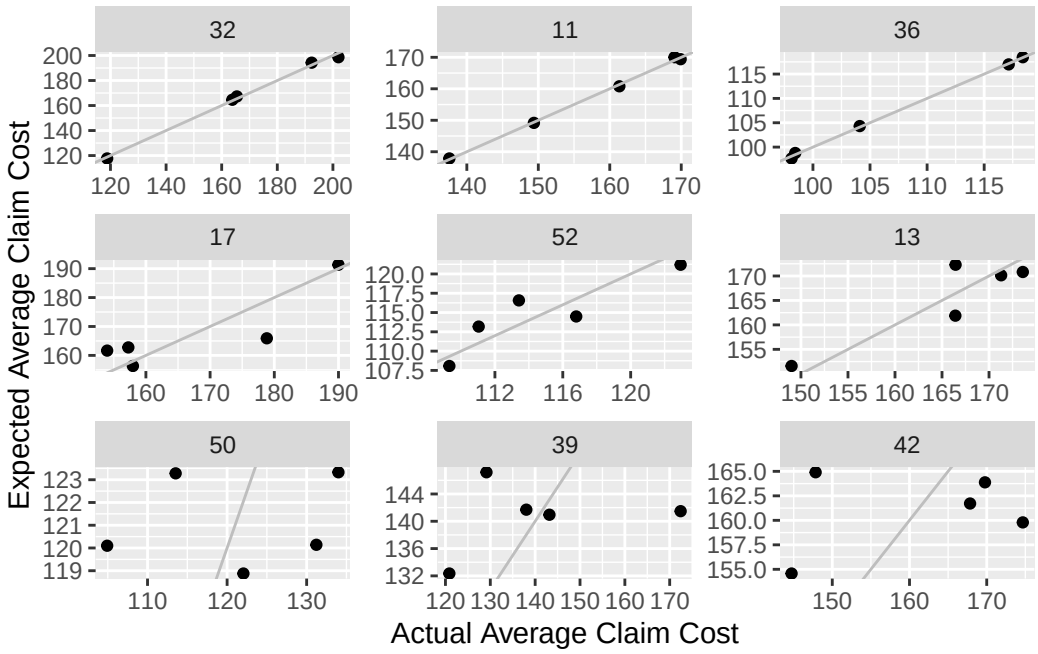


Figure 6.6: Actual versus predicted average claim costs from the regression lines fitted to each town individually. Towns 32, 11, and 36 have the highest R^2 values and towns 50, 39, and 42 have the lowest values. Towns 17, 52, and 13 are in the middle when R^2 measures are sorted. The panels are arranged from the lowest R^2 in the bottom left-hand corner to the highest R^2 value in the top right-hand corner. The gray line in each panel represents the line of perfect fit ($y = x$).

Clearly for the top three panels in Figure 6.6 the models accurately predict the actual average claims and we would feel confident in using it to predict what the claims experience in the next calendar year. But do we feel similarly about the bottom three models? For town code 50, actual experience in the past 5 years ranged from about 105 to 135; quite volatile. The model's range of values are from about 119 to 124; a very small range. The probability that our prediction (whatever it might be) reflects actual experience would be quite low.

Thus we have that some towns are highly credible in their experience while others are not. Based on Figure 6.1 we should include a town specific intercept in our model.

Fixed Effects Model

Consider incorporating a town specific intercept into the regression model. Thus we want to fit the following model

$$\mathbb{E}[\text{AC}] = \alpha_i + \beta_1 \cdot \text{PCI.k} + \beta_2 \cdot \log(\text{PPSM}) + \beta_3 \cdot \text{YR},$$

where α_i represents the intercept for town i. We can accomplish this by treating `TOWNCODE` as a categorical variable in our model.

```
bi.fixed <- lm(AC ~ TOWNCODE + PCI.k + lnPPSM + YR,
               data = db.train)
summary(bi.fixed)
```

```
Call:
lm(formula = AC ~ TOWNCODE + PCI.k + lnPPSM + YR, data = db.train)

Residuals:
    Min      1Q  Median      3Q     Max
-55.621  -8.911   0.276   9.058  50.129

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1457.753    795.425   1.833   0.0695 .
TOWNCODE11  -101.240     61.491  -1.646   0.1025
TOWNCODE12  -101.223     92.990  -1.089   0.2787
TOWNCODE13  -299.086    182.682  -1.637   0.1044
TOWNCODE14  -215.677    117.945  -1.829   0.0701 .
TOWNCODE15    21.421     19.879   1.078   0.2835
TOWNCODE16  -174.125    112.682  -1.545   0.1251
TOWNCODE17    42.113     32.229   1.307   0.1940
TOWNCODE21  -141.966     76.612  -1.853   0.0665 .
TOWNCODE30  -307.519    178.349  -1.724   0.0874 .
TOWNCODE31  -205.487    117.761  -1.745   0.0837 .
TOWNCODE32  -123.095     78.861  -1.561   0.1213
TOWNCODE33  -121.806     65.728  -1.853   0.0665 .
TOWNCODE34  -175.532     94.975  -1.848   0.0672 .
TOWNCODE35   223.190    117.709   1.896   0.0605 .
TOWNCODE36  -234.455    108.455  -2.162   0.0327 *
TOWNCODE37  -302.709    172.261  -1.757   0.0816 .
TOWNCODE38  -283.961    149.940  -1.894   0.0608 .
TOWNCODE39  -131.217     72.121  -1.819   0.0715 .
TOWNCODE40  -313.326    157.915  -1.984   0.0497 *
TOWNCODE41  -253.651    137.237  -1.848   0.0672 .
TOWNCODE42  -131.500     78.933  -1.666   0.0985 .
TOWNCODE43  -498.092    262.041  -1.901   0.0599 .
TOWNCODE44  -113.359     66.123  -1.714   0.0892 .
TOWNCODE45  -190.694    135.632  -1.406   0.1625
```

```
TOWNCODE50  -277.511    142.634  -1.946   0.0542 .
TOWNCODE51  -293.370    135.513  -2.165   0.0325 *
TOWNCODE52  -186.622     84.174  -2.217   0.0286 *
TOWNCODE53  -518.904    261.780  -1.982   0.0499 *
PCI.k         -1.374      7.595  -0.181   0.8568
lnPPSM      -176.078    106.343  -1.656   0.1005
YR             5.990      2.602   2.302   0.0231 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 18.88 on 113 degrees of freedom
Multiple R-squared:  0.7808,    Adjusted R-squared:  0.7206
F-statistic: 12.98 on 31 and 113 DF,  p-value: < 2.2e-16
```

In the above model, the intercept represents the average claims experience for `TOWNCODE` 10 with a zero per capita income, one person per square mile, and the variable `YR` set to zero (i.e. 1992). This number does not have practical significance because we know full well that town code 10 does not have a per capita income of zero or a population with only one person per square mile. In town code 10, the per capita income is about \$18,500 and the logarithm of the population per square mile is about 7.3. The other `TOWNCODE` coefficients measure deviations from `TOWNCODE` 10 with the other variables set as before.

Note that some of the town specific coefficients are not significant and many of them are significant at the 10% level but not at the 5% level. Per capita income and population per square mile are no longer significant and the effect of calendar year has increased to almost 6 and is significant at the 5% level. More importantly, an estimated yearly increase of \$6 is of practical significance since it represents about a 4.4% increase from the overall average claim cost (across all towns and years) of \$138.

With the above model we have estimated coefficients for these 29 specific towns. And while we can make inferences for them, they are just a sample of the more than 300 towns in the state and we could not easily justify using them to make inferences about other towns in the state. In the next section, we'll treat these towns as a random sample of the larger population of towns and fit a random effects model to this data.

Random Effects Model

Instead of treating the intercepts, α_i, for each of the 29 sampled towns in Massachusetts as fixed, we can treat them as random variables and fit the following linear mixed effects model

$$\mathbb{E}[\text{AC}] = \alpha_i + \beta_0 + \beta_1 \cdot \text{PCI} + \beta_2 \cdot \log(\text{PPSM}) + \beta_3 \cdot \text{YR},$$

where α_i is a normal random variable with mean zero and variance σ_α. The index i runs through all the towns and we fit this model as follows

```r
bi.rnd.int <- lme(AC ~ PCI.k + lnPPSM + YR,
                  data = db.train,
                  random = ~ 1 | TOWNCODE)
summary(bi.rnd.int)
```

```
Linear mixed-effects model fit by REML
  Data: db.train
       AIC      BIC   logLik
  1310.722 1328.415 -649.361

Random effects:
 Formula: ~1 | TOWNCODE
        (Intercept) Residual
StdDev:    18.44886 19.01929

Fixed effects:  AC ~ PCI.k + lnPPSM + YR
               Value Std.Error  DF   t-value p-value
(Intercept) 70.25708  39.65931 113  1.771516  0.0792
PCI.k       -4.19414   0.97273 113 -4.311736  0.0000
lnPPSM      21.98209   5.16832 113  4.253237  0.0000
YR           3.82988   1.14148 113  3.355195  0.0011
 Correlation:
       (Intr) PCI.k  lnPPSM
PCI.k  -0.555
lnPPSM -0.872  0.096
YR      0.079 -0.197 -0.082

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.56010534 -0.61512040  0.01209624  0.48774689  2.89511606

Number of Observations: 145
Number of Groups: 29
```

The above output tells us that the random intercepts, α_i, have a standard deviation equal to $\sigma_\alpha = 18.45$. This is the variability between the towns. Actuaries would call the square of this the *variance of the hypothetical means* and statisticians would call it the *between group variability*. The residual standard deviation is equal to $\sigma = 19.02$ and statisticians call this the *within group variability* and actuaries would say that its square is the *expected value of the process variance*. Note that the *between* town and the *within* town standard deviations are

quite similar. The ratio

$$\frac{\sigma_\alpha}{\sigma_\alpha + \sigma} = \frac{18.45}{18.45 + 19.02} = 49.2\%$$

is known as the intraclass correlation. For our data this ratio is close to 50% letting us know that the observations within a town are mildly correlated.

Note that the fixed effects for per capita income, population per square mile, and year are all statistically significant. The coefficient for calendar year is now estimated at \$3.83; still a sizable increase beyond the adjustment made to the data (prior to loading it) based on the consumer price index.

Figure 6.7 shows a diagnostic plot for the model where the y-axis has the standardized residuals and on the x-axis we have the fitted values. From this plot we can see that the assumption that our response variable has constant variance is reasonable. We do not see any fanning in or out of the residuals. There are no clear outliers in the plot and so our assumption that the data comes from a single data generating process (as defined by our model) also seems reasonable.

```
plot(bi.rnd.int)
```

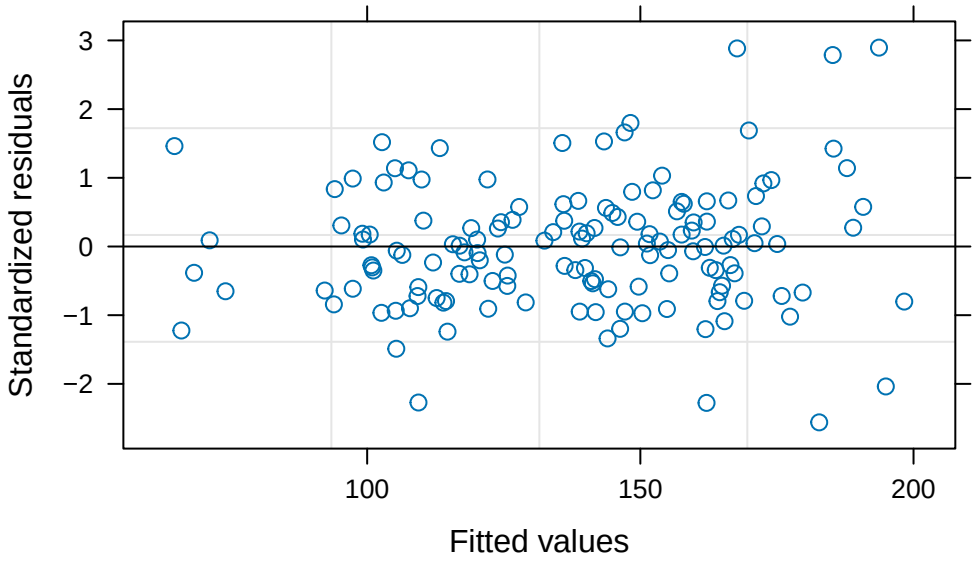


Figure 6.7: Standardized residuals versus fitted values for the random intercept model fitted to the sample of Massachusetts towns.

The between town variability, σ_α, certainly seems large enough to be significant, but we should check. The `intervals()` function will display approximate 95% confidence intervals for both fixed as well as random effects.

```
intervals(bi.rnd.int)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                lower      est.      upper
(Intercept) -8.315161 70.257082 148.829325
PCI.k       -6.121279 -4.194135  -2.266991
lnPPSM      11.742719 21.982095  32.221471
YR           1.568409  3.829885   6.091360

 Random Effects:
  Level: TOWNCODE
                    lower     est.    upper
sd((Intercept)) 13.25503 18.44886 25.67781

 Within-group standard error:
   lower     est.    upper
16.71330 19.01929 21.64345
```

Based on the above output the between town variability is clearly significant. Our current model has four fixed and one random parameters even though we have 29 different towns. If we expanded our data to include more towns, this model will still only have 5 parameters.

Exercise

From our multiple series plot, Figure 6.1, we might suspect that different towns should have different slope coefficient for the predictor variable `YR`. Should we add a random component for this variable?

Solution

If we want to add a random component to the slope of `YR` then we want to fit the following model

$$\mathbb{E}[\text{AC}] = \alpha_i + \beta_0 + \beta_1 \cdot \text{PCI} + \beta_2 \cdot \log(\text{PPSM}) + (\beta_3 + \gamma_i) \cdot \text{YR},$$

where both α_i and γ_i are random variables.
We can fit that model and display approximate 95% confidence intervals for the parameters via

```
bi.rnd.slope <- lme(AC ~ PCI.k + lnPPSM + YR,
                    data = db.train,
                    random = ~ 1 + YR | TOWNCODE)
intervals(bi.rnd.slope)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                lower      est.      upper
(Intercept) -8.483553 67.376635 143.236823
PCI.k       -5.920253 -4.037949  -2.155646
lnPPSM      12.083632 21.958795  31.833958
YR           1.408192  3.795298   6.182403

 Random Effects:
  Level: TOWNCODE
                          lower       est.      upper
sd((Intercept))       6.6223297 14.3342800 31.0270845
sd(YR)                0.3672731  2.4125204 15.8472126
cor((Intercept),YR) -0.9935728  0.4066217  0.9988532

 Within-group standard error:
   lower     est.    upper
16.11193 18.64116 21.56742
```

From the output, we can see that the standard deviation for the random slope, σ_γ, has been estimated at 2.41 and its confidence interval does not include zero; therefore, the model suggests that a random slope for each town is important.
A five point summary of the estimated random slopes yields

```
summary(ranef(bi.rnd.slope)[["YR"]])
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-2.35759 -0.82400 -0.07675  0.00000  0.32923  5.83271
```

Model Predictions

So far we have estimated four models

1. `bi.all`: complete pooling of all data,
2. `bi.fixed`: fixed effects for `TOWNCODE`,

3. `bi.rnd.int`: random effects for `TOWNCODE`, and
4. `bi.rnd.slope`: random effects for `TOWNCODE` and `YR`

```
bi.models <- list(bi.all,
                  bi.fixed,
                  bi.rnd.int,
                  bi.rnd.slope)
```

```
db.train <- db.train |>
  mutate(bi.all.mu = predict(bi.all),
         bi.fixed.mu = predict(bi.fixed),
         bi.rnd.int.mu = predict(bi.rnd.int),
         bi.rnd.slope.mu = predict(bi.rnd.slope))

db.test <- db.test |>
  mutate(bi.all.mu = predict(bi.all,
                             newdata = db.test),
         bi.fixed.mu = predict(bi.fixed,
                               newdata = db.test),
         bi.rnd.int.mu = predict(bi.rnd.int,
                                 newdata = db.test),
         bi.rnd.slope.mu = predict(bi.rnd.slope,
                                   newdata = db.test))
```

```
MSPE <- function(x, y) (mean((x - y)^2))
MAPE <- function(x, y) mean(abs(x - y))
```

```
tb <- tibble(model = c("bi.all", "bi.fixed", "bi.rnd.int", "bi.rnd.slope"),
             AIC.in = map_dbl(bi.models,
                              \(x) AIC(x)),
             BIC.in = map_dbl(bi.models,
                              \(x) BIC(x)),
             MSPE.in = c(MSPE(db.train$AC, db.train$bi.all.mu),
                         MSPE(db.train$AC, db.train$bi.fixed.mu),
                         MSPE(db.train$AC, db.train$bi.rnd.int.mu),
                         MSPE(db.train$AC, db.train$bi.rnd.slope.mu)),
             MAPE.in = c(MAPE(db.train$AC, db.train$bi.all.mu),
                         MAPE(db.train$AC, db.train$bi.fixed.mu),
                         MAPE(db.train$AC, db.train$bi.rnd.int.mu),
                         MAPE(db.train$AC, db.train$bi.rnd.slope.mu)),
             MSPE.out = c(MSPE(db.test$AC, db.test$bi.all.mu),
                          MSPE(db.test$AC, db.test$bi.fixed.mu),
```

Table 6.2: Comparison metrics for all four model using both the training as well as the validation data. AIC is the Akaike information criterion, BIC is the Bayesian information criterion, MSPE is the mean squared prediction error and MAPE is the mean absolute prediction error.

	Training Data				Validation Data	
Model	AIC	BIC	MSPE	MAPE	MSPE	MAPE
bi.all	1,362.21	1,377.09	657.01	19.50	769.90	23.42
bi.fixed	1,293.31	1,391.55	277.65	12.43	743.98	22.80
bi.rnd.int	1,310.72	1,328.41	298.22	13.06	675.22	21.41
bi.rnd.slope	1,312.77	1,336.36	280.51	12.85	728.66	22.12

```
                        MSPE(db.test$AC, db.test$bi.rnd.int.mu),
                        MSPE(db.test$AC, db.test$bi.rnd.slope.mu)),
            MAPE.out = c(MAPE(db.test$AC, db.test$bi.all.mu),
                        MAPE(db.test$AC, db.test$bi.fixed.mu),
                        MAPE(db.test$AC, db.test$bi.rnd.int.mu),
                        MAPE(db.test$AC, db.test$bi.rnd.slope.mu)))
```

```
tb |>
  kbl(booktabs = TRUE,
      digits = c(0, rep(2, 6)),
      format.args = list(big.mark = ","),
      col.names = c("Model", "AIC", "BIC", "MSPE", "MAPE", "MSPE", "MAPE")) |>
  add_header_above(c(" " = 1, "Training Data" = 4, "Validation Data" = 2)) |>
  kable_classic()
rm(tb)
```

From these models we can compute predictions for the train data as well as the validation data and compare with the actual observations. Table 6.2 shows the following performance measures on the training data:

1. AIC, Akaike's Information Criterion,
2. BIC, Bayesian Information Criterion,
3. MSPE, Mean Squared Prediction Error, and
4. MAPE, Mean Absolute Prediction Error.

For the validation data we use MSPE and MAPE.

The worst model, across all measures is `bi.all` where we ignored the variable `TOWNCODE`. Many modelers use AIC/BIC as part of their model selection criteria. Across the above models,

AIC would select the `bi.fixed` model and BIC would go with `bi.rnd.int`. Both measures of prediction error, computed on the training data, suggest that the fixed effects model, `bi.fixed`, is the better choice. We know full well that relying on any measure of performance based on the training data may lead us astray!

On the validation data, we can see that both prediction error measures would select the random intercepts model, `bi.rnd.int`, as the best choice. Therefore, our selected model is the random intercept model

$$\mathbb{E}[\text{AC}] = (\beta_0 + \alpha_i) + \beta_1 \cdot \text{PCI} + \beta_2 \cdot \log(\text{PPSM}) + \beta_3 \cdot \text{YR}.$$

Figure 6.8 shows the actual data from 1993 to 1998 together with the predictions for 1998 based on the training data 1993–1997. The predictions are in an open red colored circle and are joined to their actual values by a pink line.

```
gap <- 0.2
db.all <- bind_rows(db.train, db.test)
db.all <- db.all |>
  mutate(YEAR = ifelse(YEAR == 1998, 1998 - gap, YEAR))
db.tmp <- db.test
db.tmp <- bind_rows(db.tmp, db.tmp)
db.tmp[1:29, "YEAR"] <- 1998 - gap
db.tmp[30:58, "YEAR"] <- 1998 + gap
db.tmp[30:58, "AC"] <- db.tmp[1:29, "bi.rnd.int.mu"]

rm(gap)
```

```
ggplot(data = db.all,
       mapping = aes(x = YEAR,
                     y = AC,
                     group = TOWNCODE)) +
  geom_line(color = "gray") +
  geom_line(data = db.tmp,
            mapping = aes(x = YEAR,
                          y= AC,
                          group = TOWNCODE),
            color = "pink") +
  geom_point() +
  geom_point(data = db.tmp[1:29,],
            mapping = aes(x = YEAR,
                          y = AC,
                          group = TOWNCODE),
            color = "gray") +
```

```
  geom_point(data = db.tmp[30:58,],
             mapping = aes(x = YEAR,
                           y = AC,
                           group = TOWNCODE),
             color = "red",
             pch = 1) +
  labs(x = "Calendar Year",
       y = "Average Claim Costs")

rm(db.all, db.tmp)
```

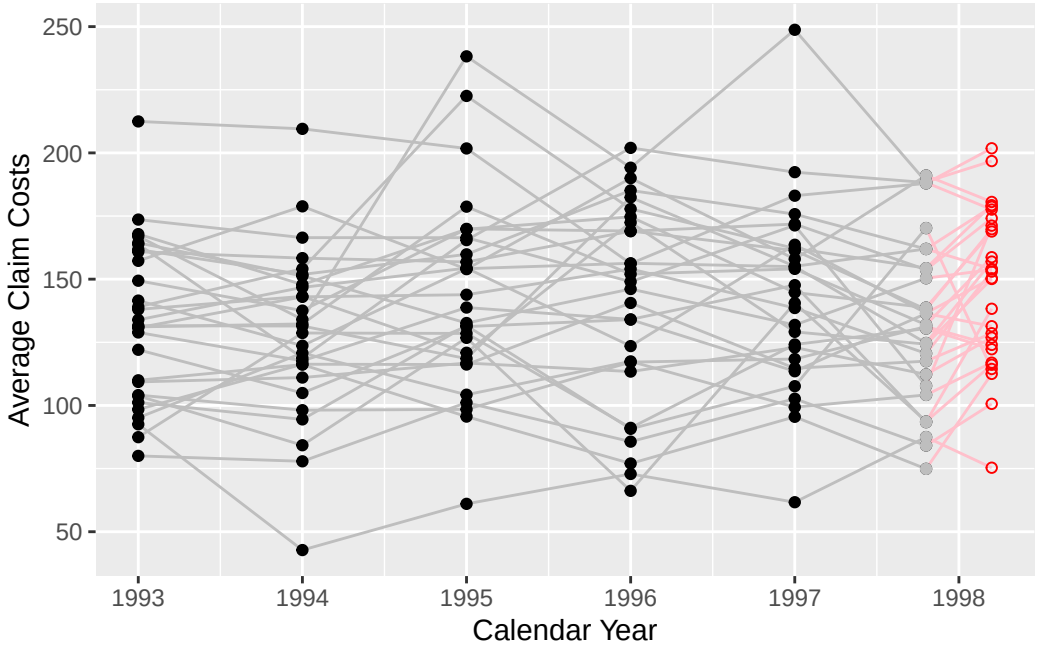


Figure 6.8: Time series plot of the actual claim costs across calendar years for the 29 randomly selected towns of Massachusetts (shown in black) for the training data (1993–1997). The actual 1998 experience is shown in gray along with the predicted values from the random intercept model (shown in red).

6.2 Hospital Length of Stay

```
db <- read_csv("medpar.csv",
               col_types = "fiiiiiiiii")
```

```
Registered S3 method overwritten by 'bit':
  method   from
  print.ri gamlss
```

```
db <- db |>
  mutate(type = case_when(type1 == 1 ~ "Elective",
                          type2 == 1 ~ "Urgent",
                          type3 == 1 ~ "Emergency"))
db$type <- factor(db$type,
                  levels = c("Elective", "Urgent", "Emergency"))
```

The length of stay at a hospital is an important measure for health organizations to track and to understand some of the patient characteristics that may influence it. The following example comes from Hilbe (2007) and the data is a random sample of patients drawn from the state of Arizona Medicare program for a single undisclosed diagnostic group. The response variable is length of stay, `los`, a count of the number of days a patient spent in the hospital. The following explanatory variables are available:

1. `provnum`: identifier for the medical provider
2. `hmo`: does the patient belong to a Health Maintenance Organization?
3. `white`: does the patient self-identify primarily as a Caucasian?
4. `type1`: was the admission to the hospital **elective**?
5. `type2`: was the admission to the hospital **urgent**?
6. `type3`: was the admission to the hospital **emergency**?
7. `age`: the age group of the patient (1 to 9)
8. `age80`: patient is older than or equal to 80?
9. `died`: did the patient die at the hospital?

```
tb <- db |>
  group_by(provnum) |>
  summarize(sz = n(),
            mean.los = mean(los),
            mean.age = mean(age),
            count.white = sum(white),
            count.type1 = sum(type1),
            count.type2 = sum(type2),
            count.type3 = sum(type3)) |>
  arrange(desc(mean.los))
```

All variables are indicator variables (1/0) except for `provnum` and `age`. The data has 1,495 observations and 54 unique medical providers. We would like to understand how these explanatory variables could help us predict the length of stay for a newly admitted patient.

The response variable is counting the days that a patient spends in the hospital and so perhaps we should use a Poisson distribution for the length of stay. But the Poisson distribution would not be entirely appropriate because length of stay can never be zero. What would work is the zero-truncated Poisson distribution defined as follows: we say that N is a zero-truncated Poisson random variable with parameter $\lambda > 0$ if

$$\text{Prob}(N = y) = \frac{1}{1 - e^{-\lambda}} \frac{e^{-\lambda}\lambda^y}{y!} \qquad \text{for } y = 1, 2, ...$$

Exercise

Show that the zero-truncated Poisson distribution is a member of the exponential family.

Solution

To show that a zero-truncated Poisson distribution is a member of the exponential family we have to rewrite the density function in the form

$$a(y, \phi) \exp\left[\frac{y\theta - \kappa(\theta)}{\phi}\right],$$

where ϕ is a dispersion parameter and $a(y, \phi)$ is a normalizing constant. The mean of the distribution is given be the first derivative of $\kappa(\theta)$ and the variance function is the derivative of the mean with respect to θ; that is, $\kappa''(\theta)$.
We can rewrite the density function as follows:

$$\frac{e^{-\lambda}\lambda^y}{y!(1 - e^{-\lambda})} = \frac{1}{y!}\frac{\lambda^y}{(e^{\lambda} - 1)} = \frac{1}{y!} e^{y\log(\lambda) - \log(e^{\lambda} - 1)}.$$

Hence we have $\theta = \log(\lambda)$, $\kappa(\theta) = \log(e^{e^{\theta}} - 1)$, $a(y, \phi) = 1/y!$, and $\phi = 1$.
Therefore, the zero-truncated Poisson distribution is a member of the exponential family of distributions. The mean of the distribution is

$$\kappa'(\theta) = \frac{e^{e^{\theta}} e^{\theta}}{e^{e^{\theta}} - 1} = \frac{e^{\lambda}\lambda}{e^{\lambda} - 1} = \frac{\lambda}{1 - e^{-\lambda}} = \mu,$$

and variance given in terms of the mean μ as

$$\kappa''(\theta) = \frac{e^{\theta + e^{\theta}}\left[e^{e^{\theta} - e^{\theta} - 1}\right]}{\left(e^{e^{\theta}} - 1\right)^2} = \mu\left[1 + W(-\mu e^{-\mu})\right],$$

where W is the principal branch of Lambert's W-function.

Exercise

Compare the Poisson and zero-truncated Poisson distributions with means equal to 1.5, 2.5, and 3.5. Based on this information, what would you conclude in terms of the usefulness of the zero-truncated Poisson distribution?

Solution

The Poisson distribution with parameter λ has mean equal to lambda. But the Zero-Truncated Poisson distribution with parameter λ has a mean equal to $\lambda/(1-e^{-\lambda})$, and thus we need to find the appropriate value of λ to give us a Zero-Truncated Poisson distribution with the correct mean. Hence, if μ is the mean we want for the ZT Poisson we need to solve the equation

$$\mu = \frac{\lambda}{1-e^{-\lambda}}.$$

Solving this equation for λ in terms of μ requires the use of Lambert's W function. Lambert's W function is also known as the product logarithm function and is the inverse of the function $f(x) = xe^x$; that is,

$$W(f(x)) = W(xe^x) = x.$$

For real numbers, x, the function $W(x)$ has two principal branches; one for $x > -1/e$ and the other one for $-1/e \leq x < 0$.
The solution we are seeking is

$$\lambda = \mu + W(-\mu e^{-\mu}),$$

with W being the principal branch.
Create a function to return a table with the values the random variable can take, say zero to 9, and their associated probabilities.

```
f <- function(mu = 1.5, x = 0:9, zp = FALSE) {
  lambda <- mu
  if(zp) {
    lambda <- mu + pracma::lambertWp(-mu * exp(-mu))
  }
  ans <- tibble(x = x,
                prob = dpois(x, lambda = lambda))
  if (zp) {

    ans$prob <- ans$prob / (1 - exp(-lambda))
    ans <- filter(ans, x > 0)
  }
  return(ans)
}
```

Plot the probabilities.

```
ggplot(mapping = aes(x = x,
                     y = prob)) +
  geom_line(data = f(1.5), color = "gray") +
  geom_point(data = f(1.5), color = "red") +
  geom_line(data = f(1.5, zp = TRUE), color = "gray") +
  geom_point(data = f(1.5, zp = TRUE), color = "red", pch = 1) +
  geom_line(data = f(2.5), color = "gray") +
  geom_point(data = f(2.5), color = "blue") +
  geom_line(data = f(2.5, zp = TRUE), color = "gray") +
  geom_point(data = f(2.5, zp = TRUE), color = "blue", pch = 1) +
  geom_line(data = f(3.5), color = "gray") +
  geom_point(data = f(3.5), color = "purple") +
  geom_line(data = f(3.5, zp = TRUE), color = "gray") +
  geom_point(data = f(3.5, zp = TRUE), color = "purple", pch = 1) +
  scale_x_continuous(breaks = seq(0, 8, by = 2),
                     labels = seq(0, 8, by = 2)) +
  labs(x = "Random Variable",
       y = "Probability")
rm(f)
```

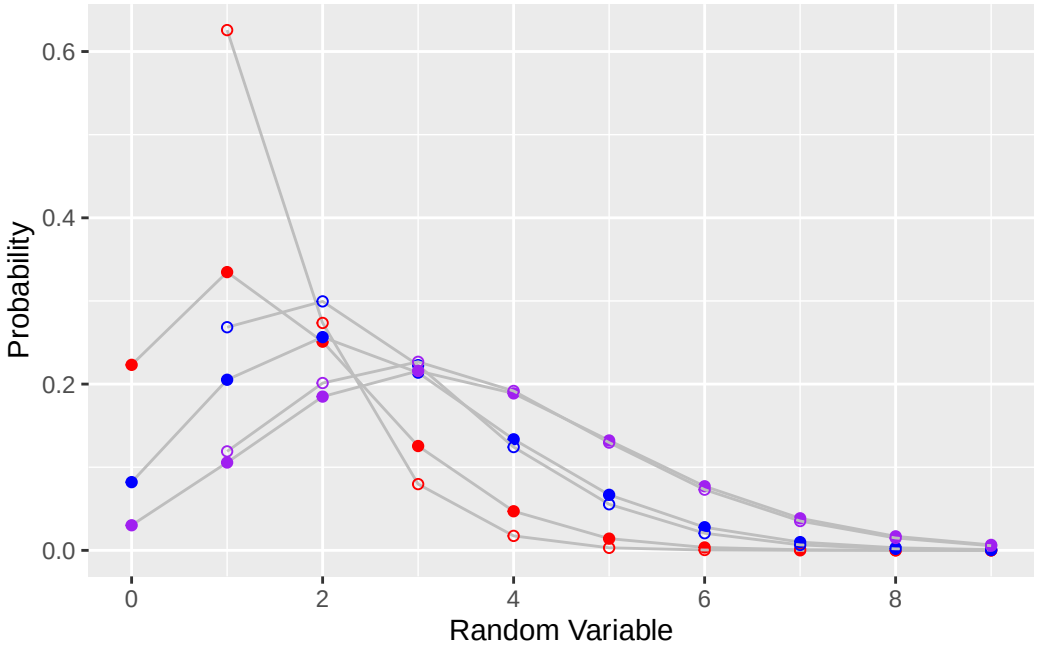


Figure 6.9: Poisson and Zero-truncated Poisson probabilities for means equal to 1.5, 2.5, and 3.5. As the mean increases, the difference between the Poisson and the Zero-Truncated Poisson distributions narrows quickly. The filled in circles represent the Poisson distribution and the open circles correspond to the Zero-Truncated Poisson distribution. Red colored points correspond to a mean of 1.5, blue corresponds to 2.5, and purple has mean of 3.5.

From Figure 6.9 we can see that for small means, the probabilities between the Poisson distribution and the Zero-Truncated Poisson distribution differ significantly. But as the mean of the random variables increase, the probabilities get closer and closer to each other. If the mean length of stay is larger than, say 4 or 5, then using the Poisson distribution instead of the zero-truncated Poisson distribution would yield nearly identical results.

Exploratory Data Analysis

On average, we should have about 30 patients per provider, but the data shows a lot of variation: we have a provider with a single patient and another with with 92 patients. The overall mean length of stay in the hospital is equal to 9.9 days. Table 6.3 shows the top 5 and bottom 5 providers in terms of their mean length of stay along with the number of patients, their mean age group, and how many of them consider themselves Caucasian, and the type of admission to the hospital.

Table 6.3: Top 5 and bottom 5 providers sorted by mean length of stay in decreasing order. Also showing the number of patients for each provider, their mean age group, and the number of patients who consider themselves Caucasian, and the number by type of admission to the hospital.

Provider	Number of Patients	Mean Length of Stay	Mean Age Group	Count of White	Type of Admission		
					Elective	Urgent	Emergency
32003	2	47.5	5.0	2	0	2	0
32002	10	28.3	5.3	9	0	0	10
32000	38	26.6	4.8	32	0	0	38
30073	4	21.8	6.5	0	2	2	0
30078	3	18.3	5.0	0	2	1	0
30025	3	4.7	4.3	2	3	0	0
30067	5	4.4	5.4	5	5	0	0
30060	2	3.5	5.5	2	1	1	0
30044	2	3.0	6.5	2	0	2	0
30068	1	2.0	4.0	1	1	0	0

```
tb[c(1:5, 50:54),] |>
  kbl(booktabs = TRUE,
      col.names = c("Provider", "Patients", "of Stay", "Age Group",
                    "White", "Elective", "Urgent", "Emergency"),
      digits = c(0,0,1,1,0,0,0,0),
      align = rep("r", 8)) |>
  add_header_above(c(" " = 1, "Number of" = 1, "Mean Length" = 1,
                     "Mean" = 1, "Count of" = 1, "Type of Admission" = 3),
                   line = FALSE,
                   align = c(rep("r", 5), "c")) |>
  kable_classic()
```

```
db |>
  group_by(age, type) |>
  summarize(mn.los = round(mean(los),1),
            .groups = "drop") |>
  pivot_wider(names_from = type,
              values_from = mn.los) |>
  kbl(booktabs = TRUE,
      digits = c(0, 1, 1, 1),
      col.names = c("Age Group", "Elective", "Urgent", "Emergency")) |>
  add_header_above(c(" " = 1, "Type of Admission" =  3)) |>
```

Table 6.4: Average length of stay by age group and type of admission. Note that in nearly all cases elective admissions are shortest and emergency admission are the longest.

	Type of Admission		
Age Group	Elective	Urgent	Emergency
1	13.8	9.5	NA
2	7.5	16.2	20.6
3	9.7	11.4	12.6
4	9.1	11.0	18.5
5	8.3	10.8	22.7
6	9.1	11.4	16.9
7	8.9	10.6	15.4
8	7.8	12.1	16.3
9	8.7	7.7	17.3

```
kable_classic()
```

The length of stay is strongly influenced by the type of admission but not the age of the patient. Table 6.4 displays the average hospital stay by age group and type of admission. As we read down the columns, there are no clear upwards or downwards trends; therefore, age group does not seem to be related to the number of days a patient stays at the hospital. Reading horizontally across, we find that nearly all elective admissions have the shortest stays, emergency admissions have the longest stays, and urgent admissions are in-between. Also note that for age group 1, the average length of stay for elective admissions is very high at 13.8 days. But this high mean value is based only on 4 observations and thus we should be careful about using these levels as base levels in our estimation of models.

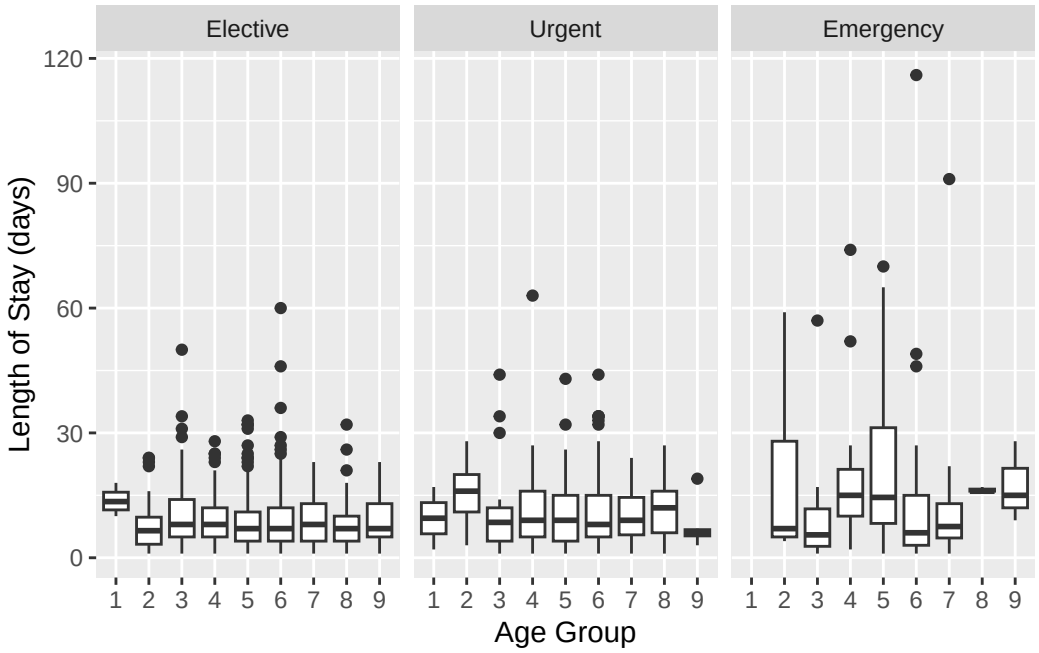
Exercise

Further explore the relationship between `age` and `los`. Do not treat age as a numeric variable and take into account the type of admission. Does the data have many outliers?

Solution

The following display shows `age` as a categorical variable and length of stay, `los`, using boxplots for each type of admission. Note that most of the outliers are for emergency and urgent admissions.

```
ggplot(data = db,
       mapping = aes(x = factor(age),
                     y = los)) +
  facet_wrap(~ type) +
  geom_boxplot() +
  labs(x = "Age Group",
       y = "Length of Stay (days)")
```



Modeling Length of Stay

Our response variable is length of stay, `los`, a count of the number of days a patient remained hospitalized. For now we ignore the information supplied by the variable, `provnum` (medical provider), because this variable has a large number of levels: (54).

```
db <- db |>
  mutate(age.cat = factor(age,
                          levels = c(6, 1:5, 7:9)))
```

For our first model, we shall use the Poisson distribution and include `hmo`, `white`, `type`, and `age.cat` as explanatory variables. The variable `age.cat` is a categorical version of `age`, and we have selected age group 6 as the base level because this level has the largest number of

observations. For type of admission, `type`, we have selected level elective as our base for the same reason.

The model fit is summarized below.

```
los.pois <- glm(los ~ type + white + hmo + age.cat,
                data = db,
                family = poisson(link = "log"))
summary(los.pois)
```

```
Call:
glm(formula = los ~ type + white + hmo + age.cat, family = poisson(link = "log"),
    data = db)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    2.36401    0.03339  70.795  < 2e-16 ***
typeUrgent     0.21949    0.02113  10.388  < 2e-16 ***
typeEmergency  0.70906    0.02620  27.066  < 2e-16 ***
white         -0.15835    0.02912  -5.437 5.41e-08 ***
hmo           -0.07505    0.02400  -3.128  0.00176 **
age.cat1       0.12164    0.11900   1.022  0.30669
age.cat2      -0.09817    0.04645  -2.113  0.03456 *
age.cat3       0.01670    0.03032   0.551  0.58184
age.cat4      -0.01421    0.02536  -0.560  0.57524
age.cat5      -0.03259    0.02507  -1.300  0.19360
age.cat7      -0.05237    0.02921  -1.793  0.07295 .
age.cat8      -0.09316    0.03895  -2.392  0.01678 *
age.cat9      -0.09152    0.05179  -1.767  0.07716 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 8901.1  on 1494  degrees of freedom
Residual deviance: 8125.4  on 1482  degrees of freedom
AIC: 13867

Number of Fisher Scoring iterations: 5
```

Note that it appears that `type`, `white`, and `hmo` are all statistically significant variables. Many of the estimated coefficients for `age.cat` have a negative sign but do not appear to be significant

at the 5% level. Relative to age group 6, every other group appears to have either a similar length of stay, or a lower one (groups 2 and 8).

```
db2 <- db |>
  mutate(mu = predict(los.pois, type = "response"),
         rD = resid(los.pois, type = "deviance"),
         rQ = qresid(los.pois))
```

```
p <- ggplot(data = db2,
            mapping = aes(x = mu, y = rD,
            color = type)) +
  geom_point(alpha = 0.4) +
  labs(x = "Fitted Values",
       y = "Deviance Residuals") +
  theme(legend.position = "none")
q <- ggplot(data = db2,
            mapping = aes(x = mu, y = abs(rD),
                          color = type)) +
  geom_point(alpha = 0.4) +
  labs(x = "Fitted Values",
       y = "|Deviance Residuals|") +
  theme(legend.position = "none")
r <- ggplot(data = db2,
       mapping = aes(sample = rD)) +
  geom_qq() + geom_qq_line() +
  labs(x = "Theoretical Quantiles",
       y = "Sample Quantiles")
s <- ggplot(data = db2,
            mapping = aes(x = rD)) +
  geom_histogram() +
  labs(x = "Deviance Residuals")
(p + q) / (r + s)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

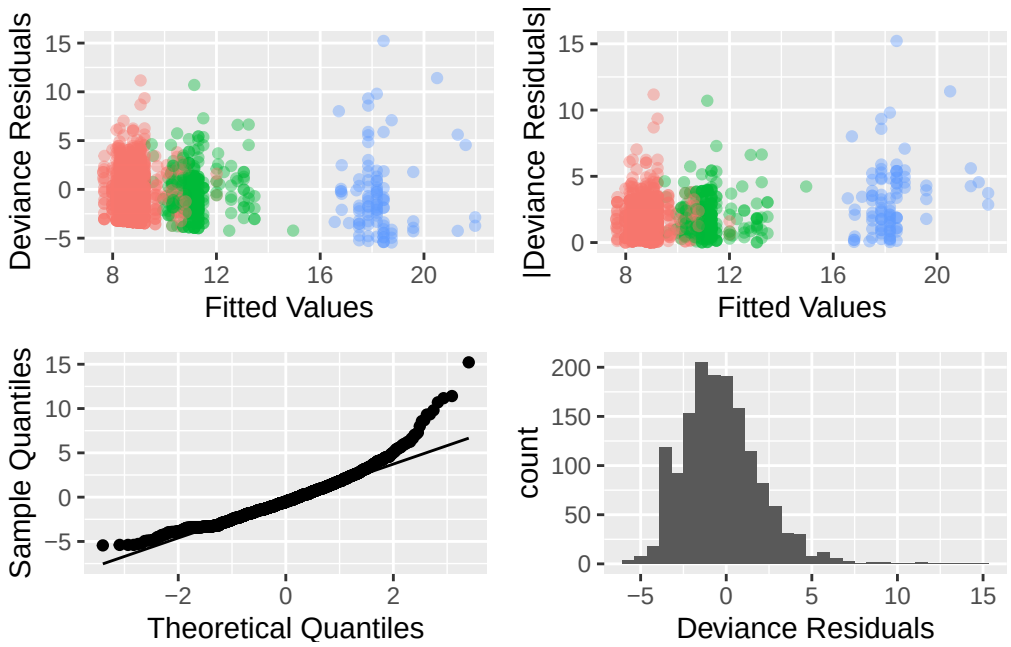


Figure 6.10: Diagnostic plots for the Poisson model predicting length of stay based on type of admission, self-reported race, age category, and whether or not the patient is a member of a health maintenance organization. The clusters, from left to right in the upper panels, correspond to elective, urgent, and emergency admissions to the hospital.

The above model unfortunately does not fit the data well. Figure 6.10 shows several diagnostic plots. The deviance residuals should be approximately normally distributed. Their mean is -0.281 and their standard deviation is 2.315. Clearly we have some very large residuals. The QQ-plot shown on the bottom left-hand panel shows that the deviance residuals have much thicker tails than the normal distribution and the bottom right-hand panel shows a non-symmetrical distribution for the deviance residuals. Finally, the upper right-hand panel, which displays the absolute value of the deviance residuals against the fitted values, shows an increasing trend pointing that the variance increases as the fitted values increase.

The cluster of observations seen in the upper panels arise because the different types of admission to the hospital (elective, urgent, and emergency) have little overlap in the response variable. There is also a clear decreasing pattern in the upper left-hand panel. The model overpredicts many of the emergency admissions.

Also if the model fit had been good, then we would expect an estimate of the dispersion parameter to be close to 1. Here both the mean deviance estimate as well as the Pearson estimate of the dispersion parameter are well above 1, indicating overdispersion.

```
c("Mean Dev. Estimate" = deviance(los.pois) / df.residual(los.pois),
  "Pearson Estimate" = sum(resid(los.pois, type = "pearson")^2) /
    df.residual(los.pois))
```

```
Mean Dev. Estimate   Pearson Estimate
          5.482701           6.257832
```

But is the overdispersion real or apparent? Apparent overdispersion can arise when our modeling of the data is deficient. For example, not including an important explanatory variable in our model or using the wrong link function or the presence of outliers might show that the estimate of the dispersion parameter is greater than 1 leading us to think that the data is overdispersed. But once we fix our model, then the estimate of the dispersion parameter falls back close to unity.

Exercise

Outlier observations violate one of the most important assumptions in regression analysis; namely, that all of the observations come from the same data generation process.
Repeat the above analysis, but first remove the largest 5% of observations in terms of length of stay. The Pearson estimate of the dispersion parameter should now be smaller, but it is still well above 1.

Solution

Remove the top 5% of observations and fit the Poisson model to the new dataset `dta`.

```
dta <- filter(db,
              db$los < quantile(db$los, probs = 0.95))
los.pois.no <- glm(los ~ type + white + hmo + age.cat,
                   data = dta,
                   family = poisson(link = "log"))
summary(los.pois.no)
```

```
Call:
glm(formula = los ~ type + white + hmo + age.cat, family = poisson(link = "log"),
    data = dta)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   2.15992    0.03889  55.541  < 2e-16 ***
typeUrgent    0.11565    0.02388   4.844 1.27e-06 ***
```

```
typeEmergency  0.15445    0.03909   3.952 7.76e-05 ***
white         -0.11289    0.03406  -3.314 0.000918 ***
hmo           -0.02104    0.02544  -0.827 0.408236
age.cat1       0.34929    0.11998   2.911 0.003600 **
age.cat2      -0.15257    0.05733  -2.661 0.007781 **
age.cat3       0.04527    0.03467   1.306 0.191613
age.cat4       0.12751    0.02834   4.499 6.84e-06 ***
age.cat5      -0.00572    0.02896  -0.198 0.843411
age.cat7       0.08706    0.03211   2.711 0.006709 **
age.cat8       0.03440    0.04200   0.819 0.412732
age.cat9       0.05025    0.05584   0.900 0.368183
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 4908.2  on 1410  degrees of freedom
Residual deviance: 4801.4  on 1398  degrees of freedom
AIC: 10096

Number of Fisher Scoring iterations: 5
```

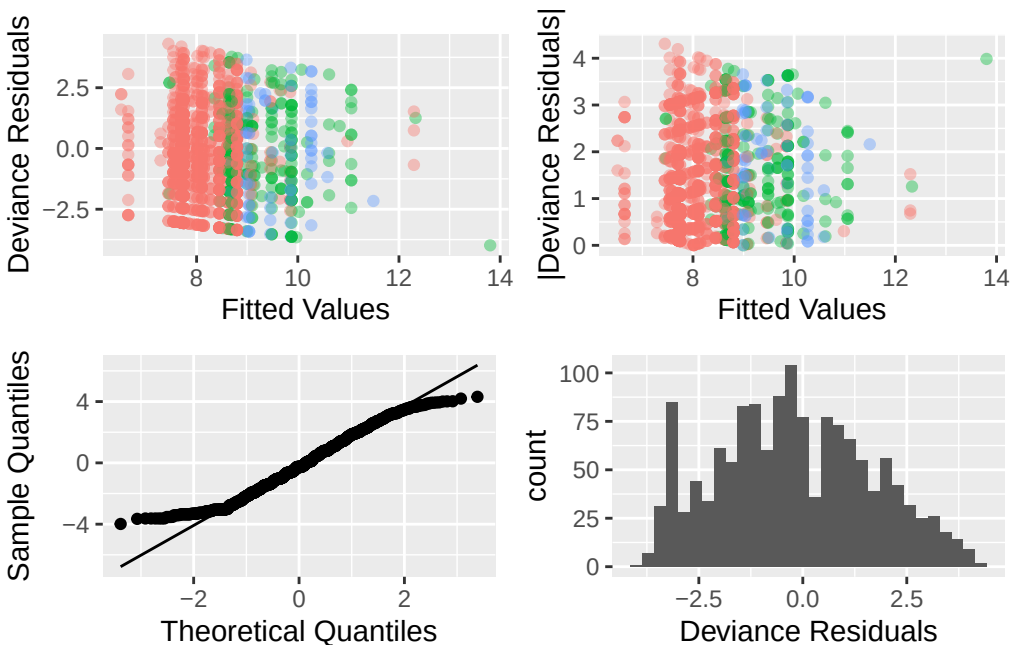
Compute fitted values and deviance residuals.

```
dta2 <- dta |>
  mutate(mu = predict(los.pois.no, type = "response"),
         rD = resid(los.pois.no, type = "deviance"))
```

Generate the diagnostic plots.

```
p <- ggplot(data = dta2,
            mapping = aes(x = mu, y = rD,
            color = type)) +
  geom_point(alpha = 0.4) +
  labs(x = "Fitted Values",
       y = "Deviance Residuals") +
  theme(legend.position = "none")
q <- ggplot(data = dta2,
            mapping = aes(x = mu, y = abs(rD),
                          color = type)) +
  geom_point(alpha = 0.4) +
  labs(x = "Fitted Values",
       y = "|Deviance Residuals|") +
  theme(legend.position = "none")
r <- ggplot(data = dta2,
       mapping = aes(sample = rD)) +
  geom_qq() + geom_qq_line() +
  labs(x = "Theoretical Quantiles",
       y = "Sample Quantiles")
s <- ggplot(data = dta2,
            mapping = aes(x = rD)) +
  geom_histogram() +
  labs(x = "Deviance Residuals")
(p + q) / (r + s)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The deviance residuals versus the fitted values (upper left-hand panel) shows that many points seem to be on straight lines with a negative slope. These 'patterns' are an artifact that our response variable is an integer and they arise for count and binomial (with response variable 0 or 1) models. For these models, it is recommended to use quantile residuals (Dunn and Smyth 1996). You can calculate quantile residuals for generalized linear models via the function `qresid()` that is available in the `statmod` package.
The QQ-plot shows that both tails of the deviance residuals are too thin compared to the normal distribution. The upper right-hand panel depicts a funnel type shape fanning inwards as the fitted values increase.
The estimates of the dispersion parameter are:

```
c("Mean Dev. Estimate" = deviance(los.pois.no) / df.residual(los.pois.no),
  "Pearson Estimate" = sum(resid(los.pois.no, type = "pearson")^2) /
    df.residual(los.pois.no))
```

```
Mean Dev. Estimate   Pearson Estimate
          3.434447           3.305389
```

Both values are much smaller than our previous model with all the observations, but they are still much larger than the theoretical value of 1.

The following exercise shows how apparent overdispersion can arise if we do not have the appropriate explanatory variables.

Exercise

Generate a dataset with a response variable that is Poisson distributed and a categorical variable with four levels. The response variable, `skip`, is the number of classes students at a university skip in one semester. The categorical variable, `class`, classifies students by how many years they have already been at the university: 0 (freshman), 1 (sophomore), 2 (junior), or 3 (senior).
Each group should have the same number of students and the mean number of classes skipped for freshmen is 4.5, sophomores is 1.5, juniors is 1.5, and seniors is 6.5.

1. Fit a Poisson generalized linear model to the data ignoring the `class` variable and compute the Pearson estimate of the dispersion parameter. Is overdispersion present in this dataset?
2. Fit a Poisson generalized linear model including the `class` variable and recompute the Pearson estimate of the dispersion parameter. Does the result indicate that the data is overdispersed?

Solution

Set a seed for the random number generator so we can reproduce our computations and generate our data based on four classes, each Poisson distributed with a different mean.

```
set.seed(12853)
N <- 100
dta <- tibble(skip = c(rpois(N, lambda = 4.5),
                       rpois(N, lambda = 2.5),
                       rpois(N, lambda = 1.5),
                       rpois(N, lambda = 6.5)),
              class = c(rep("freshman", N),
                        rep("sophomore", N),
                        rep("junior", N),
                        rep("senior", N)))
```

Ignoring the class standing of a student we fit a Poisson model across all observations and compute the Pearson estimate, $\widehat{\phi}$, of the dispersion parameter.

```
m1 <- glm(skip ~ 1,
          data = dta,
          family = poisson(link = "log"))
(phi.hat <- sum(resid(m1, type = "pearson")^2) / df.residual(m1))
```

```
[1] 2.11863
```

The value of $\hat{\phi}$ is well above its theoretical value of 1, indicating that the data is overdispersed. Next, we include the explanatory variable `class` and recompute the Pearson estimate of the dispersion parameter yielding a value that is much closer to 1.

```
m2 <- glm(skip ~ class,
          data = dta,
          family = poisson(link = "log"))
(phi.hat <- sum(resid(m2, type = "pearson")^2) / df.residual(m2))
```

```
[1] 0.9422353
```

Therefore, we conclude that the overdispersion we saw earlier is apparent.

As a consequence of the overdispersion, the estimated standard errors for our coefficients are too narrow and this might lead us to infer that some explanatory variables are significant when in fact they are not. One may compensate for the overdispersion by inflating the standard error with a multiplicative factor equal to the square root of the estimated dispersion parameter. For our data this factor would be equal to approximately 2.5 and applying it to our fitted Poisson model would render all of the estimated coefficients for `age.cat` not significant at the 5% level as well as the indicator for the health maintenance organization, `hmo`.

Exercise

Adjust the standard errors by fitting a quasi-Poisson model and verify the claims made in the previous paragraph.

Solution

```
los.qpoi <- glm(los ~ type + white + hmo + age.cat,
                data = db,
                family = quasipoisson(link = "log"))
summary(los.qpoi)
```

```
Call:
glm(formula = los ~ type + white + hmo + age.cat, family = quasipoisson(link = "log"),
    data = db)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.36401    0.08353  28.300  < 2e-16 ***
```

```
typeUrgent     0.21949    0.05286   4.153 3.48e-05 ***
typeEmergency  0.70906    0.06553  10.820  < 2e-16 ***
white         -0.15835    0.07285  -2.174   0.0299 *
hmo           -0.07505    0.06003  -1.250   0.2114
age.cat1       0.12164    0.29769   0.409   0.6829
age.cat2      -0.09817    0.11620  -0.845   0.3983
age.cat3       0.01670    0.07584   0.220   0.8258
age.cat4      -0.01421    0.06343  -0.224   0.8228
age.cat5      -0.03259    0.06272  -0.520   0.6034
age.cat7      -0.05237    0.07306  -0.717   0.4736
age.cat8      -0.09316    0.09745  -0.956   0.3392
age.cat9      -0.09152    0.12954  -0.707   0.4800
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 6.257848)

    Null deviance: 8901.1  on 1494  degrees of freedom
Residual deviance: 8125.4  on 1482  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

```
rm(los.qpoi)
```

Real overdispersion may also arise because the observations in our data are not independent of each other. That is, there are clusters of observations that are similar to each other and thus would violate the assumption that they are sampled independently. In our current application, patients belonging to a medical provider may have other (unobserved) characteristics in common that creates a cluster that contributes to the overdispersion.

There are 54 unique medical providers in the dataset. Many of them (11) with less than 5 data points each. Hence, estimating a Poisson model with `provnum` as an explanatory variable may yield estimated coefficients for the medical providers that may be extreme because they are based on a small number of observations. Estimating a Poisson generalized linear model with `provnum` as an explanatory variable yields the following summary:

```
los.pois.provnum <- glm(los ~ type + white + hmo + age.cat + provnum,
                        data = db,
                        family = poisson(link = "log"))
(sm <- summary(los.pois.provnum))
```

```
Call:
glm(formula = los ~ type + white + hmo + age.cat + provnum, family = poisson(link = "log"),
    data = db)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    2.00488    0.06025  33.277  < 2e-16 ***
typeUrgent     0.23065    0.02506   9.204  < 2e-16 ***
typeEmergency  0.09751    0.04993   1.953 0.050835 .
white         -0.01110    0.03205  -0.346 0.729081
hmo           -0.09637    0.02603  -3.702 0.000214 ***
age.cat1       0.24808    0.12120   2.047 0.040662 *
age.cat2      -0.06257    0.04726  -1.324 0.185514
age.cat3      -0.02743    0.03128  -0.877 0.380545
age.cat4      -0.02967    0.02594  -1.144 0.252772
age.cat5      -0.05041    0.02556  -1.973 0.048538 *
age.cat7      -0.07119    0.02988  -2.383 0.017182 *
age.cat8      -0.13813    0.03965  -3.483 0.000495 ***
age.cat9      -0.07188    0.05272  -1.363 0.172781
provnum30002   0.28683    0.06480   4.426 9.58e-06 ***
provnum30003   0.06364    0.15630   0.407 0.683877
provnum30006   0.31611    0.06218   5.084 3.70e-07 ***
provnum30007  -0.11236    0.12131  -0.926 0.354327
provnum30008   0.10272    0.08681   1.183 0.236687
provnum30009   0.50116    0.08813   5.687 1.29e-08 ***
provnum30010   0.39929    0.06552   6.094 1.10e-09 ***
provnum30011   0.37310    0.07103   5.252 1.50e-07 ***
provnum30012  -0.16181    0.10005  -1.617 0.105826
provnum30013   0.34277    0.06421   5.338 9.40e-08 ***
provnum30014   0.04435    0.06758   0.656 0.511686
provnum30016   0.56156    0.06725   8.350  < 2e-16 ***
provnum30017  -0.37225    0.10082  -3.692 0.000222 ***
provnum30018   0.22986    0.07877   2.918 0.003521 **
provnum30019   0.01020    0.10322   0.099 0.921248
provnum30022   0.22303    0.06876   3.244 0.001180 **
provnum30023   0.48433    0.15370   3.151 0.001627 **
provnum30024   0.25737    0.07018   3.667 0.000245 ***
provnum30025  -0.41530    0.27244  -1.524 0.127419
provnum30030   0.31769    0.07399   4.294 1.76e-05 ***
provnum30033   0.08566    0.35749   0.240 0.810630
provnum30035  -0.26337    0.17333  -1.519 0.128641
provnum30036   0.38796    0.10186   3.809 0.000140 ***
```

```
provnum30037  -0.14532    0.09916  -1.465 0.142793
provnum30038   0.34154    0.06731   5.074 3.89e-07 ***
provnum30043  -0.18059    0.11190  -1.614 0.106569
provnum30044  -1.09085    0.41219  -2.646 0.008134 **
provnum30055   0.18654    0.07591   2.457 0.014002 *
provnum30059   0.16790    0.17885   0.939 0.347845
provnum30060  -0.84098    0.38162  -2.204 0.027547 *
provnum30061   0.38228    0.05985   6.388 1.68e-10 ***
provnum30062  -0.15580    0.10219  -1.525 0.127333
provnum30064   0.37192    0.07454   4.990 6.05e-07 ***
provnum30065   0.38364    0.06843   5.606 2.07e-08 ***
provnum30067  -0.46192    0.21922  -2.107 0.035105 *
provnum30068  -1.27096    0.70906  -1.792 0.073057 .
provnum30069  -0.02031    0.09571  -0.212 0.831949
provnum30073   0.98774    0.12270   8.050 8.27e-16 ***
provnum30078   0.84927    0.14678   5.786 7.22e-09 ***
provnum30080   0.20900    0.08087   2.584 0.009753 **
provnum30083   0.17626    0.09096   1.938 0.052642 .
provnum30084   0.47395    0.16488   2.874 0.004047 **
provnum30085   0.07268    0.08337   0.872 0.383291
provnum30086   0.12834    0.08538   1.503 0.132803
provnum30087   0.37814    0.07492   5.048 4.47e-07 ***
provnum30088   0.24951    0.06398   3.900 9.62e-05 ***
provnum30089   0.18284    0.06584   2.777 0.005484 **
provnum30092  -0.06362    0.10993  -0.579 0.562791
provnum30093   0.28739    0.07169   4.009 6.10e-05 ***
provnum30094   0.12694    0.11596   1.095 0.273659
provnum32000   1.22846    0.07710  15.932  < 2e-16 ***
provnum32002   1.29685    0.09218  14.069  < 2e-16 ***
provnum32003   1.65103    0.11696  14.117  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 8901.1  on 1494  degrees of freedom
Residual deviance: 7080.2  on 1429  degrees of freedom
AIC: 12927

Number of Fisher Scoring iterations: 5
```

In Table 6.3 we displayed the top 5 and bottom 5 medical providers by the average length of stay of their patients. Provider 30068 had the lowest average length of stay, only two days,

but has only one patient. The estimated coefficient for this provider is equal to -1.271 with a standard error equal to 0.709. This is the lowest estimated coefficient and it has the highest standard error. Should we completely trust such an estimate? Most likely not.

Exercise

Which medical provider has the largest estimated coefficient? How many patients does this provider has and what is the average length of stay for these patients?

Solution

Provider 32003 has the largest estimated coefficient equal to 1.651 with a standard error of 0.117. The number of patients for this provider is equal to only 2 of them.

Since the number of observations for each medical provider vary significantly we may try to address this by applying some weights to the observations in our dataset. Unfortunately, doing a weighted quasi-Poisson regression where the weights are proportional to the number of patients for each medical provider yields estimated coefficients that are close to the unweighted estimates (except for a handful of providers).

```
wt <- db |>
  group_by(provnum) |>
  summarize(n.obs = n()) |>
  mutate(n.obs = n.obs / max(n.obs))

db2 <- left_join(db, wt, by = "provnum")

m0 <- glm(los ~ type + white + hmo + age.cat + provnum,
            data = db2,
            family = quasipoisson(link = "log"),
          weights = n.obs)
(sm0 <- summary(m0))
```

```
Call:
glm(formula = los ~ type + white + hmo + age.cat + provnum, family = quasipoisson(link = "log
    data = db2, weights = n.obs)

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.965719   0.127297  15.442  < 2e-16 ***
typeUrgent     0.233498   0.057253   4.078 4.79e-05 ***
typeEmergency  0.070619   0.109089   0.647 0.517506
```

```
white          0.053577   0.075344   0.711 0.477139
hmo           -0.095672   0.052505  -1.822 0.068641 .
age.cat1       0.241992   0.282850   0.856 0.392389
age.cat2      -0.111448   0.113022  -0.986 0.324264
age.cat3      -0.004385   0.068611  -0.064 0.949054
age.cat4      -0.054710   0.058995  -0.927 0.353893
age.cat5      -0.074684   0.056869  -1.313 0.189301
age.cat7      -0.112541   0.066895  -1.682 0.092717 .
age.cat8      -0.166028   0.086961  -1.909 0.056432 .
age.cat9      -0.181743   0.126114  -1.441 0.149774
provnum30002   0.289466   0.129847   2.229 0.025950 *
provnum30003   0.077958   0.926951   0.084 0.932988
provnum30006   0.313415   0.120664   2.597 0.009489 **
provnum30007  -0.121399   0.479380  -0.253 0.800119
provnum30008   0.105866   0.238715   0.443 0.657483
provnum30009   0.496947   0.293825   1.691 0.090997 .
provnum30010   0.402419   0.133208   3.021 0.002564 **
provnum30011   0.362179   0.159036   2.277 0.022913 *
provnum30012  -0.174294   0.305841  -0.570 0.568846
provnum30013   0.339376   0.127773   2.656 0.007993 **
provnum30014   0.039314   0.132206   0.297 0.766228
provnum30016   0.569275   0.150209   3.790 0.000157 ***
provnum30017  -0.372642   0.276532  -1.348 0.178015
provnum30018   0.224550   0.199666   1.125 0.260936
provnum30019   0.015502   0.348737   0.044 0.964551
provnum30022   0.236672   0.142651   1.659 0.097316 .
provnum30023   0.480942   1.109426   0.434 0.664713
provnum30024   0.260151   0.152573   1.705 0.088394 .
provnum30025  -0.421529   2.363310  -0.178 0.858462
provnum30030   0.310828   0.175149   1.775 0.076169 .
provnum30033   0.060146   5.410482   0.011 0.991132
provnum30035  -0.254227   1.033011  -0.246 0.805638
provnum30036   0.379939   0.403487   0.942 0.346536
provnum30037  -0.141328   0.304890  -0.464 0.643049
provnum30038   0.338189   0.140090   2.414 0.015900 *
provnum30043  -0.184867   0.402848  -0.459 0.646375
provnum30044  -1.099493   4.418330  -0.249 0.803514
provnum30055   0.175451   0.176889   0.992 0.321428
provnum30059   0.148876   1.315957   0.113 0.909943
provnum30060  -0.857764   4.090546  -0.210 0.833936
provnum30061   0.379276   0.113578   3.339 0.000861 ***
provnum30062  -0.165666   0.331095  -0.500 0.616901
provnum30064   0.395085   0.181491   2.177 0.029652 *
```

```
provnum30065   0.394153   0.147450   2.673 0.007600 **
provnum30067  -0.456194   1.462445  -0.312 0.755132
provnum30068  -1.271439  10.819353  -0.118 0.906468
provnum30069  -0.028841   0.298681  -0.097 0.923087
provnum30073   1.045035   0.829866   1.259 0.208134
provnum30078   0.904141   1.197396   0.755 0.450320
provnum30080   0.204418   0.208851   0.979 0.327859
provnum30083   0.173088   0.272612   0.635 0.525579
provnum30084   0.536975   1.353745   0.397 0.691678
provnum30085   0.078455   0.214062   0.367 0.714042
provnum30086   0.127506   0.230451   0.553 0.580152
provnum30087   0.376572   0.183396   2.053 0.040222 *
provnum30088   0.244327   0.124009   1.970 0.049005 *
provnum30089   0.174125   0.129690   1.343 0.179609
provnum30092  -0.065579   0.398792  -0.164 0.869404
provnum30093   0.289241   0.158450   1.825 0.068143 .
provnum30094   0.120033   0.491714   0.244 0.807179
provnum32000   1.253908   0.167028   7.507 1.06e-13 ***
provnum32002   1.340420   0.323211   4.147 3.56e-05 ***
provnum32003   1.634917   1.116040   1.465 0.143161
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 2.544499)

    Null deviance: 4014.1  on 1494  degrees of freedom
Residual deviance: 3434.7  on 1429  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

This approach does not resolve our issue. Moreover, these medical providers are not the universe of all medical providers. They are only a sample from all possible medical providers and we would like to be able to infer something about the population of medical providers. To this end, we fit a generalized linear mixed model with a random intercept varying by provider.

First define the model for the mean where we want a log-link, fixed effects of `type`, `white`, `hmo`, and `age.cat` and we want to define a random intercept for `provnum`.

```
model.mu <- DHGLMMODELING(Model = "mean",
                          Link = "log",
                          LinPred = los ~ type + white + hmo +
```

```
                              age.cat + (1 | provnum),
                            RandDist = "gamma")

model.phi <- DHGLMMODELING(Model = "dispersion")
```

And we fit the generalized linear mixed model with a Poisson distribution, a log-link function for the mean, a random effect for medical provider, and a constant dispersion parameter.

```
los.re <- dhglmfit(RespDist = "poisson",
                       DataMain = db,
                       MeanModel = model.mu,
                       DispersionModel = model.phi)
```

```
Distribution of Main Response :
                        "poisson"
[1] "Estimates from the model(mu)"
los ~ type + white + hmo + age.cat + (1 | provnum)
[1] "log"
              Estimate Std. Error t-value
(Intercept)    2.41205    0.07681 31.4018
typeUrgent     0.23690    0.02488  9.5218
typeEmergency  0.13434    0.04818  2.7885
white         -0.01709    0.03173 -0.5385
hmo           -0.09678    0.02600 -3.7222
age.cat1       0.24307    0.12109  2.0074
age.cat2      -0.06605    0.04717 -1.4003
age.cat3      -0.02877    0.03122 -0.9215
age.cat4      -0.02933    0.02590 -1.1324
age.cat5      -0.05183    0.02552 -2.0312
age.cat7      -0.07229    0.02982 -2.4243
age.cat8      -0.13931    0.03961 -3.5167
age.cat9      -0.07316    0.05266 -1.3894
[1] "Estimates for logarithm of lambda=var(u_mu)"
[1] "gamma"
        Estimate Std. Error t-value
provnum   -1.502     0.2001   -7.51
[1] "========== Likelihood Function Values and Condition AIC =========="
                                        [,1]
-2ML (-2 p_v(mu) (h))          :  13045.785
-2RL (-2 p_beta(mu),v(mu) (h)) :  13107.507
cAIC                           :  12929.385
```

```
Scaled Deviance                 :    7087.167
df                              :    1431.471
```

The estimated coefficients for our random intercept model do not differ significantly from the fixed effects estimated in the Poisson GLM. The intercept and the coefficient for the emergency type of admission are the only ones where the difference is a bit larger.

```
tb <- tibble(var = c("Intercept", "Type Urgent", "Type Emergency",
                     "White", "HMO", "Age Group 1", "Age Group 2",
                     "Age Group 3", "Age Group 4", "Age Group 5",
                     "Age Group 7","Age Group 8", "Age Group 9"),
             los.pois.est = coef(sm)[1:13, 1],
             los.pois.se = coef(sm)[1:13, 2],
             los.pois.tstat = los.pois.est / los.pois.se,
             los.pois.sig = ifelse(abs(los.pois.tstat) > 1.96, "*", " "),
             los.pois.wt.est = coef(sm0)[1:13, 1],
             los.pois.wt.se = coef(sm0)[1:13, 2],
             los.pois.wt.tstat = los.pois.wt.est / los.pois.wt.se,
             los.pois.wt.sig = ifelse(abs(los.pois.wt.tstat) > 1.96, "*", " "),
             los.re.est = los.re$beta_coeff[,1],
             los.re.se = los.re$beta_coeff[,2],
             los.re.tstat = los.re.est / los.re.se,
             los.re.sig = ifelse(abs(los.re.tstat) > 1.96, "*", " "))
```

```
tb |>
  select(var,
         los.pois.est, los.pois.tstat, los.pois.sig,
         los.pois.wt.est, los.pois.wt.tstat, los.pois.wt.sig,
         los.re.est, los.re.tstat, los.re.sig) |>
  kbl(booktabs = TRUE,
      col.names = c("Variable", rep(c("Est.", "t-stat", " "), 3)),
      digits = c(0, rep(c(3, 3, 0), 3))) |>
  add_header_above(c(" " = 1, "Poisson" = 3,
                     "Wtd. quasi-Poisson" = 3,
                     "Random Intercepts" = 3)) |>
  kable_classic()
```

Table 6.5 shows the estimated coefficients and their t-statistics for the Poisson, weighted Poisson, and the Poisson model with random intercepts. In all three models the type of admission to the hospital is significant. Note that in all three models urgent admissions have a longer average length of stay compared to elective admissions. But even though emergency admissions also have a positive coefficient, the size is smaller than for urgent admissions which we

Table 6.5: Estimated coefficients and their t-values from three models that include `provnum` as an explanatory variable but whose coefficients are not shown. The models are Poisson, Weighted quasi-Poisson, and Poisson with random intercepts.

	Poisson			Wtd. quasi-Poisson			Random Intercepts		
Variable	Est.	t-stat		Est.	t-stat		Est.	t-stat	
Intercept	2.005	33.277	*	1.966	15.442	*	2.412	31.402	*
Type Urgent	0.231	9.204	*	0.233	4.078	*	0.237	9.522	*
Type Emergency	0.098	1.953		0.071	0.647		0.134	2.788	*
White	-0.011	-0.346		0.054	0.711		-0.017	-0.539	
HMO	-0.096	-3.702	*	-0.096	-1.822		-0.097	-3.722	*
Age Group 1	0.248	2.047	*	0.242	0.856		0.243	2.007	*
Age Group 2	-0.063	-1.324		-0.111	-0.986		-0.066	-1.400	
Age Group 3	-0.027	-0.877		-0.004	-0.064		-0.029	-0.921	
Age Group 4	-0.030	-1.144		-0.055	-0.927		-0.029	-1.132	
Age Group 5	-0.050	-1.973	*	-0.075	-1.313		-0.052	-2.031	*
Age Group 7	-0.071	-2.383	*	-0.113	-1.682		-0.072	-2.424	*
Age Group 8	-0.138	-3.483	*	-0.166	-1.909		-0.139	-3.517	*
Age Group 9	-0.072	-1.363		-0.182	-1.441		-0.073	-1.389	

might feel goes against intuition. Moreover, for the Poisson and weighted Poisson models this coefficient is not statistically significant, but for the random intercepts model it is.

From these coefficients and their standard errors we can see that the type of admission is important, but the size of these coefficients may not be intuitive. Both urgent and emergency admissions have positive coefficients indicating that these patients will, on average, stay longer in the hospital compared with elective admissions. But the coefficient for urgent admission is much bigger than that for emergency admission. We might expect emergency admission patients to be in worse health compared to urgent admissions and thus stay longer in the hospital.

The self-identified indicator of race, `white`, is not significant, but the indicator of membership in a health maintenance organization, `hmo`, is significant with a negative coefficient suggesting that HMO patients' length of stay is about 10% shorter compared to non-HMO patients. Some of the age group coefficients are not statistically significant while others are and the coefficients do not suggest a linear relationship between increasing age and length of stay. Note that the youngest age group has a positive coefficient that is statistically significant. This suggests that young patients tend to stay longer (about 25% longer) at the hospital compared to patients in age group 6.

Exercise

Perhaps the reason that younger patients stay in the hospital longer than patients in age group 6 is because older patients are more likely to die in the hospital.
One of the variables in our data set, `died`, tells us whether or not the patient died at the hospital. We cannot use this variable to predict the length of stay, but we can check if our intuition about patients dying in the hospital tend to be the older patients.
Based on the data we have compute the probability of dying at the hospital split by age group.

Note

The variable `dead` is an indicator variable where a 1 means the patient died at the hospital, otherwise it is zero. To compute the probability of dying we need to group our data by age and compute the mean value of `died` for each group. It's important to also know how many patients are in each group.

```
db |>
  group_by(age.cat) |>
  summarize(n.dead = sum(died),
            n.patients = n(),
            prob.death = mean(died)) |>
  arrange(levels(age.cat))
```

```
# A tibble: 9 x 4
  age.cat n.dead n.patients prob.death
  <fct>    <int>      <int>      <dbl>
1 1            1          6      0.167
2 2           14         60      0.233
3 3           38        163      0.233
4 4           84        291      0.289
5 5          104        317      0.328
6 6          120        328      0.366
7 7           87        191      0.455
8 8           46         93      0.495
9 9           19         46      0.413
```

Note that as age increases, the probability of dying also increases; except for age group 9 where it drops.

Figure 6.11 displays some of the standard diagnostic plots for our random intercept model. The upper left-hand panel shows the fitted values versus the studentized residuals. The fitted values are on the scale of the linear predictor (as opposed to being on the scale of the response).

There is a cluster of 0 claims with a fitted values between 3 and 3.5. These claims come from 0 medical providers and nearly all of them are emergency admissions with a few of them being urgent ones. None of the patients belong to a health maintenance organization. Note that the studentized residuals for these observations have both positive and negative values. Hence, our current model both under-predicts as well as over-predicts these patients.

The upper right-hand side panel shows the absolute value of the studentized residuals against the fitted values. There is a general trend upward trend mainly driven by the observations with a fitted values in excess of 3.

The lower left-hand panel displays a QQ-plot showing that the studentized residuals have a thicker right-hand tail than the normal distribution. And the lower right-hand panel shows that the distribution of the residuals is skewed to the right; in agreement with the QQ-plot.

```
db2 <- db |>
  mutate(mu = los.re[7][[1]],
         SR = los.re[1][[1]])
```

```
p1 <- ggplot(data = db2,
             mapping = aes(x = mu,
                           y = SR)) +
  geom_point(alpha = 0.3) + geom_smooth(se = FALSE) +
  labs(x = "Fitted Values (lin. pred. scale)",
       y = "Studentized Residuals")
p2 <- ggplot(data = db2,
             mapping = aes(x = mu,
                           y = abs(SR))) +
  geom_point(alpha = 0.3) + geom_smooth(se = FALSE) +
  labs(x = "Fitted Values (lin. pred. scale)",
       y = "|Studentized Residuals|")
p3 <- ggplot(data = db2,
             mapping = aes(sample = SR)) +
  geom_qq() + geom_qq_line() +
  labs(x = "Theoretical Quantiles",
       y = "Sample Quantiles")
p4 <- ggplot(data = db2,
             mapping = aes(x = SR)) +
  geom_histogram() +
  labs(x = "Studentized Residuals",
       y = "Frequency")
(p1 + p2) / (p3 + p4)
```

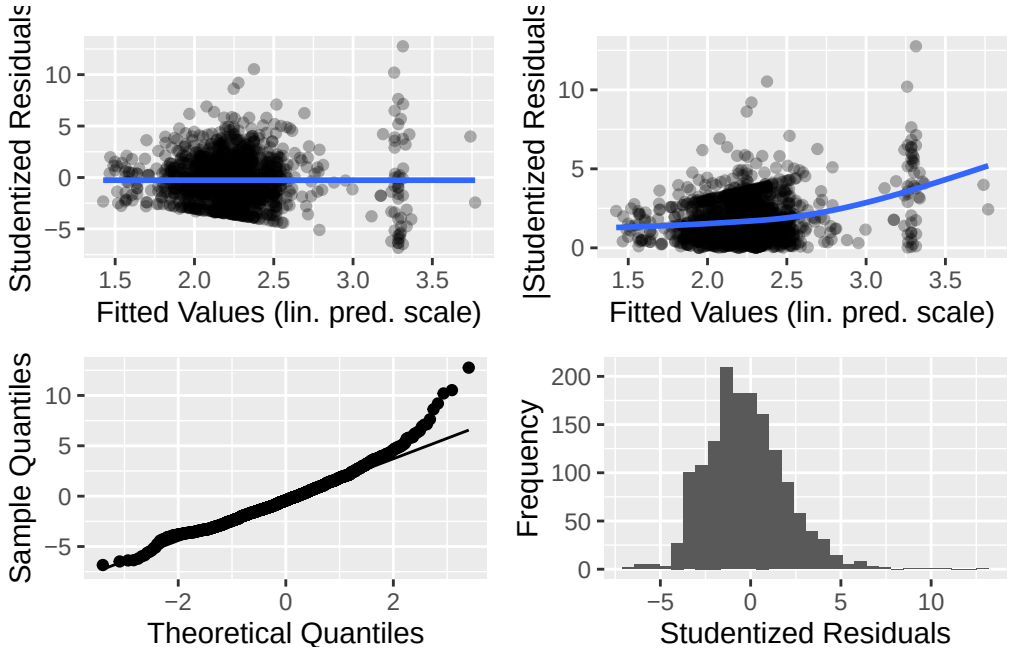


Figure 6.11: Diagnostic plots for the length of stay random intercept model by medical provider.

Figure 6.12 shows the estimated density function for the random effects along with the individual estimates for each medical provider.

```
sigma <- exp(los.re$lambda_coeff[1,1])
alpha <- 1/sigma
dg <- tibble(x = as.vector(exp(los.re$v_h)),
             y = runif(length(x), min = 0, max = 0.02))
dh <- tibble(x = seq(0.01, 3.3, length = 500),
                     y = dgamma(x, shape = alpha, scale = sigma))
rm(sigma, alpha)
```

```
ggplot(data = dh,
       mapping = aes(x = x,
                     y = y)) +
  geom_line() +
  geom_point(data = dg,
             mapping = aes(x = x, y = y),
             alpha = 0.2) +
  labs(x = "Unobserved Gamma Random Variable",
       y = "Density")
rm(dg, dh)
```

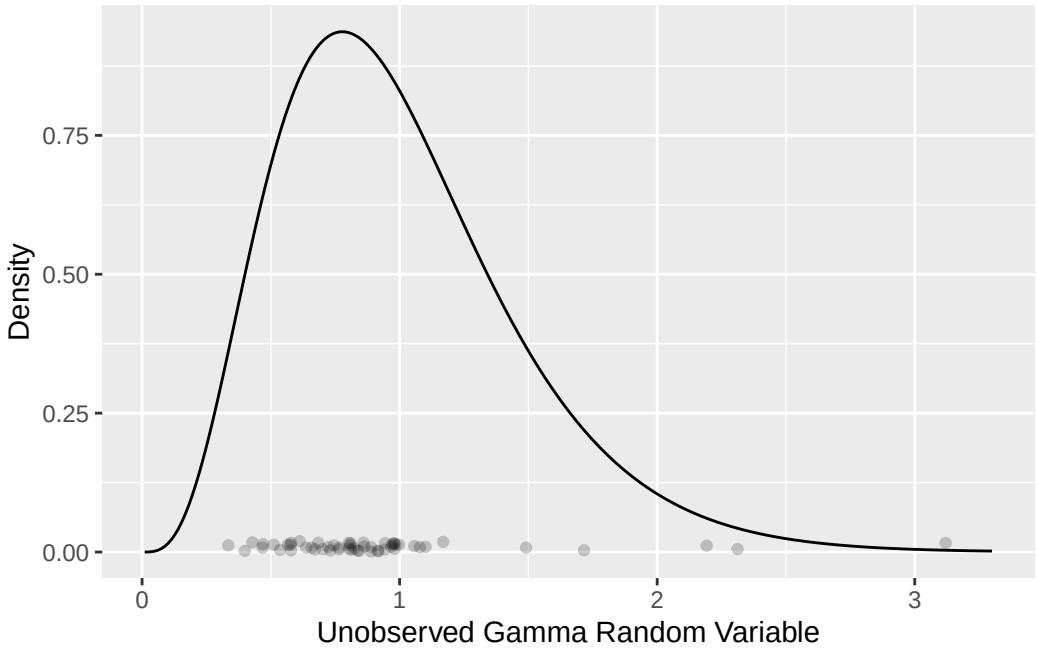


Figure 6.12: The estimated density function for the intercept random effects along with the estimated medical provider random effects.

6.3 Swedish Bus Insurance

In this section we use an example of bus insurance from (Ohlsson and Johansson 2010, sec. 4.5). The data comes from the former Swedish insurance company Wasa for the years 1990 to 1998 and it is about insuring transportation companies and can be accessed from Ohlsson & Johansson's book. Each company owns one or more buses which are insured for shorter or longer periods of time. At that time, the pricing scheme was rather simple based on geographic zone and the age class of the bus.

```
bus <- read_csv("bus-case.csv",
                col_types = "fffnnnn")
```

The variables available and their descriptions, taken from (Ohlsson and Johansson 2010), are given in the following table. We use an English abbreviation for each column and list the original Swedish acronym in parenthesis.

1. `zone` (`ZON`): geographic subdivision of Sweden into seven zones, based on parishes and numbered 1 through 7.

2. `bus.age` (`BUSSALD`): the age class of the bus, in the span 0 to 4.
3. `co.id` (`KUNDNR`): an ID for the company, re-coded here for confidentiality reasons.
4. `no.obs` (`ANTAVT`): number of observations for the company in a given tariff cell based on the zone and age class. There may be more than one observation per bus, since each renewal is counted as a new observation.
5. `dur` (`DUR`): duration measured in days and aggregated over all observations in the tariff cell.
6. `clm.cnt` (`ANTSKAD`): the corresponding number of claims.
7. `tot.cost` (`SKADKOST`): the corresponding total claim cost.

The variable `dur` is the amount of time a policy is in-force and thus we may use it as a measure of exposure. The premium is based on a *bus-year* unit of exposure and the premium for a company would be the sum across all buses in its fleet.

Exploratory Data Analysis

The data set has 1,542 and 7 variables. The variables geographic zone, `zone`, age class of the bus, `bus.age`, and company identification, `co.id`, are categorical and the remaining variables, number of observations, `no.obs`, duration (or exposure), `dur`, claim counts, `clm.cnt`, and total claim cost, `tot.cost`, are numeric.

There are 666 unique company ID's in the dataset. Some companies have a large amount of exposure while others have very little and so using this categorical variable when estimating frequency or severity at the level of a single company would be problematic.

Table 6.6 displays summary statistics for the numeric variables. Note that the number of claims, `clm.cnt`, range from zero to 402, but the 75-th percentile is just one claim. Hence, we suspect that something is not quite right with the number claims for one or more companies. Similarly, the total claim cost has some negative entries and we can see that only 616 records have a non-missing entry. The missing entries correspond to having zero claim counts.

```
f <- function(data, variable) {
  ans <- data |>
    summarize(var = as_label(enquo(variable)),
              n = sum(!is.na({{variable}})),
              mn = mean({{variable}}, na.rm = TRUE),
              sd = sd({{variable}}, na.rm = TRUE),
              p25 = quantile({{variable}}, probs = 0.25, na.rm = TRUE),
              p50 = quantile({{variable}}, probs = 0.50, na.rm = TRUE),
              p75 = quantile({{variable}}, probs = 0.75, na.rm = TRUE),
              min = min({{variable}}, na.rm = TRUE),
              max = max({{variable}}, na.rm = TRUE))
  return(ans)
```

Table 6.6: Summary statistics for the numeric variables in the bus data set. Variable `no.obs` is the number of observations in a particualr tariff cell. Note that each renewal counts as one observation. Duration, `dur`, is measured in days. The variable `clm.cnt` is the number of claims and `tot.cost` is the total loss cost.

				Percentiles				
Variable	Count	Mean	Std. Dev.	25	50	75	Min	Max
no.obs	1,542	12.87	32.70	2	4	9.0	1	392
dur	1,542	2,283.81	4,969.80	365	725	1,876.5	1	66,327
clm.cnt	1,542	1.95	14.52	0	0	1.0	0	402
tot.cost	616	52,870.68	143,943.99	0	3,525	39,897.2	-17,318	1,330,610

```
}

tb <- bind_rows(f(bus, no.obs),
                f(bus, dur),
                f(bus, clm.cnt),
                f(bus, tot.cost))
rm(f)
```

```
kbl(tb,
    booktabs = TRUE,
    digits = c(0,0,2,2,1,1,1,1,1),
    col.names = c("Variable", "Count", "Mean", "Std. Dev.",
                  "25", "50", "75", "Min", "Max"),
    align = "lrrrrrrrr",
    format.args = list(big.mark = ","), nsmall = 1) |>
  add_header_above(c(" " = 1, " " = 1, " " = 1, " " = 1, "Percentiles" = 3,
                     " " = 1, " " = 1)) |>
  kable_classic()
rm(tb)
```

Regarding the negative entries for total loss cost, these arise from 60 rows of data. We do not know why these 60 entries in the data have negative loss costs and we will remove them from our analysis.

```
idx <- pull(bus, tot.cost) >= 0
idx[is.na(idx)] <- TRUE
db <- bus[idx,]
rm(idx)
```

The top ten claim counts are:

```
sort(pull(db, clm.cnt), decreasing = TRUE)[1:10]
```

```
 [1] 402 377  55  53  38  34  34  29  28  27
```

The two largest ones, (402 and 377), come from company 145. We suspect that these entries are erroneous and because we cannot go back to the source systems or other sources of information to correct them we will delete company 145 from our analysis. This will remove a total of 9 rows of data.

```
db <- db |>
  filter(co.id != "145")
```

In Figure 6.13 we have the histograms of our four numeric variables. Note that all four of them are highly skewed to the right and in order to enhance each of the displays we have omitted some large observations.

```
p1 <- ggplot(data = db,
       mapping = aes(x = no.obs)) +
  geom_histogram(bins = 50) +
  labs(x = "Number of Bus Observations",
       y = "Count") +
  coord_cartesian(xlim = c(0,100))

p2 <- ggplot(data = db,
       mapping = aes(x = dur)) +
  geom_histogram(bins = 50) +
  labs(x = "Duration (in days)",
       y = "Count") +
  scale_x_continuous(breaks = c(0, 5000, 10000, 15000, 20000),
                     labels = c("0", "5K", "10K", "15K", "20K")) +
  coord_cartesian(xlim = c(0,20000))

p3 <- ggplot(data = db,
       mapping = aes(x = clm.cnt)) +
  geom_histogram(bins = 50) +
  labs(x = "Number of Claims",
       y = "Count") +
  coord_cartesian(xlim = c(0,25))
```

```
p4 <- ggplot(data = db,
       mapping = aes(x = tot.cost)) +
  geom_histogram(bins = 50) +
  labs(x = "Total Loss Cost",
       y = "Count") +
  scale_x_continuous(breaks = c(0, 100000, 200000, 300000),
                     labels = c("0", "100K", "200K", "300K")) +
  coord_cartesian(xlim = c(0, 300000))

(p1 + p2) / (p3 + p4)
rm(p1, p2, p3, p4)
```

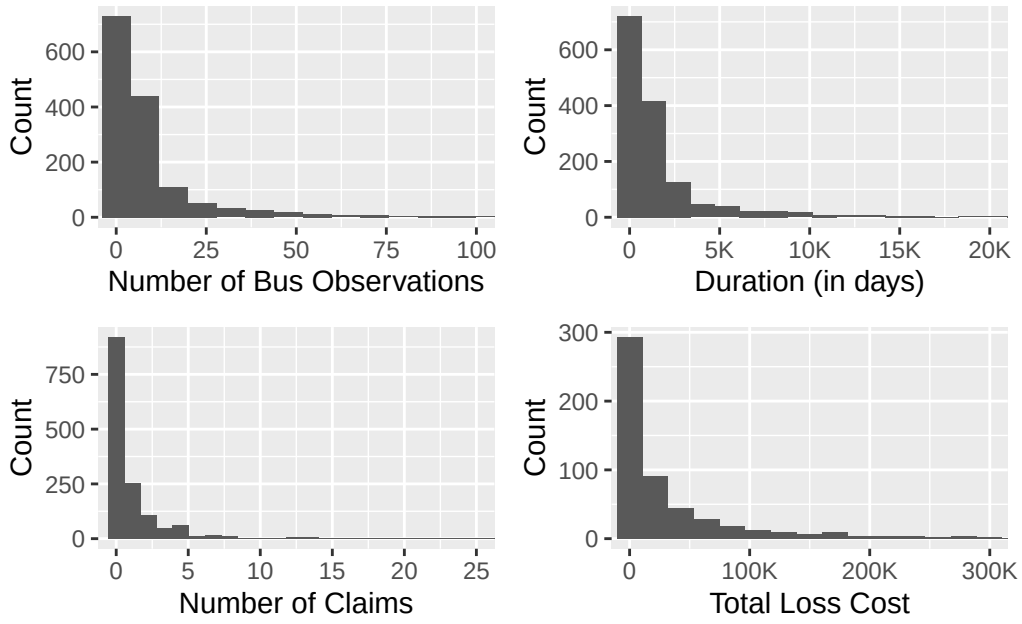


Figure 6.13: Histograms of the numeric variables in the bus dataset. Note that all four of them are heavily skewed to the right and we have not shown all data points in order to enhance the display. For the number of bus observations, 30 observations greater than 100 are not shown. For duration, 26 observations greater than 20,000 are not displayed. Seven claim counts greater than 25 are not shown and 23 observations for total loss cost greater than 300,000 are also not shown.

In the 1990's the rating plan might have been relatively simple; perhaps, it would have only used two rating factors: zone and age class. Using these two variables, Table 6.7 shows the empirical frequency (per year of exposure) for each of the 35 (7×5) rating cells in the plan. There is considerable variation in the empirical frequencies and thus we can use `zone` and `bus.age` as part of a rating plan.

Exercise

Summarize the number of claims in the bus data by `zone` and `bus.age`. How many cells have a small number of claims, say less than 5?
You can repeat the exercise with exposure.

Solution

We summarize the data as follows:

```
db |>
  group_by(zone, bus.age) |>
  summarize(clm.cnt = sum(clm.cnt),
            .groups = "drop") |>
  pivot_wider(names_from = bus.age,
              values_from = clm.cnt)
```

```
# A tibble: 7 x 6
  zone    `0`   `1`   `2`   `3`   `4`
  <fct> <dbl> <dbl> <dbl> <dbl> <dbl>
1 1        20     8    14    24    38
2 2        14    26    13    25   151
3 3        11    19    19    11   132
4 4        83   105   150   118   709
5 5         3     8     0     1    51
6 6        29    14    14    16   184
7 7         1     2     1     3     7
```

Note that most of the cells in zone 7 and zone 5 have very few counts. All other cells have many more claim counts. In particular, buses with age category 4 have the most claims and zone 4 has also the most claims.
Looking at exposure, we have:

```
db |>
  group_by(zone, bus.age) |>
  summarize(expo = sum(dur),
            .groups = "drop") |>
  pivot_wider(names_from = bus.age,
              values_from = expo)
```

```
# A tibble: 7 x 6
  zone    `0`    `1`    `2`    `3`     `4`
```

```
  <fct> <dbl>  <dbl>  <dbl>  <dbl>   <dbl>
1 1     10061  10823  15073  30924   79106
2 2     22407  19023  18736  24372  168620
3 3     25624  24768  22716  29321  278519
4 4     88665 112912 143189 151654 1434918
5 5      3299   4541   3800   2958   87265
6 6     20791  21966  20740  23695  296228
7 7      2739   2496   3135   3377   31632
```

Again, zone 4 and age category 4 have the most exposure. Zones 5 and 7 have much smaller exposures.

Since both `zone` and `bus.age` have few levels we can reliably estimate the frequency for most cells; but, within each cell we may still have many companies whose experience is not similar to each other. Hence, we would like to include company identifier, `co.id`, into the rating plan. Unfortunately, company identifier has over 600 unique entries. Hence, adding this as a regular classification variable would give us a rating plan with $7 \times 5 \times 660 = 23{,}100$ individual cells. Since our data only has around 1,500 observations most of the cells would be empty. We cannot go in this direction.

What we can do is bring to bear the tools of credibility and mixed effects models to help us incorporate the information we have regarding the experience of each company.

Modeling Frequency

Let us start with modeling frequency of claims by first looking at the empirical data we have. The overall *yearly frequency*, without regard to any classification variables, is equal to 0.228. If we cross-classify our data by geographical zone and bus age category, then the *yearly frequency* for each combination is displayed in Table 6.7.

```
tb <- db |>
  group_by(zone, bus.age) |>
  summarize(sz = n(),
            dur.yrs = sum(dur),
            clm.cnt = sum(clm.cnt),
            fq = clm.cnt / dur.yrs * 365,
            .groups = "drop")
tbl <- tb |>
  select(zone, bus.age, fq) |>
  pivot_wider(names_from = "bus.age",
              values_from = fq)
```

Table 6.7: Empirical frequency (per year of exposure) for the bus dataset by geographic zone and age class of the bus.

	Bus Age Class				
zone	0	1	2	3	4
1	0.726	0.270	0.339	0.283	0.175
2	0.228	0.499	0.253	0.374	0.327
3	0.157	0.280	0.305	0.137	0.173
4	0.342	0.339	0.382	0.284	0.180
5	0.332	0.643	0.000	0.123	0.213
6	0.509	0.233	0.246	0.246	0.227
7	0.133	0.292	0.116	0.324	0.081

```
kbl(tbl,
    booktabs = TRUE,
    digits = 3,
    align = "crrrrr") |>
  add_header_above(c(" " = 1, "Bus Age Class" =  5)) |>
  kable_classic()
```

Note that zone 1 and bus age category 0 has a very high empirical yearly frequency of 0.726. The data for this cell, along with the individual companies annual frequency, given in Table 6.8 and we can see that the experience is quite heterogeneous. We have companies with a small amount of exposure (356 and 460 days) along with companies with more exposure (2,191 and 1,949 days).

```
db |>
  select(zone, bus.age, co.id, dur, clm.cnt) |>
  filter(zone == "1",
         bus.age == "0") |>
  mutate(freq = clm.cnt / dur * 365) |>
  arrange(desc(freq), dur) |>
  kbl(booktabs = TRUE,
      col.names = c("Zone", "Bus Age", "Company", "Duration",
                    "Claim Count", "Frequency"),
      digits = c(0,0,0,0,0,3),
      align = "cccrrr") |>
  kable_classic()
```

We can fit a generalized linear model to geographic zone and bus age, ignoring company for

Table 6.8: Observations available for zone 1 and bus age category 0 along with the empirical annual frequency for each company. The overall annual frequency for this cell is equal to 0.726.

Zone	Bus Age	Company	Duration	Claim Count	Frequency
1	0	518	1079	8	2.706
1	0	184	1213	4	1.204
1	0	226	1949	5	0.936
1	0	597	457	1	0.799
1	0	231	1777	1	0.205
1	0	385	2191	1	0.167
1	0	471	356	0	0.000
1	0	539	460	0	0.000
1	0	15	579	0	0.000

the moment, to get an initial sense of how we might model frequency.

Exercise

Fit a Poisson model to frequency of claims using `zone` and `bus.age` as classification variables and a log-link function. Does the model fit the data well?

Solution

The Poisson model can be fitted via

```
fq.poi <- glm(clm.cnt ~ zone + bus.age,
              data = db,
              family = poisson(link = "log"),
              offset = log(dur))
summary(fq.poi)
```

```
Call:
glm(formula = clm.cnt ~ zone + bus.age, family = poisson(link = "log"),
    data = db, offset = log(dur))

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.943244   0.123755 -56.105  < 2e-16 ***
zone2        0.279470   0.118671   2.355   0.0185 *
```

```
zone3       -0.271849   0.122356  -2.222   0.0263 *
zone4       -0.084820   0.103045  -0.823   0.4104
zone5       -0.002343   0.160733  -0.015   0.9884
zone6        0.034376   0.117055   0.294   0.7690
zone7       -0.718687   0.284924  -2.522   0.0117 *
bus.age1     0.008051   0.108257   0.074   0.9407
bus.age2     0.014442   0.104842   0.138   0.8904
bus.age3    -0.213333   0.106407  -2.005   0.0450 *
bus.age4    -0.531115   0.083951  -6.327 2.51e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 2142.9  on 1472  degrees of freedom
Residual deviance: 1985.3  on 1462  degrees of freedom
AIC: 3502.9

Number of Fisher Scoring iterations: 5
```

The mean deviance estimate of the dispersion parameter ϕ is equal to

```
round(deviance(fq.poi) / df.residual(fq.poi), 3)
```

```
[1] 1.358
```

which is clearly larger than the theoretical value of $\phi = 1$, indicating that the data is overdispersed and the Poisson model does not fit well.
The Pearson estimate of dispersion parameter yields a similar conclusion.

```
round(sum(resid(fq.poi, type = "pearson")^2) /
        df.residual(fq.poi), 3)
```

```
[1] 1.465
```

We fit a Negative Binomial model to the data using `zone` and `bus.age` as main effects. Before fitting the model we select zone 4 and bus age 4 as the base levels for these factors variables because at these levels they have the most exposure.

```
db$zone.f <- fct_relevel(db$zone, "4")
db$bus.age.f <- fct_relevel(db$bus.age, "4")
```

```
fq.nb <- glm.nb(clm.cnt ~ zone.f + bus.age.f + offset(log(dur)),
                data = db)
summary(fq.nb)
```

```
Call:
glm.nb(formula = clm.cnt ~ zone.f + bus.age.f + offset(log(dur)),
    data = db, init.theta = 2.027840354, link = log)

Coefficients:
            Estimate Std. Error  z value Pr(>|z|)
(Intercept) -7.66330    0.06203 -123.536  < 2e-16 ***
zone.f1      0.17132    0.16878    1.015 0.310083
zone.f2      0.30441    0.12369    2.461 0.013848 *
zone.f3     -0.07103    0.11665   -0.609 0.542551
zone.f5      0.20278    0.18607    1.090 0.275806
zone.f6      0.16002    0.10442    1.533 0.125394
zone.f7     -0.49637    0.32616   -1.522 0.128046
bus.age.f0   0.59037    0.11936    4.946 7.57e-07 ***
bus.age.f1   0.46727    0.11740    3.980 6.88e-05 ***
bus.age.f2   0.40988    0.11836    3.463 0.000534 ***
bus.age.f3   0.37029    0.11220    3.300 0.000965 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(2.0278) family taken to be 1)

    Null deviance: 1252.1  on 1472  degrees of freedom
Residual deviance: 1196.4  on 1462  degrees of freedom
AIC: 3188.6

Number of Fisher Scoring iterations: 1

              Theta:  2.028
          Std. Err.:  0.252

 2 x log-likelihood:  -3164.615
```

Note that the coefficients for `bus.age` are positive and they steadily decline as the age category increases. All of the `zone` coefficients have large standard errors compared to their estimated values except for zone 2. Hence, it appears that all zones, except zone 2, have similar claim frequency as zone 4.

Table 6.9: Annual mean frequency predictions from the negative binomial model with main effects for geographic zone and bus age class.

	Bus Age Class				
Zone	0	1	2	3	4
1	0.367	0.325	0.307	0.295	0.204
2	0.420	0.371	0.350	0.337	0.232
3	0.288	0.255	0.241	0.231	0.160
4	0.309	0.274	0.258	0.248	0.171
5	0.379	0.335	0.316	0.304	0.210
6	0.363	0.321	0.303	0.291	0.201
7	0.188	0.167	0.157	0.151	0.104

Annual frequency predictions from the negative binomial model are displayed in Table 6.9 and they do not take into account that the claims experience comes from different companies.

```
df <- expand_grid(zone.f = factor(1:7),
                  bus.age.f = factor(0:4))
df$dur <- 365
```

```
df <- df |>
  mutate(nb.mu = predict(fq.nb, newdata = df, type = "response"))
df |>
  select(zone.f, bus.age.f, nb.mu) |>
  pivot_wider(names_from = bus.age.f,
              values_from = nb.mu) |>
  kbl(booktabs = TRUE,
      col.names = c("Zone", "0", "1", "2", "3", "4"),
      digits = c(0, 3, 3, 3, 3, 3),
      align = "cccrrr") |>
  add_header_above(c(" " = 1, "Bus Age Class" = 5)) |>
  kable_classic()
```

Next, we incorporate the information available in the variable company ID, `co.id`, by adding it as a random effect for the intercept. We first define the model structure we want for the mean and we will keep the dispersion parameter fixed; therefore, we are defining a generalized linear mixed effects model where the response distribution is Poisson and the random effect is gamma distributed.

```
model.mu <- DHGLMMODELING(Model = "mean",
                          Link = "log",
                          LinPred = clm.cnt ~ zone.f + bus.age.f +
                            (1 | co.id),
                          RandDist = "gamma",
                          Offset = log(db$dur))

model.phi <- DHGLMMODELING(Model = "dispersion")
```

We fit our model via the `dhglmfit()` function as follows:

```
fq.poi.re <- dhglmfit(RespDist = "poisson",
                      DataMain = db,
                      MeanModel = model.mu,
                      DispersionModel = model.phi)
```

```
Distribution of Main Response :
                       "poisson"
[1] "Estimates from the model(mu)"
clm.cnt ~ zone.f + bus.age.f + (1 | co.id)
[1] "log"
            Estimate Std. Error   t-value
(Intercept) -7.65055    0.05797 -131.9806
zone.f1      0.22644    0.14761    1.5340
zone.f2      0.44224    0.13197    3.3512
zone.f3      0.03305    0.12813    0.2579
zone.f5      0.28805    0.18542    1.5535
zone.f6      0.20941    0.11306    1.8522
zone.f7     -0.57862    0.36349   -1.5919
bus.age.f0   0.51538    0.08706    5.9201
bus.age.f1   0.50149    0.08194    6.1204
bus.age.f2   0.46698    0.07892    5.9170
bus.age.f3   0.36315    0.08001    4.5390
[1] "Estimates for logarithm of lambda=var(u_mu)"
[1] "gamma"
      Estimate Std. Error t-value
co.id   -1.097     0.1033  -10.62
[1] "========== Likelihood Function Values and Condition AIC =========="
                                       [,1]
-2ML (-2 p_v(mu) (h))          :  3171.369
-2RL (-2 p_beta(mu),v(mu) (h)) :  3199.679
```

```
cAIC                          :  3107.045
Scaled Deviance               :  1194.541
df                            :  1264.553
```

From the above output we can see that the estimated coefficients are not too different from those we obtained for the Negative Binomial model. We also have the estimated variance for our unobserved gamma random effect. Its value is equal to 0.334. The density function for the random effect, along with company estimated random effects, is shown in Figure 6.14.

```
sigma <- exp(fq.poi.re$lambda_coeff[1,1])
alpha <- 1/sigma
dg <- tibble(x = exp(fq.poi.re$v_h),
             y = runif(length(x), min = 0, max = 0.02))
dh <- tibble(x = seq(0.01, 3.2, length = 500),
                     y = dgamma(x, shape = alpha, scale = sigma))
rm(sigma, alpha)
```

```
ggplot(data = dh,
       mapping = aes(x = x,
                     y = y)) +
  geom_line() +
  geom_point(data = dg,
             mapping = aes(x = x, y = y),
             alpha = 0.2) +
  labs(x = "Unobserved Gamma Random Variable",
       y = "Density")
rm(dg, dh)
```

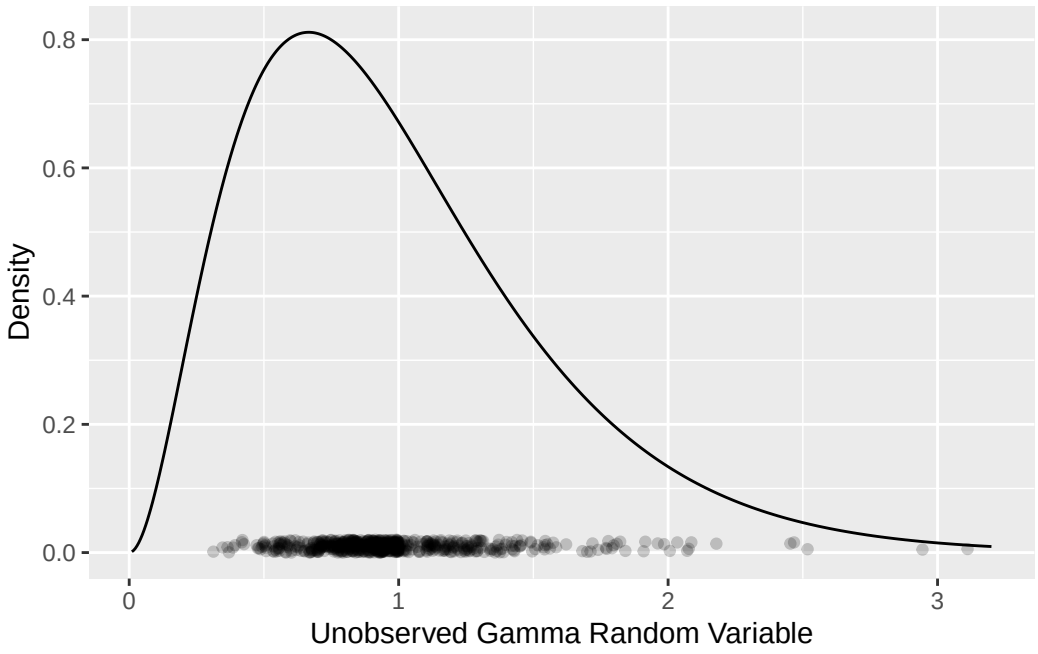


Figure 6.14: The estimated density function for the intercept random effects along with the individual estimated company effects.

Note that in Figure 6.14 we have five points with an estimated random effect greater than 2.25. In Table 6.10 we have extracted these companies and provided their total claim count, exposure, empirical annual frequency, and the estimated random effect. Note that for these companies the empirical frequency is quite large, given that the portfolio annual frequency is 0.228, and so we would expect them to have large estimated random effects.

```
db |>
  group_by(co.id) |>
  summarize(clm.cnt = sum(clm.cnt),
            dur = sum(dur)) |>
  mutate(ran.ef = exp(fq.poi.re[["v_h"]][,1])) |>
  select(co.id, clm.cnt, dur, ran.ef) |>
  mutate(freq = clm.cnt / dur * 365) |>
  select(co.id, clm.cnt, dur, freq, ran.ef) |>
  arrange(desc(ran.ef)) |>
  slice(1:5) |>
  kbl(booktabs = TRUE,
      col.names = c("Company", "of Claims", "(in days)",
                    "Frequency", "Effect"),
      digits = c(0, 0, 0, 3, 3),
      align = "crrrr",
```

Table 6.10: Empirical annual frequency and estimated random effect for the top five companies ranked on the size of their random effect.

Company	Number of Claims	Exposure (in days)	Annual Frequency	Random Effect
561	37	13,615	0.992	3.112
559	106	59,833	0.647	2.944
301	26	17,216	0.551	2.518
535	41	29,378	0.509	2.467
406	12	4,709	0.930	2.454

```
    format.args = list(big.mark = ",")) |>
  add_header_above(c(" " = 1, "Number" = 1, "Exposure" = 1,
                     "Annual" = 1, "Random" = 1),
                   align = rep("r", 5),
                   line = FALSE) |>
  kable_classic()
```

Based on this generalized linear mixed effects model the *annual frequency* would be calculated via the formula

$$\mu_F = \exp(\text{intercept} + \text{zone} + \text{bus.age} + \log(365)) \times (\text{random effect of company}).$$

The first term in the above expression represents all the combinations of the fixed effects. The last term is the random effect for the company. Table 6.11 shows the annual frequency for each combination of geographic zone and bus age category based on the fixed effects and Table 6.12 shows 30 companies along with their estimated random effects. The companies have been sorted by the magnitude of the random effects and the table displays the 10 smallest and largest random effects along with 10 entries from the middle. Figure 6.15 displays a scatterplot of all companies by their total exposure and estimated random effects along with their total claim count. Companies with large random effects tend to have worse claims experience; that is, lower exposure and large number of claims. Companies with random effects below 1 tend to have better claims experience.

```
cf <- fq.poi.re$beta_coeff[,1]
zn <- c(cf[2:4], zone.f4 = 0, cf[5:7])
ba <- c(cf[8:11], bus.age.f4 = 0)
m <- exp(cf[1]) * outer(exp(zn), exp(ba)) * 365
dimnames(m) <- list(str_c("Zone ", 1:7), str_c("Bus Age ", 0:4))
rm(cf, zn, ba)
```

Table 6.11: Estimated annual frequency based only on the fixed effects from the generalized linear mixed model.

	Bus Age Category				
Zone	0	1	2	3	4
1	0.365	0.360	0.347	0.313	0.218
2	0.452	0.446	0.431	0.389	0.270
3	0.301	0.296	0.286	0.258	0.179
4	0.291	0.287	0.277	0.250	0.174
5	0.388	0.382	0.369	0.333	0.232
6	0.358	0.354	0.342	0.308	0.214
7	0.163	0.161	0.155	0.140	0.097

```
fe.tbl <- bind_cols("Zone" = 1:7, as_tibble(m))
fe.tbl |>
  kbl(booktabs = TRUE,
      col.names = c("Zone", 0:4),
      digits = c(0, 3, 3, 3, 3, 3),
      align = "crrrrr") |>
  add_header_above(c(" " = 1, "Bus Age Category" = 5)) |>
  kable_classic()
rm(fe.tbl, m)
```

```
rn.ef <- exp(fq.poi.re$v_h[,1])
ord <- order(rn.ef)
rn.ef <- rn.ef[ord]
nms <- as.numeric(dimnames(fq.poi.re$v_h)[[1]])[ord]
tbl <- bind_cols(co1 = nms[1:10],
                 re1 = rn.ef[1:10],
                 co2 = nms[325:334],
                 re2 = rn.ef[325:334],
                 co3 = nms[651:660],
                 re3 = rn.ef[651:660])
rm(rn.ef, ord, nms)
```

```
tbl |>
  kbl(booktabs = TRUE,
      col.names = rep(c("Company", "Effect"), 3),
      digits = rep(c(0, 3), 3),
```

Table 6.12: Estimated company random effects. Companies have been sorted from smallest to largest random effects.

Smallest		Medium		Largest	
Company	Random Effect	Company	Random Effect	Company	Random Effect
524	0.312	478	0.926	271	2.034
136	0.346	587	0.929	169	2.071
289	0.365	154	0.929	297	2.077
368	0.370	314	0.930	548	2.086
549	0.385	91	0.931	226	2.179
285	0.393	665	0.932	406	2.454
279	0.418	526	0.932	535	2.467
183	0.420	5	0.932	301	2.518
522	0.426	437	0.932	559	2.944
373	0.473	184	0.932	561	3.112

```
      align = "crcrcr") |>
  add_header_above(rep(c(" " = 1, "Random" = 1), 3),
                   line = FALSE) |>
  add_header_above(c("Smallest" = 2, "Medium" = 2, "Largest" = 2)) |>
  kable_classic()
rm(tbl)
```

```
dc <- db |>
  group_by(co.id) |>
  summarize(clm.cnt = sum(clm.cnt),
            dur = sum(dur)) |>
  mutate(ran.ef = exp(fq.poi.re[["v_h"]][,1]),
         freq = clm.cnt / dur * 365)
```

```
ggplot(data = dc,
       mapping = aes(y = dur,
                     x = ran.ef,
                     size = clm.cnt)) +
  geom_point(alpha = 0.2) +
  labs(x = "Estimated Random Effect",
       y = "Total Company Exposure (in days)") +
  scale_size_continuous(guide = guide_legend(title = "Total Claim Count")) +
```

```
  theme(legend.position = "bottom")
rm(dc)
```

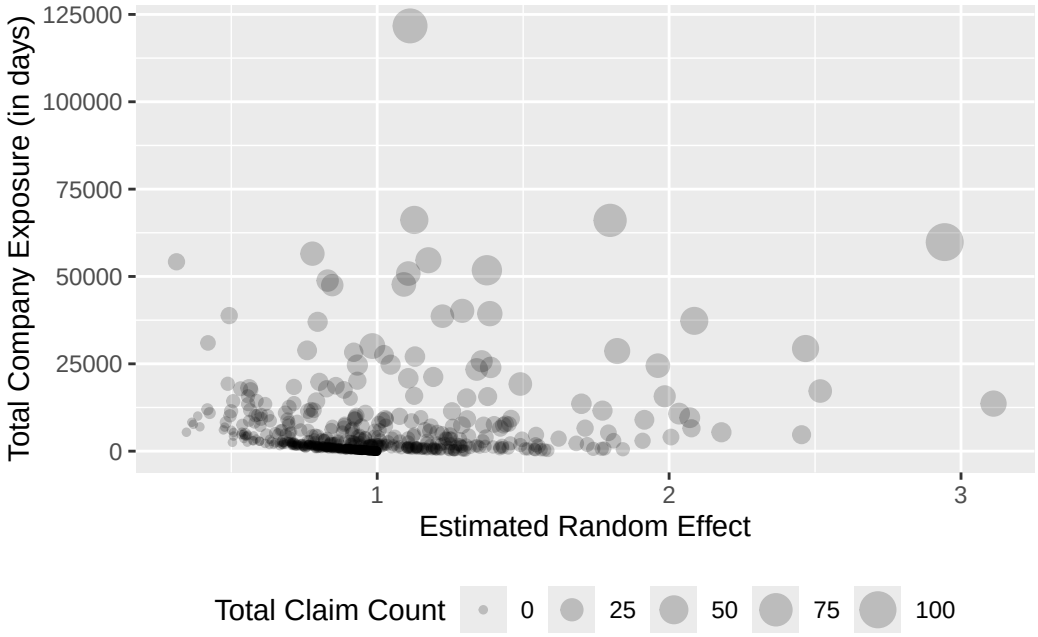


Figure 6.15: Total company exposure and claim counts along with the estimated company random effect.

References

Agresti, Alan. 2002. *Categorical Data Analysis*. 2nd ed. Wiley Series in Probability and Statistics. New York: Wiley-Interscience.

Antoniadis, Anestis, Irène Gijbels, Sophie Lambert-Lacroix, and Jean-Michel Poggi. 2016. "Joint Estimation and Variable Selection for Mean and Dispersion in Proper Dispersion Models." *Electronic Journal of Statistics* 10 (1). https://doi.org/10.1214/16-EJS1152.

Bissell, A. F. 1972. "A Negative Binomial Model with Varying Element Sizes." *Biometrika* 59 (2): 435–41. https://doi.org/10.2307/2334588.

Bühlmann, Hans. 1967. "Experience Rating and Credibility." *ASTIN Bulletin* 4 (3): 199–207. https://doi.org/10.1017/S0515036100008989.

Bühlmann, Hans, and Alois Gisler. 2005. *A Course in Credibility Theory and Its Applications*. Universitext. Berlin ; New York: Springer.

Bühlmann, H, and A Gisler. 1997. "Credibility in the Regression Case Revisited (A Late Tribute to Charles A. Hachemeister)." *ASTIN Bulletin: The Journal of the IAA* 27 (1): 83–98.

Charpentier, Arthur, ed. 2015. *Computational Actuarial Science with R*. Chapman & Hall/CRC the R Series. Boca Raton: CRC Press.

Cleveland, William S. 1979. "Robust Locally Weighted Regression and Smoothing Scatterplots." *Journal of the American Statistical Association* 74 (368): 829–36. https://doi.org/10.1080/01621459.1979.10481038.

Danneburg, DR. 1996. "Basic Actuarial Credibility Models: Evaluation and Extension." Doct. thesis. Amsterdam.

De Vylder, F. 1981. "Regression Model with Scalar Credibility Weights." *Mitt. Ver. Schweiz. Vers. Math.*

De Vylder, F. 1985. "Non-Linear Regression in Credibility Theory." *Insurance: Mathematics and Economics* 4 (3): 163–72. https://doi.org/10.1016/0167-6687(85)90012-5.

Dunn, Peter K, and Gordon K Smyth. 1996. "Randomized Quantile Residuals." *J. Comput. Graph. Stat.* 5 (3): 236–44.

———. 2018. *Generalized Linear Models With Examples in R*. Springer.

Dutang, Christophe, and Arthur Charpentier. 2020. "CASdatasets: Insurance Datasets."

Efron, Bradley, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. 2004. "Least Angle Regression." *The Annals of Statistics* 32 (2): 407–51. https://www.jstor.org/stable/3448465.

Frees, Edward W. 2003. "Multivariate Credibility for Aggregate Loss Models." *North American Actuarial Journal* 7 (1): 13–37. https://doi.org/10.1080/10920277.2003.10596074.

Frees, Edward W., and Ping Wang. 2005. "Credibility Using Copulas." *North American Actuarial Journal* 9 (2): 31–48. https://doi.org/10.1080/10920277.2005.10596196.

Frees, Edward W, Virginia R Young, and Yu Luo. 1999. "A Longitudinal Data Analysis Interpretation of Credibility Models." *Insur. Math. Econ.* 24 (3): 229–47.

Gelman, Andrew, and Jennifer Hill. 2007. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Analytical Methods for Social Research. Cambridge ; New York: Cambridge University Press.

Hachemeister, Charles A. 1975. "Credibility for Regression Models with Application to Trend." In *Credibility: Theory and Applications*, edited by P M Kahn, 129–63. New York: Academic Press.

Herzog, Thomas N. 2010. *Introduction to Credibility Theory*. 4th ed. ACTEX Academic Series. Winsted, CT: ACTEX Publications.

Hilbe, Joseph M. 2007. *Negative Binomial Regression*. Cambridge ; New York: Cambridge University Press.

Jewell, William S. 1975. "The Use of Collateral Data in Credibility Theory: A Hierarchical Model." Research Memorandum RM-75-024. Laxenburg, Austria: International Institute for Applied Systems Analysis. https://pure.iiasa.ac.at/492.

Kaas, Rob, ed. 2009. *Modern Actuarial Risk Theory: Using R*. 2nd ed. Berlin ; [New York]: Springer.

Klugman, Stuart A., Harry H. Panjer, and Gordon E. Willmot. 1998. *Loss Models: From Data to Decisions*. Wiley Series in Probability and Statistics. New York: Wiley.

LEE, MAENGSEOK, LARS NOH. 2020. *DATA ANALYSIS USING HIERARCHICAL GENERALIZED LINEAR MODELS WITH R*. S.l.: CRC PRESS.

Lee, Y., and J. A. Nelder. 1996. "Hierarchical Generalized Linear Models." *Journal of the Royal Statistical Society. Series B (Methodological)* 58 (4): 619–78. https://www.jstor.org/stable/2346105.

Lee, Youngjo, John A. Nelder, and Yudi Pawitan. 2021. *Generalized Linear Models with Random Effects: Unified Analysis via H-likelihood*. Second edition, first issued in paperback. Monographs on Statistics and Applied Probability 153. Boca Raton London New York: CRC Press, Taylor & Fancis Group.

Mowbray, Albert H. 1914. "How Extensive a Payroll Exposure Is Necessary to Give a Dependable Pure Premium?" *Proceedings of the Casualty Actuarial Society* 1 (1): 36–42. https://www.casact.org/sites/default/files/database/proceed_proceed14_1914.pdf.

Ohlsson, Esbjørn, and Bjørn Johansson. 2010. *Non-Life Insurance Pricing with Generalized Linear Models*. EAA Lecture Notes. Heidelberg ; New York: Springer.

Rempala, Grzegorz A., and Richard A. Derrig. 2005. "Modeling Hidden Exposures in Claim Severity Via the Em Algorithm." *North American Actuarial Journal* 9 (2): 108–28. https://doi.org/10.1080/10920277.2005.10596206.

Straub, Erwin. 1997. *Non-Life Insurance Mathematics*. 2nd, corr. print. Berlin ; New York : Zürich: Springer ; Swiss Association of Actuaries.

Tager, I B, S T Weiss, A Muñoz, B Rosner, and F E Speizer. 1983. "Longitudinal Study of the Effects of Maternal Smoking on Pulmonary Function in Children." *N. Engl. J. Med.* 309 (12): 699–703.

Tager, Ira B., Scott T. Weiss, Bernard Rosner, and Frank E. Speizer. 1979. "EFFECT OF PARENTAL CIGARETTE SMOKING ON THE PULMONARY FUNCTION OF CHILDREN." *American Journal of Epidemiology* 110 (1): 15–26. https://doi.org/10.1093/oxfordjournals.aje.a112783.

Venter, Gary G. 1996. "Credibility." In *Foundations of Casualty Actuarial Science*, 3rd ed, Chapter 7:375–483. Arlington, Va: Casualty Actuarial Society.

A Bühlmann-Straub Simulation

```
library(tidyverse)
library(patchwork)
library(lme4)
```

The following function, `sim.BS()`, simulates a Bühlmann-Straub data set with given parameter values.

```
sim.BS <- function(
    sim.label = "A",
    J = 100,
    N = 5,
    beta = 80,
    sigma.b.sq = 64,
    sigma.sq = 100,
    weight.min = 0.5,
    weight.spread = 1) {

  risk.dev <- rep(rnorm(J, mean = 0, sd = sqrt(sigma.b.sq)),
                  each = N)
  time.dev <- rnorm(J * N, mean = 0, sd = sqrt(sigma.sq))
  weight <- weight.min + runif(J * N, min = 0, max = weight.spread)
  tb <- tibble(
    sim.label = rep(sim.label, J * N),
    risk = factor(rep(1:J, each = N)),
    Wt = weight,
    rsk.dev = risk.dev,
    tme.dev = time.dev,
    Y = beta + rsk.dev + tme.dev
  )
  return(tb)
}
```

With this function we can generate four different data sets with different number of observation per risks, $N = 5, 10, 20, 40$ and store them in a list.

```
set.seed(398845)
BS.data <- list(
  BS.5  = sim.BS(sim.label =  "5 Obs. per Risk", N =  5),
  BS.10 = sim.BS(sim.label = "10 Obs. per Risk", N = 10),
  BS.20 = sim.BS(sim.label = "20 Obs. per Risk", N = 20),
  BS.40 = sim.BS(sim.label = "40 Obs. per Risk", N = 40)
)
```

To each data set we will fit a mixed effects linear model and store the fitted models in a list.

```
BS.models <- map(BS.data,
                 \(d) lmer(Y ~ 1 + (1 | risk),
                           data = d,
                           weights = Wt))
```

Next, we compute the fitted values and standardized residuals for each model and append them to the data set. We also calculate the slopes of the fitted linear regression where the response variable is the standardized residual and the predictor variable is fitted value.

```
BS.FV.Res <- map(BS.models,
                 \(m) {
                   tb <- getData(m)
                   tb$mu <- fitted(m)
                   tb$rsP <- resid(m, type = "pearson") / sigma(m)
                   return(tb)})

BS.slopes <- map_dbl(BS.FV.Res,
                     \(tb) {
                       fm <- lm(rsP ~ mu,
                                data = tb)
                       sfm <- summary(fm)
                       return(coef(sfm)[2,1])})

BS.pvals <- map_dbl(BS.FV.Res,
                    \(tb) {
                      fm <- lm(rsP ~ mu,
                               data = tb)
                      sfm <- summary(fm)
                      return(coef(sfm)[2,4])})
```

We collect all the information into a single data frame and create a categorical variable to identify the simulation.

```
BS.results <- reduce(BS.FV.Res, bind_rows)
BS.results$sim.label <- factor(BS.results$sim.label)
BS.results$sim.label <- fct_relevel(
  BS.results$sim.label,
  str_c(str_sub(names(BS.slopes), 4),
        " Obs. per Risk")[order(BS.slopes,
                                decreasing = TRUE)])
```

Figure A.1 shows the results where we can see that as the number of observations increases the slope of the line decreases. For this particular set of simulated data, the slopes for the 5, 10, and 20 observations per risks sets are statistically different from zero. But the not for the 40 observations per risk set.

```
tb <- round(rbind(BS.slopes,
      BS.pvals), 4)
dimnames(tb) <- list(c("Slope", "P-Value"),
                     c("5 Obs.", "10 Obs.", "20 Obs.", "40 Obs."))
tb
```

```
        5 Obs. 10 Obs. 20 Obs. 40 Obs.
Slope   0.0393  0.0177   0.008  0.0031
P-Value 0.0000  0.0001   0.007  0.0889
```

```
ggplot(data = BS.results,
       mapping = aes(x = mu,
                     y = rsP)) +
  facet_wrap(vars(sim.label)) +
  geom_point(pch = 1, alpha = 0.2) +
  geom_smooth(method = "lm") +
  labs(x = "Fitted Values",
       y = "Standardized Residuals")
```

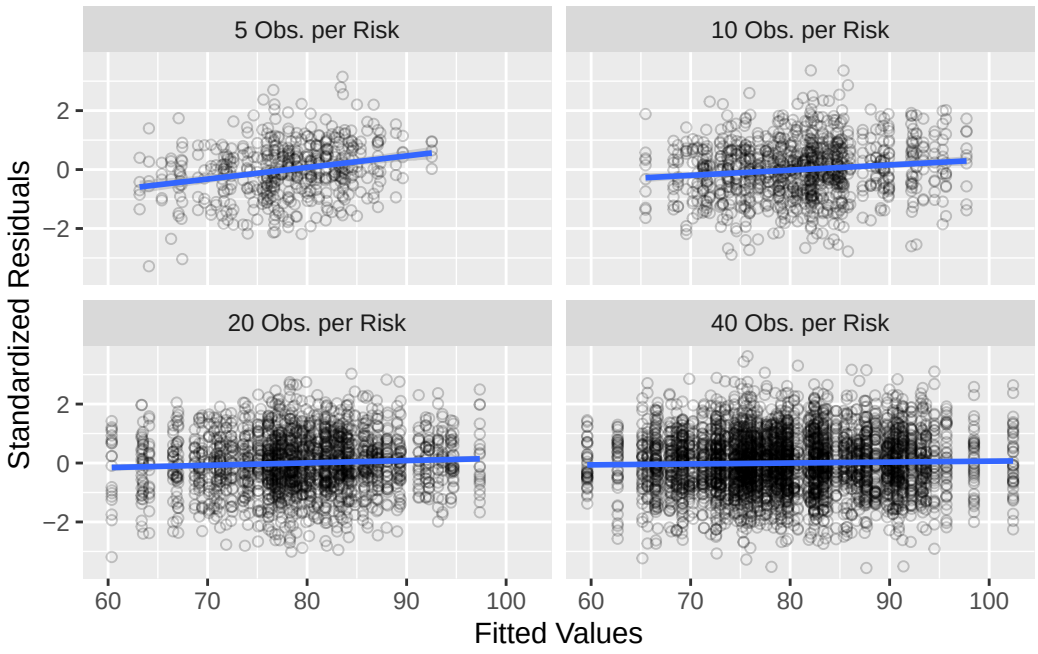


Figure A.1: Results of fitting a linear regression model to simulated data from the Bühlmann-Straub model. The panels have been arranged from the largest slope in the upper-left (5 Obs. per Risk) to the smallest slope in the lower-right (40 Obs. per Risk).

B Equivalence of Credibility Matrices

In Chapter 3 we found an expression for the credibility matrix A_j as shown in Equation 3.30 and in Section 4.3, using linear mixed model theory, we found another expression (Equation 4.2). We want to show here that these two, seemingly different, expressions are equivalent.

Equation 4.2 says that, assuming the matrices D and W_j have the forms

$$D = \begin{bmatrix} \tau_0^2 & 0 \\ 0 & \tau_1^2 \end{bmatrix} \qquad \text{and} \qquad W_j = \begin{bmatrix} w_{j\bullet} & \sum_t t w_{jt} \\ \sum_t t w_{jt} & \sum_t t^2 w_{jt} \end{bmatrix},$$

then the credibility matrix A_j is given by

$$A_j = \frac{\det(DW_j) I_2 + \hat{\sigma}^2 DW_j}{\det(DW_j) + \hat{\sigma}^2 \text{trace}(DW_j) + \hat{\sigma}^4}. \tag{B.1}$$

We want to show that this is equivalent to Equation 3.30, that is,

$$A_j = \frac{w_{j\bullet}}{d} \begin{bmatrix} w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_1 & \kappa_1 E_j^{(s)}[t] \\ \kappa_0 E_j^{(s)}[t] & w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_0 E_j^{(s)}[t^2] \end{bmatrix}, \tag{B.2}$$

where

$$d = (w_{j\bullet} + \kappa_0)\left(w_{j\bullet} E_j^{(s)}[t^2] + \kappa_1\right) - \left(w_{j\bullet} E_j^{(s)}[t]\right)^2$$

and $\kappa_0 = \sigma^2/\tau_0^2$ and $\kappa_1 = \sigma^2/\tau_1^2$.

The notations $E_j^{(s)}[t]$ and $\text{Var}_j^{(s)}[t]$ are shorthand for

$$E_j^{(s)}[t] = \sum_t \frac{w_{jt}}{w_{j\bullet}} t \qquad \text{and} \qquad \text{Var}_j^{(s)}[t] = E_j^{(s)}[t^2] - \left(E_j^{(s)}[t]\right)^2.$$

Equivalence of the Denominator

We will start by showing the that the denominator of Equation B.1 is equal to d in Equation B.2.

The matrix DW_j is equal to

$$\begin{bmatrix} \tau_0^2 w_{j\bullet} & \tau_0^2 \sum_t t w_{jt} \\ \tau_1^2 \sum_t t w_{jt} & \tau_1^2 \sum_t t^2 w_{jt} \end{bmatrix}$$

and hence its determinant is

$$\det(DW_j) = \tau_0^2\tau_1^2 w_{j\bullet}\sum_t t^2 w_{jt} - \tau_0^2\tau_1^2\left(\sum_t t w_{jt}\right)^2,$$

and its trace is

$$\text{trace}(DW_j) = \tau_0^2 w_{j\bullet} + \tau_1^2\sum_t t^2 w_{jt}.$$

Therefore, the denominator is equal to

$$\tau_0^2\tau_1^2 w_{j\bullet}\sum_t t^2 w_{jt} - \tau_0^2\tau_1^2\left(\sum_t t w_{jt}\right)^2 + \sigma^2\tau_0^2 w_{j\bullet} + \sigma^2\tau_1^2\sum_t t^2 w_{jt} + \sigma^4.$$

In the above expression we can replace the sums with the $E_j^{(s)}$ notation to obtain

$$\tau_0^2\tau_1^2 w_{j\bullet}^2 E_j^{(s)}[t^2] - \tau_0^2\tau_1^2 w_{j\bullet}^2\left(E_j^{(s)}[t]\right)^2 + \sigma^2\tau_0^2 w_{j\bullet} + \sigma^2\tau_1^2 w_{j\bullet}E_j^{(s)}[t^2] + \sigma^4.$$

Next we will factor out $\tau_0^2\tau_1^2$ yielding

$$\tau_0^2\tau_1^2\left[w_{j\bullet}^2 E_j^{(s)}[t^2] - \left(w_{j\bullet}E_j^{(s)}[t]\right)^2 + \kappa_1 w_{j\bullet} + \kappa_0 w_{j\bullet}E_j^{(s)}[t^2] + \kappa_0\kappa_1\right]$$

and now we just rearrange like terms to get

$$\tau_0^2\tau_1^2\left[(w_{j\bullet}+\kappa_0)\left\{w_{j\bullet}E_j^{(s)}[t^2] + \kappa_1\right\} - \left(w_{j\bullet}E_j^{(s)}[t]\right)^2\right].$$

Apart from the factor $\tau_0^2\tau_1^2$ this is the same expression for the denominator in Equation B.2 that we wanted to show. The factor of $\tau_0^2\tau_1^2$ will also appear in our calculations of the numerator and thus it will cancel out.

Equivalence of the Numerators

The numerator in equations Equation B.1 and Equation B.2 is a 2×2 matrix and so we will show the equivalence of the $(1,1)$, $(1,2)$, and $(2,2)$ entries. The entry at $(2,1)$ is similar to the $(1,2)$.

The $(1,1)$ entry in Equation B.1 is equal to

$$\tau_0^2\tau_1^2 w_{j\bullet}\sum_t t^2 w_{jt} - \tau_0^2\tau_1^2\left(\sum_t t w_{jt}\right)^2 + \sigma^2\tau_0^2 w_{j\bullet}$$

Again we will factor $\tau_0^2\tau_1^2$ out to get

$$\tau_0^2\tau_1^2\left[w_{j\bullet}\sum_t t^2 w_{jt} - \left(\sum_t t w_{jt}\right)^2 + \kappa_1 w_{j\bullet}\right].$$

Next, we multiply the first factor inside the square brackets by $w_{j\bullet}/w_{j\bullet}$ and the second factor by $w_{j\bullet}^2/w_{j\bullet}^2$. This will allow us to rewrite these two items using $\text{Var}_j^{(s)}[t]$ as follows

$$\tau_0^2\tau_1^2\left[w_{j\bullet}^2\text{Var}_j^{(s)}[t] + \kappa_1 w_{j\bullet}\right].$$

Now we factor out $w_{j\bullet}$ to obtain our final result

$$\tau_0^2\tau_1^2 w_{j\bullet}\left[w_{j\bullet}\text{Var}_j^{(s)}[t] + \kappa_1\right],$$

which apart from the $\tau_0^2\tau_1^2$ factor matches the $(1,1)$ entry in Equation B.2.

For the $(1,2)$ entry we start with

$$\sigma^2\tau_0^2\sum_t t w_{jt}.$$

Multiply this expression by $w_{j\bullet}/w_{j\bullet}$ to obtain

$$\sigma^2\tau_0^2 w_{j\bullet}\sum_j t\frac{w_{jt}}{w_{j\bullet}} = \sigma^2\tau_0^2 w_{j\bullet}E_j^{(s)}[t] = \tau_0^2\tau_1^2 w_{j\bullet}\left[\kappa_1 E_j^{(s)}[t]\right].$$

Again, apart from the factor $\tau_0^2\tau_1^2$, this is the expression we wanted to arrive at.

Finally, for the $(2,2)$ entry we begin with

$$\tau_0^2\tau_1^2 w_{j\bullet}\sum_t t^2 w_{jt} - \tau_0^2\tau_1^2\left(\sum_t t w_{jt}\right)^2 + \sigma^2\tau_1^2\sum_t t^2 w_{jt}.$$

Multiplying the first and the last terms by $w_{j\bullet}/w_{j\bullet}$ and the second term by $w_{j\bullet}^2/w_{j\bullet}^2$ and simplifying by using the $E_j^{(s)}[]$ notation we get

$$\tau_0^2\tau_1^2 w_{j\bullet}^2 E_j^{(s)}[t^2] - \tau_0^2\tau_1^2 w_{j\bullet}^2\left(E_j^{(s)}[t]\right)^2 + \sigma^2\tau_1^2 w_{j\bullet}^2 E_j^{(s)}[t^2].$$

Now factor out $\tau_0^2\tau_1^2 w_{j\bullet}$ to arrive at

$$\tau_0^2\tau_1^2 w_{j\bullet}\left[w_{j\bullet}\text{Var}_j^{(s)}[t] - \kappa_0 E_j^{(s)}[t^2]\right].$$

The factor $\tau_0^2\tau_1^2$ will cancel with the same factor we have in the denominator and so we have shown the equivalence of both expressions for the credibiliy matrix A_j.