Regression mandatory assignment

Group 28: Anton Bendsen (glq892), William Lin (lgt226) and Magnus Sivertsen (rpv759)

30/10-2024

Contents

Contributions to the assignment:

The assignment has been made in cooperation between all three students, who have contributed roughly equally to the assignment.

Theoretical exercises on the Tweedie distributions

The Tweedie exponential dispersion model has variance function

$$\mathcal{V}(\mu) = \mu^k$$

for $k \in \mathbf{R}\backslash(0,1)$*, i.e., k can take any real value except values in the interval* $(0,1)$*. The nuisance parameter k is called the Tweedie index parameter. If Y follows a Tweedie exponential dispersion model, we write* $Y \sim Tw_k(\mu, \psi)$*.*

- *Which common distributions are the Tweedie distributions in the special cases of* $k = 0, 1, 2,$ *and* 3*? (It is sufficient to list them.)*

To answer this question, we use knowledge about the variance functions for different known distributions. From page 116 we are given that for a normal $\mathcal{N}(\mu, \sigma^2)$, $\kappa(\theta) = \frac{\theta^2}{2}$ which implies that $\kappa'(\theta) = \theta$ and $\kappa''(\theta) = 1$. From Definition 5.9. the variance function is thus $\mathcal{V}(\mu) = 1$.

For $k = 1$ the Tweedie distribution has variance function: $\mathcal{V}(\mu) = \mu$. From Example 5.13. it is stated for a Poisson distribution with mean $\mu > 0$ that $\kappa(\theta) = e^\theta$ which implies $\kappa'(\theta) = \kappa''(\theta) = e^\theta$. It has therefore canonical link function $g(\mu) = \log(\mu)$, and by using this in the formula on page 121 we get:

$$\mathcal{V}(\mu) = \kappa''(g(\mu)) = e^{\log(\mu)} = \mu.$$

Hence exactly the same variance function as the Tweedie distibution for $k = 1$.

For $k = 2$ the Tweedie distribution has variance function: $\mathcal{V}(\mu) = \mu^2$. In Example 5.7. it is stated that for at $\Gamma(\lambda, \alpha)$ with λ being the shape parameter, and α the scale parameter, the unit cumulant function is given as $\kappa(\theta) = -\log(-\theta)$, thus $\kappa'(\theta) = \frac{-1}{\theta}$ and $\kappa''(\theta) = \frac{1}{\theta^2}$. Notice the canonical link is $g(\mu) = \frac{-1}{\mu}$. From the formula on page 121 we get

$$\mathcal{V}(\mu) = \kappa''(g(\mu)) = \frac{1}{\left(\frac{-1}{\mu}\right)^2} = \mu^2$$

This is exactly the variance function of the Tweedie distribution for $k = 2$.

For $k = 3$ the Tweedie distribution has variance function: $\mathcal{V}(\mu) = \mu^3$. By recalling Exercise 5.6 from week 3, we calculated that for an inverse Gaussian distrubiton $\mathcal{V}(\mu) = \mu^3$, exactly the same as the Tweedie distribution.

To sum up, the following k-values makes the Tweedie distrubtion these common distibutions:

- $k = 0$: A $\mathcal{N}(\mu, \sigma^2)$-distribution.
- $k = 1$: A Pois(μ)-distribtion.
- $k = 2$: A $\Gamma(\lambda, \alpha)$-disribution, with shape λ and scale α.
- $k = 3$: An inverse Gaussian distribution.

In the following, we assume $k \geq 1$. Then $\mu > 0$.

- *Find $\mu(\theta)$.*

From Theorem 5.8. and Definition 5.9. we have

$$\kappa'(\theta) = \mu(\theta),$$
$$\mathcal{V}(\mu(\theta)) = \kappa''(\theta) = \mu(\theta)^k.$$

At first, assume $k \neq 1$. From the two equations above we get

$$\kappa''(\theta) = \frac{d\mu}{d\theta} = \mu^k \quad \iff \quad \mu^{-k} d\mu = d\theta \quad \iff \quad \int \mu^{-k} d\mu = \int d\theta \quad \iff \quad \frac{1}{1-k}\mu^{1-k} = \theta + C,$$

for a constant C. Notice that since $k \neq 1$, we do not divide by 0. We then get

$$\frac{1}{1-k}\mu^{1-k} = \theta + C \quad \iff \quad \mu^{1-k} = (1-k)(\theta + C) \quad \iff \quad \mu(\theta) = ((1-k)(\theta + C))^{\frac{1}{1-k}},$$

for $k \neq 1$.

For $k = 1$ we have

$$\kappa''(\theta) = \frac{d\mu}{d\theta} = \mu^k = \mu \quad \iff \quad \mu^{-1} d\mu = d\theta \quad \iff \quad \int \mu^{-1} d\mu = \int d\theta \quad \iff \quad \log|\mu| = \log(\mu) = \theta + C,$$

for a constant C, and we have used that since $\mu > 0$, $|\mu| = \mu$. We now get

$$\log(\mu) = \theta + C \quad \iff \quad \mu(\theta) = e^{\theta + C},$$

for $k = 1$.

We can therefore write $\mu(\theta)$ as follows

$$\mu(\theta) = \begin{cases} ((1-k)(\theta + C))^{\frac{1}{1-k}}, & k \neq 1 \\ e^{\theta + C}, & k = 1 \end{cases}.$$

Notice the canonical link is by Definition 5.12 $g = (\kappa')^{-1} = \mu^{-1}$.

$$g(\mu) = \theta(\mu) = \begin{cases} \frac{1}{1-k}\mu^{1-k} - C, & k \neq 1 \\ \log(\mu) - C, & k = 1 \end{cases}.$$

- *Show that up to an additive constant, the unit cumulant function is given by*

$$\kappa(\theta(\mu)) = \begin{cases} \frac{\mu^{2-k}}{2-k}, & k \neq 2 \\ \log(\mu), & k = 2 \end{cases}$$

At first, assume $k \neq 1$ and $k \neq 2$. We see

$$\kappa(\theta) = \int \kappa'(\theta) d\theta = \int \mu(\theta) d\theta = \int ((1-k)(\theta + C))^{\frac{1}{1-k}} d\theta = (1-k)^{\frac{1}{1-k}} \int (\theta + C)^{\frac{1}{1-k}} d\theta.$$

with $t = \theta + C$, we get $\frac{dt}{d\theta} = 1$ and thus $dt = d\theta$. Hence by integration by substitution, we get

$$\kappa(\theta) = (1-k)^{\frac{1}{1-k}} \int t^{\frac{1}{1-k}} dt = (1-k)^{\frac{1}{1-k}} \left(\frac{1}{1 + \frac{1}{1-k}} t^{1 + \frac{1}{1-k}} + A \right) = (1-k)^{\frac{1}{1-k}} \frac{1-k}{2-k} (\theta + C)^{1 + \frac{1}{1-k}} + B,$$

for constants A and B. Notice that since we assume $k \notin \{1, 2\}$, we do not divide by 0. Recall that

$$\theta + C = \frac{1}{1-k} \mu^{1-k},$$

for $k \neq 1$. We thus get

$$\kappa(\theta(\mu)) = (1-k)^{\frac{1}{1-k}} \frac{1-k}{2-k} \left(\frac{1}{1-k} \mu^{1-k} \right)^{1 + \frac{1}{1-k}} + B = \frac{1}{2-k} \frac{(1-k)^{1 + \frac{1}{1-k}}}{(1-k)^{1 + \frac{1}{1-k}}} \mu^{1-k+1} + B = \frac{1}{2-k} \mu^{2-k} + B.$$

Assume now $k = 2$. We have

$$\kappa(\theta) = (1-2)^{\frac{1}{1-2}} \int (\theta + C)^{\frac{1}{1-2}} d\theta = - \int (\theta + C)^{-1} d\theta.$$

With $t = \theta + C$, $\frac{dt}{d\theta} = 1$ and thus $dt = d\theta$, we get

$$\kappa(\theta) = - \int t^{-1} dt = -(\log |t| + A) = - \log |\theta + C| + B,$$

for constants A and B. Since

$$\theta + C = \frac{1}{1-k} \mu^{1-k} = \frac{1}{1-2} \mu^{1-2} = -\mu^{-1},$$

we get the following

$$\kappa(\theta(\mu)) = - \log | - \mu^{-1}| + B = - \log(\mu^{-1}) + B = \log(\mu) + B.$$

Here we have used that since $\mu > 0$, $| - \mu^{-1}| = \mu^{-1}$.

Assume now $k = 1$. We see

$$\kappa(\theta) = \int \kappa'(\theta) d\theta = \int \mu(\theta) d\theta = \int e^{\theta + C} d\theta.$$

With $t = \theta + C$, $\frac{dt}{d\theta} = 1$ and thus $dt = d\theta$, we get

$$\kappa(\theta) = \int e^t dt = e^t + B = e^{\theta + C} + B$$

for a constant B. Recall that for $k = 1$ we have

$$\theta + C = \log(\mu).$$

We thus get

$$\kappa(\theta(\mu)) = e^{\log(\mu)} + B = \mu + B = \frac{1}{2-1}\mu^{2-1} + B = \frac{1}{2-k}\mu^{2-k} + B.$$

We have thus shown

$$\kappa(\theta(\mu)) = \begin{cases} \frac{\mu^{2-k}}{2-k}, & k \neq 2 \\ \log(\mu), & k = 2 \end{cases}$$

up to an additive constant.

- *Compute the canonical link function for the Tweedie exponential dispersion models.*

We have already found the canonical link before:

$$g(\mu) = \theta(\mu) = \begin{cases} \frac{1}{1-k}\mu^{1-k} - C, & k \neq 1 \\ \log(\mu) - C, & k = 1 \end{cases}.$$

- *Compute the unit deviance for the Tweedie exponential dispersion models.*

Hint: If $Y \sim Tw_k(\mu, \psi)$, then $P(Y = 0) > 0$ for $1 \leq k < 2$, and $P(Y = 0) = 0$ for $k \geq 2$.

To find the unit deviance for the Tweedie exponential dispersion models, we use the remark for Definition 5.15. on page 123. Recall the canonical link and unit cumulant function

$$g(\mu) = \begin{cases} \frac{1}{1-k}\mu^{1-k} - C, & k \neq 1 \\ \log(\mu) - C, & k = 1 \end{cases} \quad \text{and} \quad \kappa(\theta(\mu)) = \begin{cases} \frac{\mu^{2-k}}{2-k}, & k \neq 2 \\ \log(\mu), & k = 2 \end{cases}$$

Let $k = 1$. We see for $Y \in J = (0, \infty)$

$$\begin{aligned} d(Y, \mu) &= 2\left(Y(\log(Y) - \log(\mu)) - \frac{1}{2-1}Y^{2-1} + \frac{1}{2-1}\mu^{2-1}\right) \\ &= 2\left(Y\log\left(\frac{Y}{\mu}\right) - Y + \mu\right). \end{aligned}$$

For $Y = 0$, it is clear from Definition 5.15 (and Example 5.17) that

$$d(0, \mu) = 2\mu.$$

Now, let $k = 2$. For $Y \in J = (0, \infty)$ (note that $P(Y = 0) = 0$ for $k = 2$) we have

$$\begin{aligned} d(Y, \mu) &= 2\left(Y\left(\frac{1}{1-2}Y^{1-2} - \frac{1}{1-2}\mu^{1-2}\right) - \log(Y) + \log(\mu)\right) \\ &= 2\left(Y\left(\mu^{-1} - Y^{-1}\right) + \log\left(\frac{\mu}{Y}\right)\right) \\ &= 2\left(\log\left(\frac{\mu}{Y}\right) + \frac{Y}{\mu} - 1\right). \end{aligned}$$

For $k \neq 1, 2$, we have for $Y \in J = (0, \infty)$

$$\begin{aligned} d(Y, \mu) &= 2\left(Y\left(\frac{1}{1-k}Y^{1-k} - \frac{1}{1-k}\mu^{1-k}\right) - \frac{1}{2-k}Y^{2-k} + \frac{1}{2-k}\mu^{2-k}\right) \\ &= \frac{2}{1-k}\left(Y^{2-k} - Y\mu^{1-k}\right) + \frac{2}{2-k}\left(\mu^{2-k} - Y^{2-k}\right). \end{aligned}$$

In the case where $1 < k < 2$, we know $P(Y = 0) > 0$. In this case, for $Y = 0$ we have

$$\sup_{\mu_0 \in J} \{g(\mu_0)Y - \kappa(g(\mu_0))\} = \sup_{\mu_0 \in J} \left\{ -\frac{\mu_0^{2-k}}{2-k} \right\} = 0.$$

Hence from Definition 5.15. we simply get

$$\begin{aligned} d(0, \mu) &= 2\left(0 + 0 + \frac{\mu^{2-k}}{2-k}\right) \\ &= \frac{2\mu^{2-k}}{2-k}. \end{aligned}$$

We now conclude that the unit deviance is given as

$$d(Y, \mu) = \begin{cases} 2\left(Y \log\left(\frac{Y}{\mu}\right) - Y + \mu\right), & k = 1 \\ 2\left(\log\left(\frac{\mu}{Y}\right) + \frac{Y}{\mu} - 1\right), & k = 2 \\ \frac{2}{1-k}\left(Y^{2-k} - Y\mu^{1-k}\right) + \frac{2}{2-k}\left(\mu^{2-k} - Y^{2-k}\right), & k \neq 1, 2 \end{cases}$$

for $Y \in J = (0, \infty)$. In the case where $k = 1$ and $Y = 0$, we have

$$d(0, \mu) = 2\mu.$$

And in the case where $1 < k < 2$ and $Y = 0$, we have

$$\frac{2}{2-k}\mu^{2-k}.$$

- *Usually, it is not the canonical link function that is used when using the Tweedie exponential dispersion model in a generalized linear model, but the link function* $\log(\mu(\eta))$*, where* η *is the linear predictor. Find the function* $\theta(\eta)$ *for the log link function.*

Since the link function is $\log(\mu(\eta)) = \eta$, we have

$$\mu(\eta) = e^{\eta}.$$

Recall

$$\theta(\mu) = \begin{cases} \frac{1}{1-k}\mu^{1-k} - C, & k \neq 1 \\ \log(\mu) - C, & k = 1 \end{cases}.$$

By plugging $\mu(\eta) = e^{\eta}$ in we get

$$\begin{aligned} \theta(\eta) &= \begin{cases} \frac{1}{1-k}(e^{\eta})^{1-k} - C, & k \neq 1 \\ \log(e^{\eta}) - C, & k = 1 \end{cases} \\ &= \begin{cases} \frac{1}{1-k}e^{\eta(1-k)} - C, & k \neq 1 \\ \eta - C, & k = 1 \end{cases}. \end{aligned}$$

The Tweedie exponential dispersion model for $1 < k < 2$ *can be used to model continuous data with exact zeros. Assume* N *follows a Poisson distribution with mean* λ^**, and assume that* Z_i *follows a gamma distribution with mean* μ^* *and dispersion parameter* ψ^* *for* $i = 1, \ldots, N$*. Then (it can be used without proof)* $Y = \sum_{i=1}^{N} Z_i$ *where* $Y = 0$ *for* $N = 0$*, follows a Tweedie exponential dispersion model* $Tw_k(\mu, \psi)$ *with* $\lambda^* = \frac{\mu^{2-k}}{\psi(2-k)}$*;* $\mu^* = (2-k)\psi\mu^{k-1}$*;* $\psi^* = (2-k)(k-1)\psi^2\mu^{2(k-1)}$*.*

- *Compute the probability that* $Y = 0$*.*

We note that since $Y = \sum_{i=1}^{N} Z_i$, and Z_i is gamma distributed and thus $Z_i > 0$ for all $i = 1, \ldots, N$, $Y = 0$ if and only if $N = 0$. Since N is Poisson distributed with mean λ^*, we get

$$P(Y = 0) = P(N = 0) = \frac{e^{-\lambda^*}(\lambda^*)^0}{0!} = e^{-\lambda^*}.$$

- *Find the parameter μ as a function of λ^*, μ^* and ψ^*. Interpret the parameter μ.*

From the equation $\lambda^* = \frac{\mu^{2-k}}{\psi(2-k)}$ we get

$$\lambda^* = \frac{\mu^{2-k}}{\psi(2-k)} \quad \Longleftrightarrow \quad \lambda^*\psi(2-k) = \mu^{2-k} \quad \Longleftrightarrow \quad \lambda^*(2-k)\psi\mu^{k-1} = \mu^{2-k}\mu^{k-1} = \mu.$$

We know $\mu^* = (2-k)\psi\mu^{k-1}$, hence we get the following

$$\lambda^*(2-k)\psi\mu^{k-1} = \mu \quad \Longleftrightarrow \quad \mu(\lambda^*, \mu^*, \psi^*) = \lambda^*\mu^*.$$

We see that the mean of Y, μ, is the product of λ^* and μ^*. Since $Y = \sum_{i=1}^{N} Z_i$, we can interpret μ the as the average total value of a series of events, where λ^* is the average number of times an event has happened, and μ^* is the average value of each event.

Exploratory analysis

We want to make an exploratory analysis of the data about rainfall in Eromanga, Queensland, Australia. Before we do so, we import all relevant packages:

```
library(ggplot2)
library(reshape2)
library(splines)
library(RwR)
library(grid)
library(gridExtra)
library(xtable)
library(lattice)
library(pracma)
library(GLMsData)
library(statmod)
library(tweedie)
library(mgcv)
library(tidyverse)
library(hexbin)
library(latticeExtra)
```

Now the packages are loaded, hence we can import the data. Here one must be careful to select the correct classes for each variable. Especially "Phase" has to be a factor and NOT a numeric variable. This is the case since the numbers 1-5 in Phase are different categories, and not labelled in any order.

```
Rain.data = read.table("RaindataEromanga.txt", header = TRUE,
  colClasses = c("numeric", "numeric", "factor", "numeric", "factor"))
```

Now that the data has been imported, we take a quick glance at it:

```
head(Rain.data)
```

```
##   Year Rain Month   SOI Phase
## 1 1905  0.0     7 -19.8     1
## 2 1906 20.8     7   6.3     4
## 3 1907 12.0     7  -5.1     5
## 4 1908   NA     7  -3.2     5
## 5 1909   NA     7   9.9     2
## 6 1910   NA     7  19.0     2
```

```
summary(Rain.data)
```

```
##       Year           Rain         Month        SOI            Phase
##  Min.   :1905   Min.   :  0.00   7:120   Min.   :-21.0000   1:18
##  1st Qu.:1935   1st Qu.:  0.00           1st Qu.: -5.3575   2:26
##  Median :1964   Median :  4.60           Median :  0.8000   3: 9
##  Mean   :1964   Mean   : 15.13           Mean   :  0.4817   4:30
##  3rd Qu.:1994   3rd Qu.: 20.70           3rd Qu.:  5.8500   5:37
##  Max.   :2024   Max.   :137.10           Max.   : 26.3000
##                 NA's   :9
```

We can quickly see that there are 9 years where data for rainfall is missing. We also see that these are the only 9 cases in the whole data set where some values are missing. To conclude what to do with these data points, we take a glance at them:

```
missingdata <- Rain.data[is.na(Rain.data$Rain), ]
missingdata
```

```
##     Year Rain Month    SOI Phase
## 4   1908   NA     7  -3.20     5
## 5   1909   NA     7   9.90     2
## 6   1910   NA     7  19.00     2
## 7   1911   NA     7 -11.90     1
## 29  1933   NA     7   3.30     4
## 117 2021   NA     7  16.26     4
## 118 2022   NA     7   7.63     2
## 119 2023   NA     7  -3.32     5
## 120 2024   NA     7  -5.83     5
```

It can be seen above that most of the missing data are clutted together in 4-year clumps starting in years 1908 and 2021. This could indicate some sort of pattern in the missing data, which could suggest MAR. We want to study this missingness further by comparing the summary of the missing data set and the data set with the NAs removed.

```
Rain.data.small<-Rain.data[,c("Year","Rain", "SOI","Phase")]
Rain.data.small <- subset(Rain.data.small, !is.na(Rain))
```

```
summary(missingdata)
```

```
##       Year           Rain     Month      SOI          Phase
##  Min.   :1908   Min.   : NA   7:9   Min.   :-11.900   1:1
##  1st Qu.:1910   1st Qu.: NA         1st Qu.: -3.320   2:3
##  Median :1933   Median : NA         Median :  3.300   3:0
##  Mean   :1962   Mean   :NaN         Mean   :  3.538   4:2
##  3rd Qu.:2022   3rd Qu.: NA         3rd Qu.:  9.900   5:3
##  Max.   :2024   Max.   : NA         Max.   : 19.000
##                 NA's   :9
```

```
summary(Rain.data.small)
```

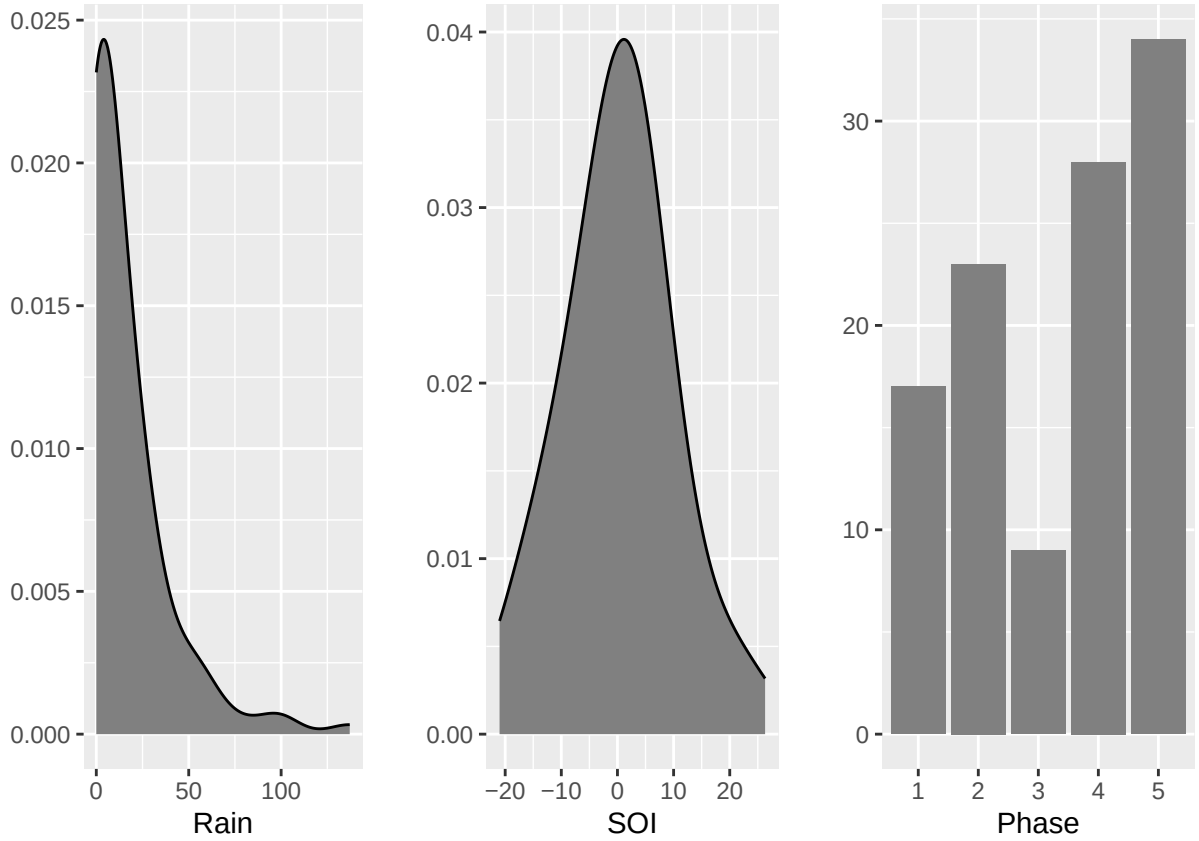
```
##       Year           Rain             SOI           Phase
##  Min.   :1905   Min.   :  0.00   Min.   :-21.0000   1:17
##  1st Qu.:1938   1st Qu.:  0.00   1st Qu.: -5.5300   2:23
##  Median :1965   Median :  4.60   Median :  0.8000   3: 9
##  Mean   :1965   Mean   : 15.13   Mean   :  0.2339   4:28
##  3rd Qu.:1992   3rd Qu.: 20.70   3rd Qu.:  5.1500   5:34
##  Max.   :2020   Max.   :137.10   Max.   : 26.3000
```

We see that the distributions for SOI in the data sets look similar where the minimum value and maximum value for the missing data are numerically smaller, but so are its quantiles and the mean. We also see that when looking at Phase, the distributions of the data sets also look similar. We suspect removing the data points with NAs will thus not influence the result too much. As we will see later, Year is not a significant predictor for Rain, but SOI and Phase are. And when looking at the SOI and Phase variables above, it seems the missingness of Rain does not depend on these variables. Thus when predicting Rain, if we only look at the variables SOI and Phase, we suspect the data about rainfall is missing completely at random (MCAR). Even though we can not be entirely sure about why the data is missing, we decide to throw away the data with missing values about rainfall. We understand that if we are wrong about this assumption, our results might be biased cf. page 99. We also remove the column for month, since this variable is constantly equal to 7, which makes it obsolete.

To get a better understanding about the data, we plot the densities of the two continous variables Rain and SOI, and a bar plot for the discrete variable Phase:

```
tmp<-lapply(names(Rain.data.small), function(x)
  ggplot(data=Rain.data.small[, x, drop=FALSE])+
    aes_string(x)+xlab(x)+ylab(""))
gd<-geom_density(adjust=2, fill=gray(0.5)) #Define function for density plot
gb<-geom_bar(fill=gray(0.5)) #Define function for barplot
```

```
grid.arrange(tmp[[2]]+gd,tmp[[3]]+gd,tmp[[4]]+gb, ncol=3)
```

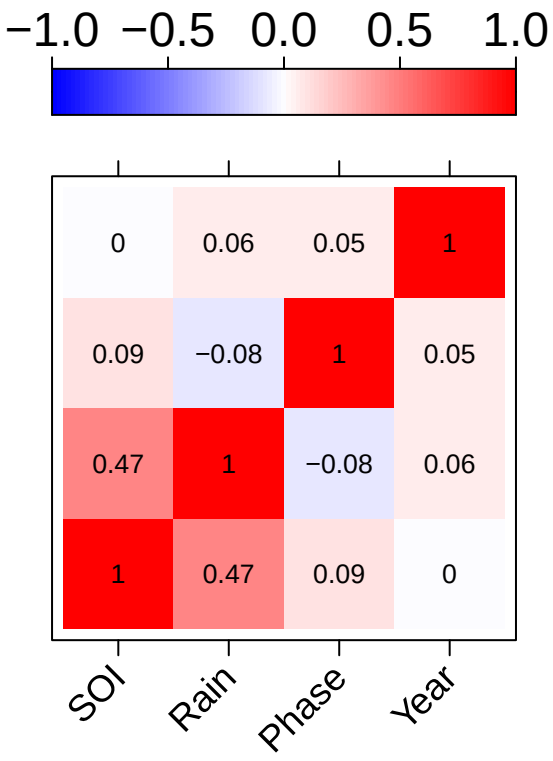


By studying the above plots, it is clear that the distribution of Rain is quite skewed. This is probably the case, since most years have either moderate amount of rain or no Rain at all in July, whereas other years experience more extreme weather in July. This might make it harder to draw conclusions for extreme amounts of rain. The distribution of SOI looks quite symmetric around 0, and only vaguely skewed. Phase seems to be a bit skewed, with only very few observations for Phase=3, however the other four phases all have roughly between 20 and 30 observations each.

We have now studied the different variables in the data set, however we would also like to find out how they correlate with each other. To do so, we make a plot of the Spearman correlations using formula from p. 71, however we also add the values of the Spearman correlations to make the results more clear. It has to be Spearman Correlation and not Pearson, as the variable Phase is categorical, something only Spearman can handle:

```
cp <- cor(data.matrix(Rain.data.small), method = "spearman")

ord <- rev(hclust(as.dist(1 - abs(cp)))$order)

colPal <- colorRampPalette(c("blue", "white", "red"), space = "rgb")(100)

plt <- levelplot(cp[ord, ord], xlab = "", ylab = "", col.regions = colPal,
                 at = seq(-1, 1, length.out = 100),
                 colorkey = list(space = "top", labels = list(cex = 1.5)),
                 scales = list(x = list(rot = 45),
                               y = list(draw = FALSE),
                               cex = 1.2))
box_ <- cp[ord,ord]
```

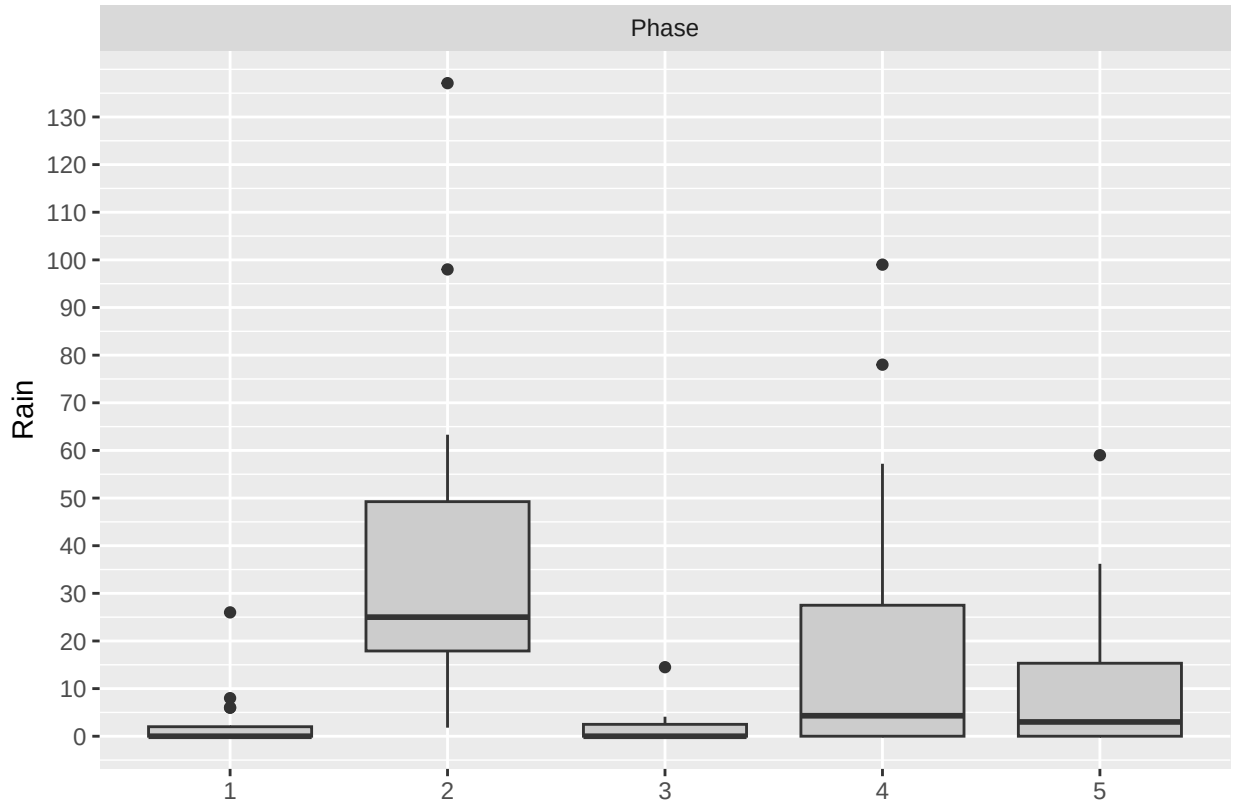
```
plt + layer(panel.text(row(box_), col(box_),
                       labels = round(box_, 2),
                       col = "black", cex = 0.8))
```



On the matrix above, one can see a positive correlation between SOI and Rain, and a slightly negative correlation between Phase and Rain. Year seems to be weakly correlated with Rain, and SOI is weakly correlated with SOI phase. Even though the above plots show how the different variables correlate, it not fully possible to decide which variables to include in a predictive model for rainfall yet. This will require further analysis.

We would like to find out out how Phase affects Rain. To do so, boxplots of Rain for each Phase are plotted below:

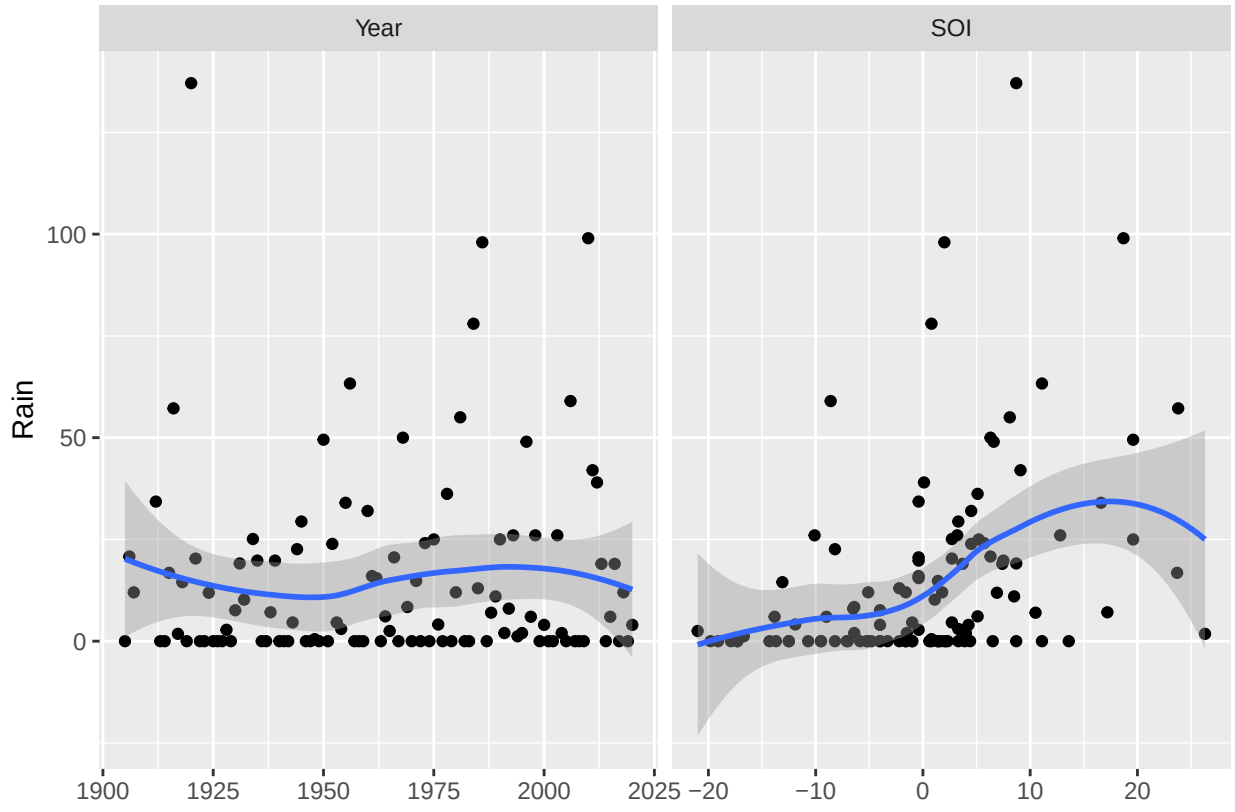
```
mRain<-melt(Rain.data.small[, c("Rain","Phase")], id="Rain")
ggplot(mRain,
       aes(x=factor(value), y=Rain))+
  geom_boxplot(fill=I(gray(0.8)))+xlab("")+
  facet_wrap(~variable, scale="free_x",ncol=5)+
   scale_y_continuous(breaks = seq(0,
                                   max(mRain$Rain),
                                   by = 10))
```



The boxplots above suggest that Phase has an effect on the amount of rainfall. Where groups 1 and 3 both have very low observations for rain, with only a few outliers with more rain than 5mm. Groups 2 and 4 on the other hand has significantly larger variation in data, and generally much larger observations of rainfall. The same is partially true for group 5, though a bit more moderately. From the plot above one would find it sensible to include Phase as a covariate in a model predicting rainfall in Eromanga.

For the two other variables Year and SOI, we would like to find out how well Rain is explained by the covariates. We therefore make scatter plots for the two including a "geom_smooth"-line and confidence interval bands:

```
mmRain<-melt(Rain.data.small[, c("Year","SOI","Rain")], id.vars="Rain")

binScale <- scale_fill_continuous(breaks = c(1, 10, 100, 1000),
  low = "gray80", high = "black",
  trans = "log", guide = "none")
ggplot(mmRain, aes(x = value, y = Rain)) +
  geom_point()+
  facet_wrap(~ variable, scales = "free_x", ncol = 2) +
  geom_smooth() +
  xlab("")
```



From the above plots it is clear that Year does not seem to show any effect on rainfall. However one could believe that climate change would have an effect on rainfall, but this does not seem to be the case for our data. The plot for SOI on the other hand seems to give a clear indication that on average, higher SOI leads to more rainfall.

The fact that Year does not have any effect on rainfall is also seen when a 95% confidence interval for the slope for a simple linear model of rainfall given Year:

```
confint(lm(Rain~Year, data=Rain.data.small))
```

```
##                   2.5 %      97.5 %
## (Intercept) -312.2533951 221.6681166
## Year          -0.1051087   0.1666147
```

We see that Year has 0 included in the interval, hence we cannot conclude any effect from Year on the rainfall.

Analysis using SOI phase

In this analysis the objective is to fit a Tweedie exponential dispersion model to the data, and to predict rainfall as a function of SOI phase. Throughout this problem, we assume $1 < k < 2$.

Fitting a Tweedie GLM model requires the nuisance parameter k *to be known. First, use the mean-variance relationship to estimate* k. *Since* $var(Y) = \psi \mathcal{V}(\mu) = \psi \mu^k$, *then* $\log(var(Y)) = \log(\psi) + k \log(\mu)$.

Compute the empirical mean and variance within each SOI phase, and make a linear regression of $\log(var(Y))$ *on* $\log(mean(Y))$*. Use the output from this linear regression to report some simple estimates of* k *and* ψ*. Make a plot of the data with the estimated regression line on top.*

We compute the empirical mean and variance for each of the five SOI phases using the following formulas:

$$\tilde{\mu} = \frac{\sum x_i}{n}, \qquad \tilde{\sigma^2} = \frac{\sum (x_i - \bar{x})^2}{n-1}$$

The two formulas are implemented with the code below.

```
data <- cbind("mean"=numeric(5),"variance"=numeric(5))

for (i in 1:5){
  sub <- subset(Rain.data.small, Phase==i)
  n <- dim(sub)[1]
  mu <- 1/n*sum(sub$Rain)
  data[i,1] <- mu
  data[i,2] <- 1/(n-1)*sum((sub$Rain-mu)^2)
}
data <- data.frame(data)
data
```

```
##        mean  variance
## 1  2.894118  42.19059
## 2 34.891304 984.55810
## 3  2.344444  22.98028
## 4 17.750000 657.09296
## 5  9.097059 168.71787
```

We now have data of the empirical mean and variance for each SOI phase, which we use to make the linear regression $\log(var(Y)) = \log\psi + k\log\mu$ that we are asked to do:

```
lm1 <- lm(log(variance) ~ log(mean), data = data)

summary(lm1)
```

```
##
## Call:
## lm(formula = log(variance) ~ log(mean), data = data)
##
## Residuals:
##        1        2        3        4        5
##  0.14093 -0.21617 -0.16991  0.33155 -0.08641
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1042     0.2705   7.779  0.00442 **
## log(mean)     1.4087     0.1151  12.240  0.00117 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2664 on 3 degrees of freedom
## Multiple R-squared:  0.9804, Adjusted R-squared:  0.9738
## F-statistic: 149.8 on 1 and 3 DF,  p-value: 0.001174
```

It is given that $\log(var(Y)) = \log(\psi) + k\log(\mu)$, hence the value of ψ is the exponential of the intercept in the linear model, and k is the estimate for log(mean). This gives us the coefficients:

```
k<-summary(lm1)$coefficients[2]
k
```

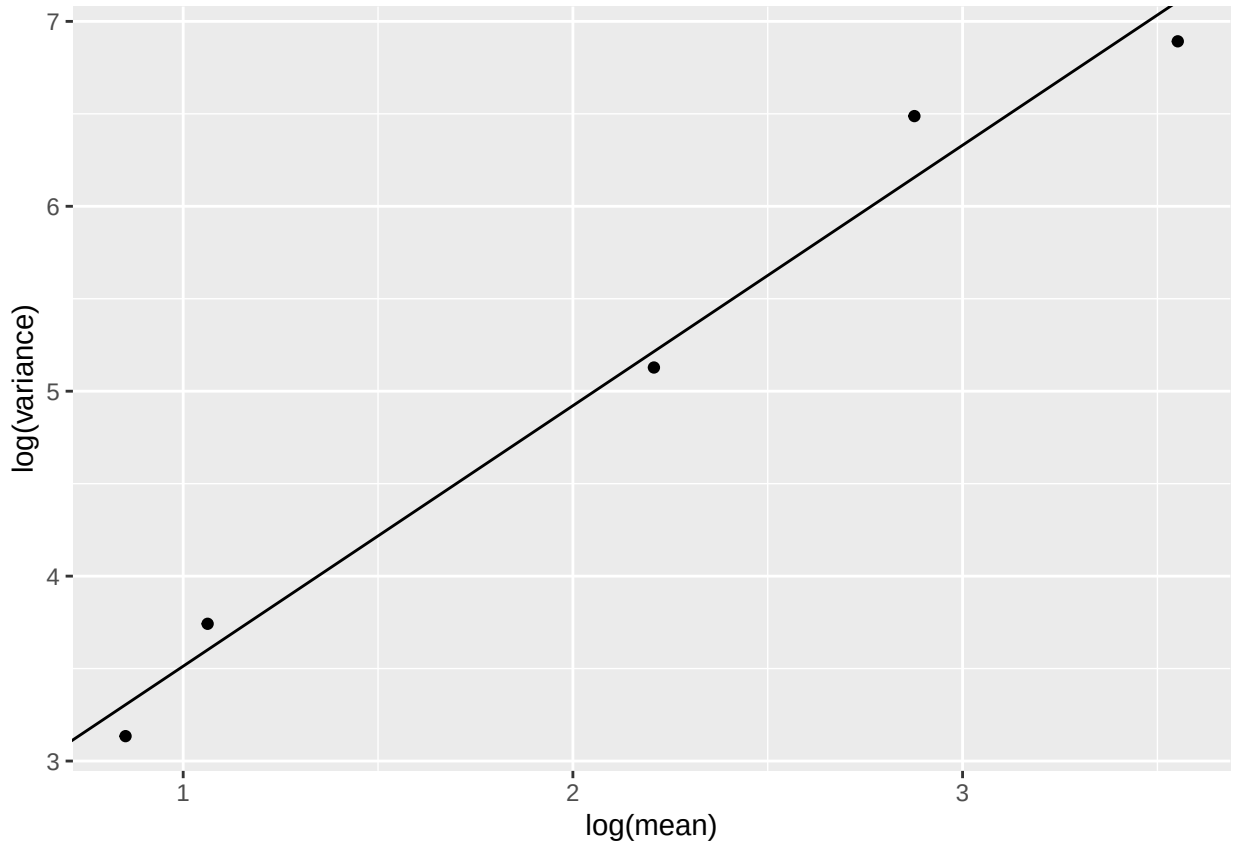
```
## [1] 1.408723
```

```
psi<-exp(summary(lm1)$coefficients[1])
psi
```

```
## [1] 8.200896
```

We are also asked to do a plot of the data with the regression line on top, this is done below:

```
qplot(log(mean), log(variance), data = data)+
  geom_abline(slope = 1.4087, intercept = 2.1042)
```



It seems like the data fits the regression line quite well and the residuals are quite small, however due to only five data points significant conclusions might be hard to draw.

Fit a Tweedie exponential dispersion model using the estimate of k found in this regression (if you did not succeed, then use k = 1.4) with one categorial explanatory variable, the SOI phase (it should enter as a factor). Use the log-link, not the canonical link. The Tweedie GLM can be fitted by loading the package statmod, then use family = tweedie(var.power = 1.4, link.power = 0) (insert the value of k that was estimated

above for var.power), where link.power = 0 provides the log-link function. If link.power is not given, it will use the canonical link function. Comment on the results.

We fit the GLM using the given code, where we insert the value for k which we found earlier to be 1.4087. We remove the intercept of the model, to just get the true value for each SOI phase,

$$\log(\mathbb{E}(\text{Rain}_i \mid \text{Phase}_i)) = \beta_1 1_{\{\text{Phase}_i=1\}} + \beta_2 1_{\{\text{Phase}_i=2\}} + \beta_3 1_{\{\text{Phase}_i=3\}} + \beta_4 1_{\{\text{Phase}_i=4\}} + \beta_5 1_{\{\text{Phase}_i=5\}}.$$

```
glm1 <- glm(Rain ~ Phase-1, family = tweedie(var.power = 1.4087, link.power = 0),
            data = Rain.data.small)

summary(glm1)
```

```
##
## Call:
## glm(formula = Rain ~ Phase - 1, family = tweedie(var.power = 1.4087,
##     link.power = 0), data = Rain.data.small)
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
## Phase1   1.0627     0.5186   2.049   0.0429 *
## Phase2   3.5522     0.2136  16.632  < 2e-16 ***
## Phase3   0.8520     0.7586   1.123   0.2639
## Phase4   2.8764     0.2364  12.168  < 2e-16 ***
## Phase5   2.2080     0.2614   8.447 1.71e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Tweedie family taken to be 8.570993)
##
##     Null deviance: 5231.18  on 111  degrees of freedom
## Residual deviance:  888.75  on 106  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

From the above R output we see that each of the five SOI phases have their own β-coefficient, exactly as we wanted. We see that the coefficients of Phase are quite different for each of the five phases, but this is also what one would expect when recalling the boxplot for each phase which was made in the exploratory analysis. The standard error for phase 1 and phase 3 are quite a lot higher than the value for the rest of them, but this can probably be explained by the lower number of observations for these two groups.

Find the estimated probability that it will not rain in July. Compare to the empirical proportion of zeros in the data.

Compare the estimate of the dispersion parameter with the estimate found in the linear regression

Since all datapoints are from July, the probability that it will not rain in July $P(Y = 0)$ in the Tweedie model is equal to

$$P(Y = 0) = e^{-\lambda^*},$$

where

$$\lambda^* = \frac{\mu^{2-k}}{\psi(2-k)}.$$

We first have to calculate the estimate of ψ for this model. This can be done using the formula (6.5) on page 140 in the book:

$$\hat{\psi} = \frac{1}{n-p}\chi^2,$$

where χ^2 is the Pearson statistic. The value of n is the amount of observations in the data set, and p are the amount of coefficients. We hereby get the following:

```
1/(dim(Rain.data.small)[1]-5)*sum(residuals(glm1, type = "pearson")^2)
```

```
## [1] 8.570977
```

Hence the estimate for ψ is 8.570977 when calculated using the Tweedie GLM with the k-value we found earlier. By estimating μ as the mean of Rain, we can now compute λ^* and thus $P(Y = 0)$.

```
lambdastar1 <- mean(Rain.data.small$Rain)^(2-1.408723)/(8.570977*(2-1.408723))

exp(-lambdastar1)
```

```
## [1] 0.374029
```

We estimate the probability that it will not rain in July to be 37,4%.

We now want to compare this probability to the empirical proportion of zeros in the data, which we compute below:

```
emp_Prop_zero <- nrow(subset(Rain.data.small, Rain == 0))/nrow(Rain.data.small)
emp_Prop_zero
```

```
## [1] 0.3783784
```

We find that in the data 37,84% of the observations had not registered any rain in the month of July. Our estimate of $P(Y = 0)$ is thus very similar to the empirical proportion of zeros in the data.

We now wish to compare the dispersion parameter from above with the one from the linear regression. The estimate for ψ is 8.570977 when calculated using the Tweedie GLM with the k-value from earlier. This value is slightly higher than what was found in the linear model earlier, where we had $\hat{\psi} = 8.2009$.

Now explore another way to estimate k: Estimate k by minimizing the AIC by using a profile likelihood of a model with SOI phase as explanatory variable entering as a factor. To extract the AIC from a Tweedie GLM, load the package tweedie, then use AICtweedie(model). You can either calculate the AIC for a finite number of points between k = 1 and 2, or use optimize to find the k that provides the optimal AIC value, as done on the slides 17-19 from week 3, lecture 2, where the nuisance parameter in the Weibull distribution was estimated using the profile likelihood (note on the slides it is the log-likelihood, here we use the AIC). Plot the AIC values as a function of k.

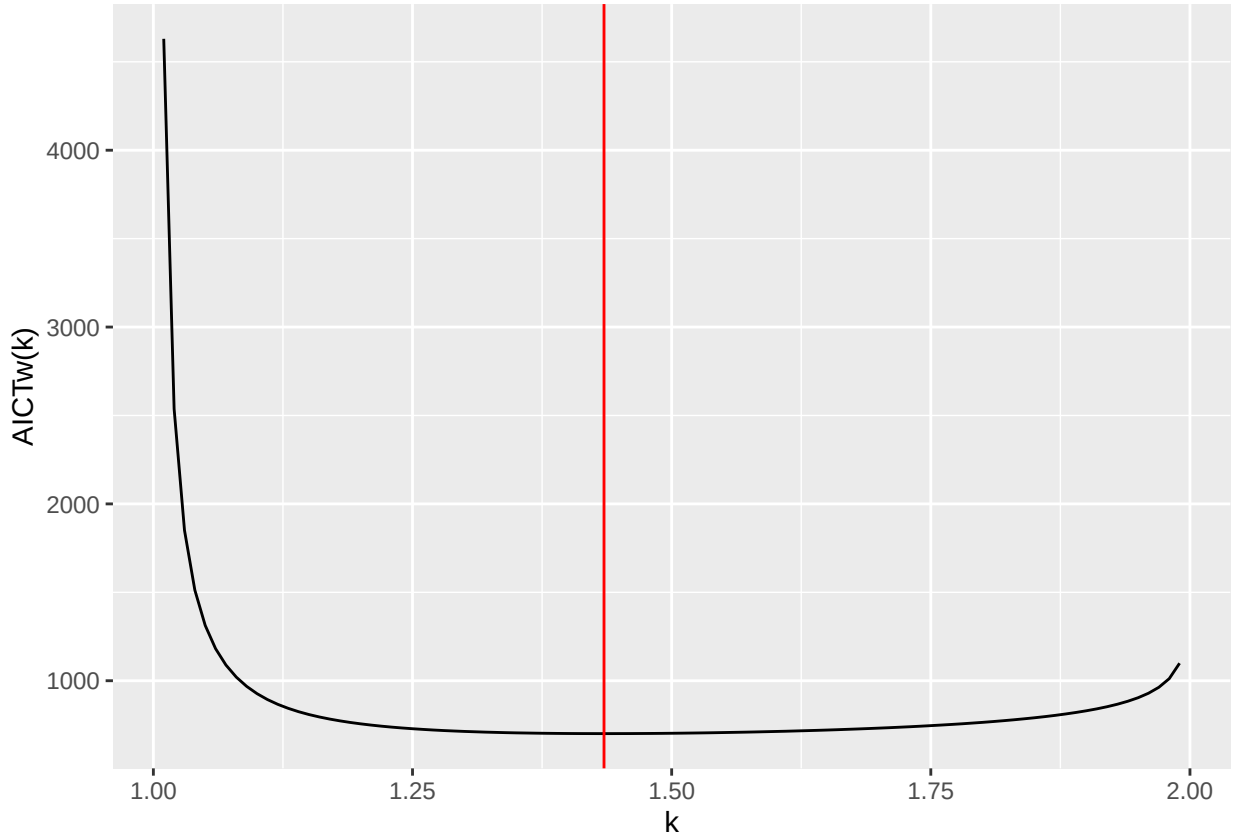
In order to minimize the AIC we use the optimize-function in R in combination with the AICtweedie-function. We let k be a variable in the GLM, and find the k-value which gives us the lowest value for AIC. We set maximum to false in the optimize-function, as the minimum is what we want.

```
AICTw <- Vectorize(function(k)
AICtweedie(glm(Rain ~ Phase, family = tweedie(var.power = k, link.power = 0),
               data=Rain.data.small)))

(koptimal <- optimize(AICTw, c(1, 2), maximum = FALSE))
```

```
## $minimum
## [1] 1.434735
##
## $objective
## [1] 701.0144
```

We hereby find the value to be $k = 1.434735$ in order to minimize AIC. We also make a plot of how AIC changes for different values of k. We limit the lower limit to be just above 1, as a value of 1 returns an infinite AIC-value. We mark the optimal k-value found above with a vertical red line:

```
k <- seq(1.01, 1.99, 0.01); qplot(k, AICTw(k), geom = "line") +
geom_vline(xintercept = koptimal$minimum, color="red")
```



The plot above supports our argument that $k = 1.434735$ gives us the minimal value for AIC.

Repeat the above calculations with the new estimate of k. Discuss the results.

We make a new GLM with the new estimate of k found using the optimize-function:

$$\log(\mathbb{E}(\text{Rain}_i \mid \text{Phase}_i)) = \beta_1 1_{\{\text{Phase}_i=1\}} + \beta_2 1_{\{\text{Phase}_i=2\}} + \beta_3 1_{\{\text{Phase}_i=3\}} + \beta_4 1_{\{\text{Phase}_i=4\}} + \beta_5 1_{\{\text{Phase}_i=5\}}.$$

Note that the coefficients are different in this model than the previous model.

```
glm2<-glm(Rain ~ Phase-1, family = tweedie(var.power = koptimal$minimum, link.power = 0),
          data = Rain.data.small)
summary(glm2)
```

```
##
## Call:
## glm(formula = Rain ~ Phase - 1, family = tweedie(var.power = koptimal$minimum,
##     link.power = 0), data = Rain.data.small)
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
## Phase1   1.0627     0.5099   2.084   0.0395 *
## Phase2   3.5522     0.2169  16.378  < 2e-16 ***
## Phase3   0.8520     0.7437   1.146   0.2545
## Phase4   2.8764     0.2380  12.088  < 2e-16 ***
## Phase5   2.2080     0.2608   8.465 1.56e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Tweedie family taken to be 8.058693)
##
##     Null deviance: 5083.50  on 111  degrees of freedom
## Residual deviance:  853.43  on 106  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

By directly looking at the new GLM, we do not see any significant changes. No changes at all are to be seen for the β-estimate, only a small change in the estimated standard error, where some are higher and others lower. However these small changes might have an impact later on.

We now wish to find the estimated probability that it will not rain in July using the new k-value. We first compute the estimate of the dispersion parameter for the new GLM. This is done with exactly the same method as we used earlier:

```
1/(dim(Rain.data.small)[1]-5)*sum(residuals(glm2, type = "pearson")^2)
```

```
## [1] 8.058688
```

We hereby find that $\hat{\psi}$ for the GLM based on k found with optimize is 8.058688. We can now calculate $P(Y = 0)$ in the following:

```
lambdastar2 <- mean(Rain.data.small$Rain)^(2-koptimal$minimum)/
  (8.058688*(2-koptimal$minimum))
exp(-lambdastar2)
```

```
## [1] 0.3608009
```

Thus with the k-value from the optimal AIC, we estimate the probability that it will not rain to be 36,08%, which is lower than the empirical proportion of zeros.

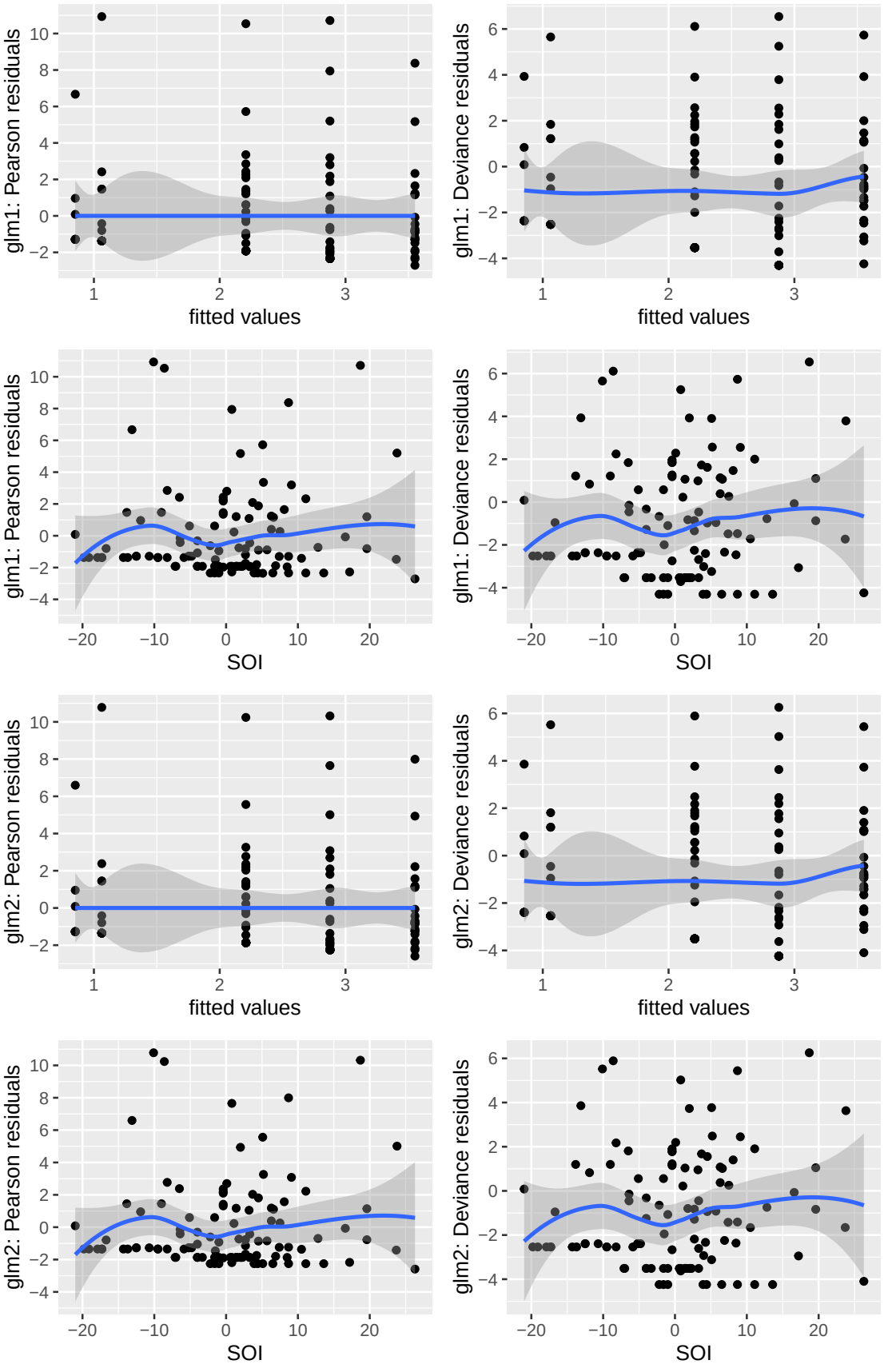
The dispersion parameter $\hat{\psi}$ for the GLM based on k found with optimize is 8.058688, which is quite a lot lower than 8.570977 which we found for the first GLM.

It seems like the new model with lower dispersion parameter is better at fitting the data, as the sum of squared residuals are lower. This might indicate that the latter model is better at predicting rainfall.

Make residual plots for the models. Plot residuals against fitted values and against SOI as well (note: against SOI, not Phase). Discuss the results.

We make plots for both models, where we arrange the residuals against fitted values in one array for both models, and the same for SOI. This is to make it easier to compare the different plots. We both make plots using deviance and Pearson residuals, in order to see results both with and without the GA3 assumption from p. 130 in the book. In the plots "GLM 1" refers to the first GLM, where $k = 1.4087$, and "GLM 2" to the other GLM with $k = 1.4347$:

```
d1<-qplot(fortify(glm1)$.fitted, residuals(glm1, type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("glm1: Deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

d2<-qplot(Rain.data.small$SOI, residuals(glm1, type="deviance"))+
  geom_smooth()+xlab("SOI")+ylab("glm1: Deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

p1<-qplot(fortify(glm1)$.fitted, residuals(glm1, type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("glm1: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

p2<-qplot(Rain.data.small$SOI, residuals(glm1, type="pearson"))+
  geom_smooth()+xlab("SOI")+ylab("glm1: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

d1_<-qplot(fortify(glm2)$.fitted, residuals(glm2, type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("glm2: Deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

d2_<-qplot(Rain.data.small$SOI, residuals(glm2, type="deviance"))+
  geom_smooth()+xlab("SOI")+ylab("glm2: Deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

p1_<-qplot(fortify(glm2)$.fitted, residuals(glm2, type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("glm2: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

p2_<-qplot(Rain.data.small$SOI, residuals(glm2, type="pearson"))+
  geom_smooth()+xlab("SOI")+ylab("glm2: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

grid.arrange(p1,d1,p2,d2,p1_,d1_,p2_,d2_,ncol=2)
```


glm1: Pearson residuals
fitted values
glm1: Deviance residuals
fitted values
glm1: Pearson residuals
SOI
glm1: Deviance residuals
SOI
glm2: Pearson residuals
fitted values
glm2: Deviance residuals
fitted values
glm2: Pearson residuals
SOI
glm2: Deviance residuals
SOI

From the above plots it is very difficult to see any significant difference between the two models. When looking very closely one can see that the confidence intervals are just a tiny bit smaller for model 2, however this is only a few decimal points. A difference is however more clear between Pearson residuals and deviance residuals, which also makes great sense, as they rely on different model assumptions.

On the plot for residuals against fitted values we see that for deviance residuals both the mean and variance assumptions do not seems to be fulfilled. The mean is for both models constantly below 0, which is should not. The variance is also not ideal since the lack of data points between 1 and 2 makes the variance very big. This is also the case for the Pearson residuals. The Pearson residuals however has captured the mean structure well, but the residuals seems to be larger on the other hand.

Regarding the objective of testing if there is an association between rainfall and SOI phase, what is your final conclusion?

In order to conclude whether or not there is an association between rainfall and the SOI phase, we look at the two models one more time:

```
summary(glm1)
```

```
##
## Call:
## glm(formula = Rain ~ Phase - 1, family = tweedie(var.power = 1.4087,
##     link.power = 0), data = Rain.data.small)
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
## Phase1   1.0627     0.5186   2.049   0.0429 *
## Phase2   3.5522     0.2136  16.632  < 2e-16 ***
## Phase3   0.8520     0.7586   1.123   0.2639
## Phase4   2.8764     0.2364  12.168  < 2e-16 ***
## Phase5   2.2080     0.2614   8.447 1.71e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Tweedie family taken to be 8.570993)
##
##     Null deviance: 5231.18  on 111  degrees of freedom
## Residual deviance:  888.75  on 106  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

```
summary(glm2)
```

```
##
## Call:
## glm(formula = Rain ~ Phase - 1, family = tweedie(var.power = koptimal$minimum,
##     link.power = 0), data = Rain.data.small)
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
## Phase1   1.0627     0.5099   2.084   0.0395 *
## Phase2   3.5522     0.2169  16.378  < 2e-16 ***
## Phase3   0.8520     0.7437   1.146   0.2545
```

```
## Phase4   2.8764     0.2380  12.088  < 2e-16 ***
## Phase5   2.2080     0.2608   8.465 1.56e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Tweedie family taken to be 8.058693)
##
##     Null deviance: 5083.50  on 111  degrees of freedom
## Residual deviance:  853.43  on 106  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

We see that for both models the p-values are below 0,05 except for phase 3. This means that all phases except phase 3 do show a statistically significant association between rainfall and SOI phase. The reason for the high p-value at phase 3 is probably due to the very few observations of this phase, which requires the change in rainfall to be significantly higher in order to draw any conclusions.

In order to verify that removing SOI phase as a variable, leaving only the intercept, will make the model worse, we use the drop1-command. This command makes us able to compare the intercept-only model with the two GLM's used above:

```
drop1(glm1, test="F")
```

```
## Single term deletions
##
## Model:
## Rain ~ Phase - 1
##        Df Deviance F value    Pr(>F)
## <none>       888.8
## Phase   5   5231.2  103.58 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(glm2, test="F")
```

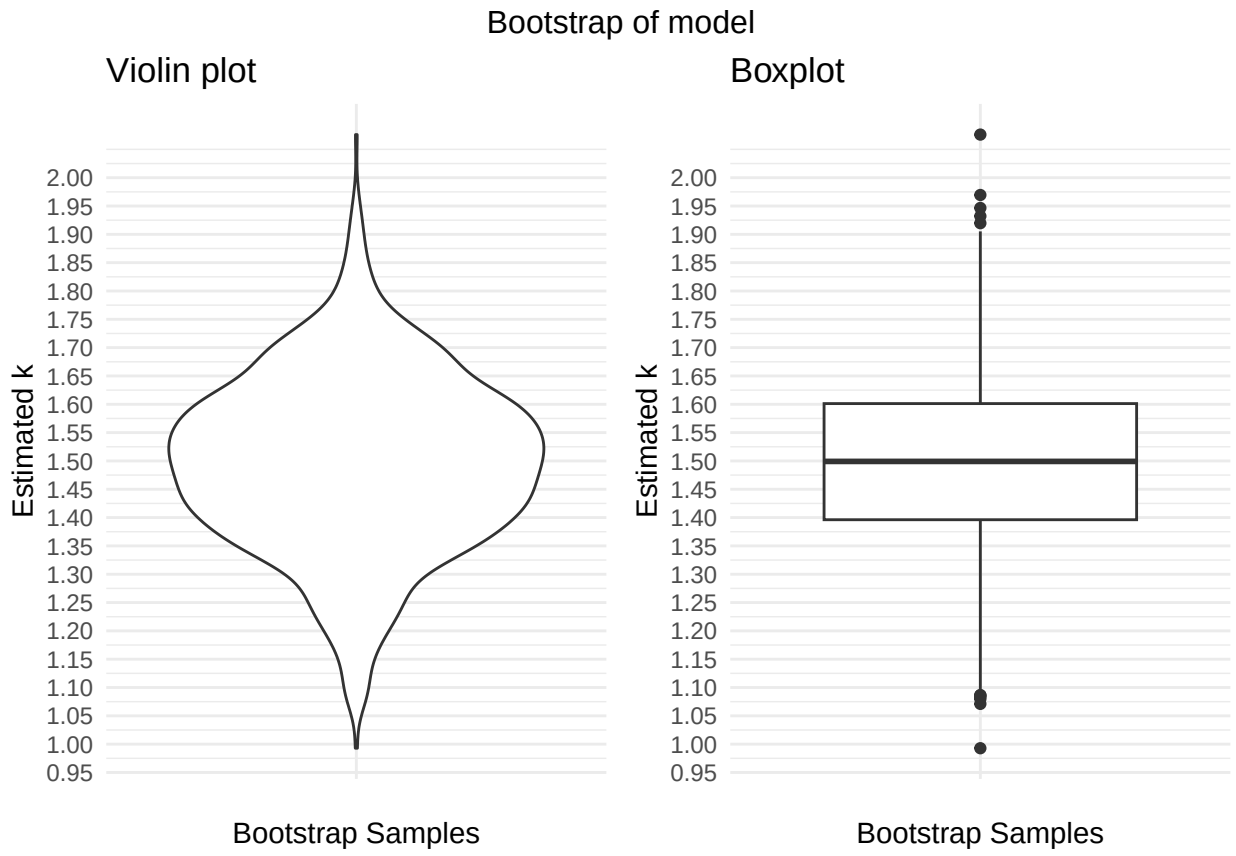
```
## Single term deletions
##
## Model:
## Rain ~ Phase - 1
##        Df Deviance F value    Pr(>F)
## <none>       853.4
## Phase   5   5083.5  105.08 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It is clear from above that removing Phase will make the model significantly worse at predicting the amount of rainfall. We thereby conclude that there is an association between SOI phase and the amount of rainfall observed in the Australian town.

Finally, investigate how well the Tweedie index k is estimated. Simulate 1000 bootstrap samples from the estimated model. On each simulated data sample, estimate k. You decide which method to use, or if you want to compare different methods to estimate k. Make a boxplot or a violinplot of the estimates of k. Discuss the results.

In the following code we do parametric bootstrapping from the first Tweedie GLM (glm1) i.e. the one where k is estimated through linear regression with the variance and mean. This is done exactly as described on page 226 in the book. For each sample, we estimate k the same way. Notice we have to remove mean and variances that are equal to 0 as log(0) makes no sense. We get the following plot:

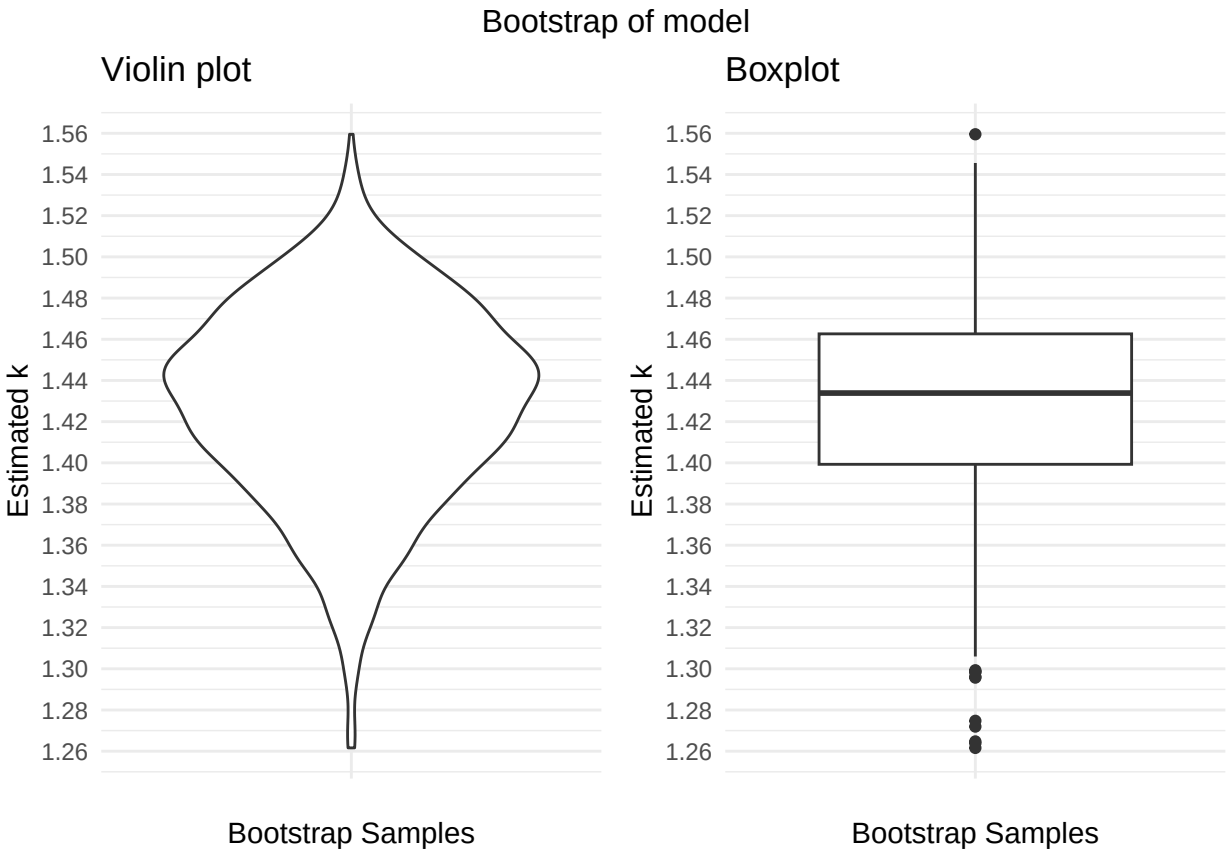
```
set.seed(2024)
B <- 1000
k_est1 <- numeric(B)
Rain.data.samp <- Rain.data.small
Rain_predict <- fitted(glm1)

for (b in 1:B){
  Rain.data.samp$Rain <- rTweedie(Rain_predict, p = 1.408723, phi = 8.570977)

  bootdata <- cbind("mean"=numeric(5),"variance"=numeric(5))

  for (i in 1:5){
    bootsub <- subset(Rain.data.samp, Phase==i)
    bootdata[i,1] <- mean(bootsub$Rain)
    bootdata[i,2] <- var(bootsub$Rain)
  }
  bootdata <- subset(data.frame(bootdata), mean != 0, variance != 0)

  bootlm <- lm(log(variance) ~ log(mean), data = bootdata)

  k_est1[b] <- unname(coef(bootlm)[2])
}

plot1<-ggplot(data.frame(k_est1), aes(x = "", y = k_est1)) +
  geom_violin() +
  labs(title = "Violin plot",
       x = "Bootstrap Samples", y = "Estimated k") +
  theme_minimal()+scale_y_continuous(breaks = seq(0, 2, by = 0.05))

plot2<-ggplot(data.frame(k_est1), aes(x = "", y = k_est1)) +
  geom_boxplot() +
  labs(title = "Boxplot",
       x = "Bootstrap Samples", y = "Estimated k") +
  theme_minimal()+scale_y_continuous(breaks = seq(0, 2, by = 0.05))

grid.arrange(plot1,plot2,ncol=2, top="Bootstrap of model")
```



We see the median is around 1,5 and interquantile range is approximately 1,4 to 1,6. We may also do parametric bootstrap for the second Tweedie GLM model (the one we called glm2) i.e. the tweedie model where k is estimated by optimizing AIC. In each sample, we estimate k by optimizing AIC. We get the following:

```
set.seed(2024)
B <- 1000
k_est2 <- numeric(B)
Rain.data.samp <- Rain.data.small
Rain_predict <- fitted(glm2)

bootAICglm <- Vectorize(function(k)
  AICtweedie(glm(Rain ~ Phase-1, family = tweedie(var.power = k, link.power = 0),
                 data=Rain.data.samp))
)

for (b in 1:B){
  Rain.data.samp$Rain <- rTweedie(Rain_predict, p = koptimal$minimum, phi = 8.058688)

  optimum <- suppressWarnings(optimize(bootAICglm, c(1, 2), maximum = FALSE))

  k_est2[b] <- optimum$minimum
}

plot1<-ggplot(data.frame(k_est2), aes(x = "", y = k_est2)) +
  geom_violin() +
```

```
  labs(title = "Violin plot",
       x = "Bootstrap Samples", y = "Estimated k") +
  theme_minimal()+scale_y_continuous(breaks = seq(0, 1.6, by = 0.02))

plot2<-ggplot(data.frame(k_est2), aes(x = "", y = k_est2)) +
  geom_boxplot() +
  labs(title = "Boxplot",
       x = "Bootstrap Samples", y = "Estimated k") +
  theme_minimal()+scale_y_continuous(breaks = seq(0, 1.6, by = 0.02))

grid.arrange(plot1,plot2,ncol=2, top="Bootstrap of model")
```

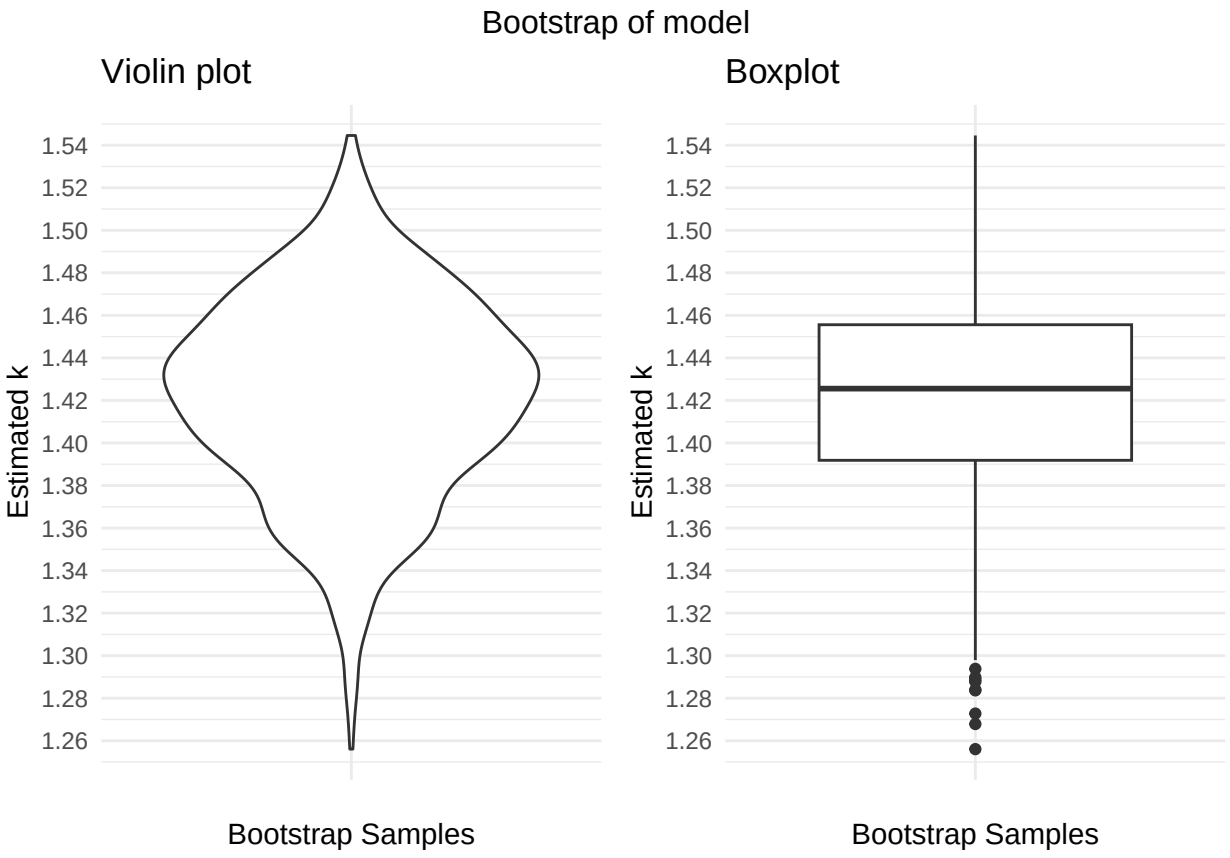


We see that the median is around 1,43 with the first quantile being around 1,395 and third quantile being around 1,46. Both of our estimated k's from before are in both of the interquantile ranges. But the two bootstrap results differ greatly, as the median and third quantile is way higher making the interquantile range far larger than for the second bootstrap. This is likely due to the fact that because we have to remove the data points where mean and variance are equal to 0, the estimated k's from each bootstrap sample will vary a lot more.

We can also implement a code to do bootstrap samples from the model by making samples of the dataset where we draw as many data points out as the data set has, but with replacement. This makes it possible to draw the same observation multiple times, but also for some observations not to be drawn. This is exactly the principle of nonparametric bootstrapping. For each new data set we use the optimize function to find the value of k that minimizes the AIC-value, exactly what we did earlier. All the different optimal values for k are saved in a vector. This makes us able to do with a box plot and a violin plot.

Bootstrap to estimate k:

```r
set.seed(2024)
B<-1000
k_est<-numeric(B)

for (b in 1:B){
  boot_sample <- Rain.data.small[sample(1:nrow(Rain.data.small), replace = TRUE), ]
  koptimal <- optimize(function(k) {
  AICtweedie(glm(Rain ~ Phase, family = tweedie(var.power = k, link.power = 0),
                 data=boot_sample)) }, interval=c(1, 2), maximum = FALSE)
  k_est[b] <- koptimal$minimum
}

plot1<-ggplot(data.frame(k_est), aes(x = "", y = k_est)) +
  geom_violin() +
  labs(title = "Violin plot",
       x = "Bootstrap Samples", y = "Estimated k") +
  theme_minimal()+scale_y_continuous(breaks = seq(0, 1.6, by = 0.02))

plot2<-ggplot(data.frame(k_est), aes(x = "", y = k_est)) +
  geom_boxplot() +
  labs(title = "Boxplot",
       x = "Bootstrap Samples", y = "Estimated k") +
  theme_minimal()+scale_y_continuous(breaks = seq(0, 1.6, by = 0.02))

grid.arrange(plot1,plot2,ncol=2, top="Bootstrap of model")
```



We notice the results from the nonparametric bootstrap are very similar to the ones for the second parametric bootstrap. From the plots above we see how the optimal value of k has first and third quantile around 1.39 and 1.46. The values we found for k earlier in the assignment are all in this interval, the Interquantile Range. The median of the optimal k values is around 1,425, which is quite close to the k-value of 1,434 we found earlier.

We also see some quite large outliers, where k is estimated to be around 1,28 and maximum values of the optimal k around 1,54. These datapoint are however only there because of the randomness in the bootstrapping. As each sample is randomly chosen from data, it sometimes results in extreme values. By making even more than the 1000 bootstrap samples we did above, we might get a more accurate approximation of k.

Analysis using SOI directly

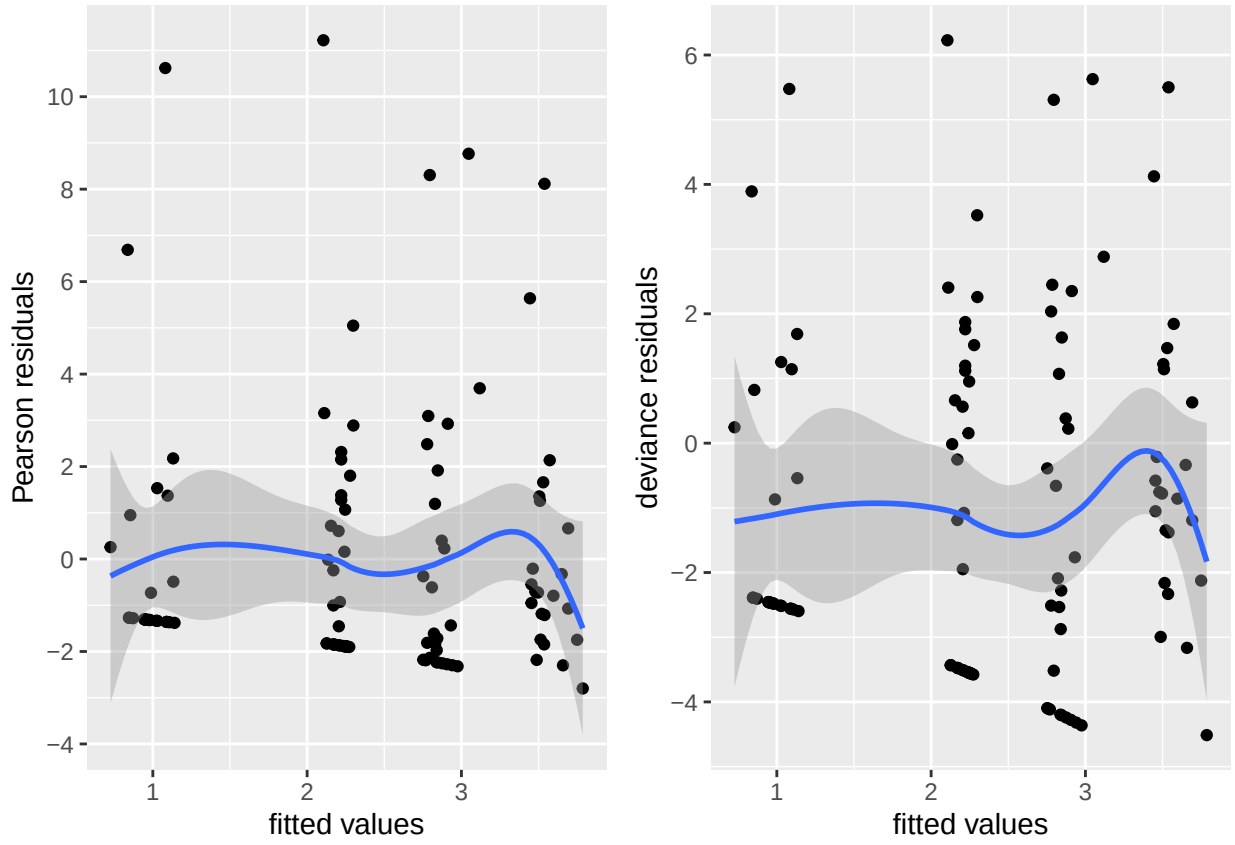
We now want to model the rainfall as a function of the variable SOI. This can be done in multiple different ways with different GLMs. We saw previously from the Spearman correlation matrix that SOI only has a very weak positive correlation with Phase. There is thus not a clear reason to not include both variables in our model for now. We ignore the variable month again since it is constantly 7, and Year is not helpful since it basically just enumerates the data. We will first try to fit a Tweedie GLM and estimate k by optimizing the AIC in the interval $(1, 2)$.

$$\log(\mathbb{E}(\text{Rain}_i \mid \text{Phase}_i, X_{i,\text{SOI}})) = \beta_0 + \beta_{SOI} X_{i,\text{SOI}} + \beta_2 1_{\{\text{Phase}_i=2\}} + \beta_3 1_{\{\text{Phase}_i=3\}} + \beta_4 1_{\{\text{Phase}_i=4\}} + \beta_5 1_{\{\text{Phase}_i=5\}}.$$

```
form1 <- Rain ~ SOI + Phase
AICglm1 <- Vectorize(function(k)
  AICtweedie(glm(form1, family = tweedie(var.power = k, link.power = 0),
                 data=Rain.data.small))
)
SOIglm1 <- glm(form1, family = tweedie(var.power = optimize(AICglm1, c(1, 2),
  maximum = FALSE)$minimum, link.power = 0), data=Rain.data.small)
summary(SOIglm1)
```

```
##
## Call:
## glm(formula = form1, family = tweedie(var.power = optimize(AICglm1,
##     c(1, 2), maximum = FALSE)$minimum, link.power = 0), data = Rain.data.small)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.22307    0.58124   2.104  0.03774 *
## SOI          0.01408    0.02198   0.641  0.52320
## Phase2       2.19320    0.75223   2.916  0.00434 **
## Phase3      -0.20145    0.90114  -0.224  0.82354
## Phase4       1.56068    0.68831   2.267  0.02541 *
## Phase5       1.00359    0.62773   1.599  0.11288
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Tweedie family taken to be 8.069734)
##
##     Null deviance: 1162.27  on 110  degrees of freedom
## Residual deviance:  850.48  on 105  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 6
```

```
plot1 <- qplot(fortify(SOIglm1)$.fitted, residuals(SOIglm1, type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),labels = seq(-10, 15, by = 2))
plot2 <- qplot(fortify(SOIglm1)$.fitted, residuals(SOIglm1, type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),labels = seq(-10, 15, by = 2))

grid.arrange(plot1,plot2,ncol=2)
```



To determine whether Phase should be included in the model, we do a confidence interval based on profile log-likelihood using the confint-command:

```
confint(SOIglm1)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %     97.5 %
## (Intercept)  0.10848734 2.31024948
## SOI         -0.02725825 0.05517398
## Phase2       0.80807436 3.58894894
## Phase3      -1.97126525 1.52459849
## Phase4       0.24014901 2.89592435
## Phase5      -0.17961849 2.20927851
```

From the above interval, we see that for Phase=2, Phase=4 and the intercept the value 0 is not included. This means that these phases does seem to have an effect on the rainfall, it is therefore in our interest to keep the variable Phase in the model.

We notice that there are multiple problems with the residual plots, particularly for the deviance residuals. We wish to investigate this further by simulating from the model 4 times and create deviance residual plots for each dataset. We estimate k and ψ to be the following:

```
optimize(AICglm1, c(1, 2), maximum = FALSE)$minimum
```

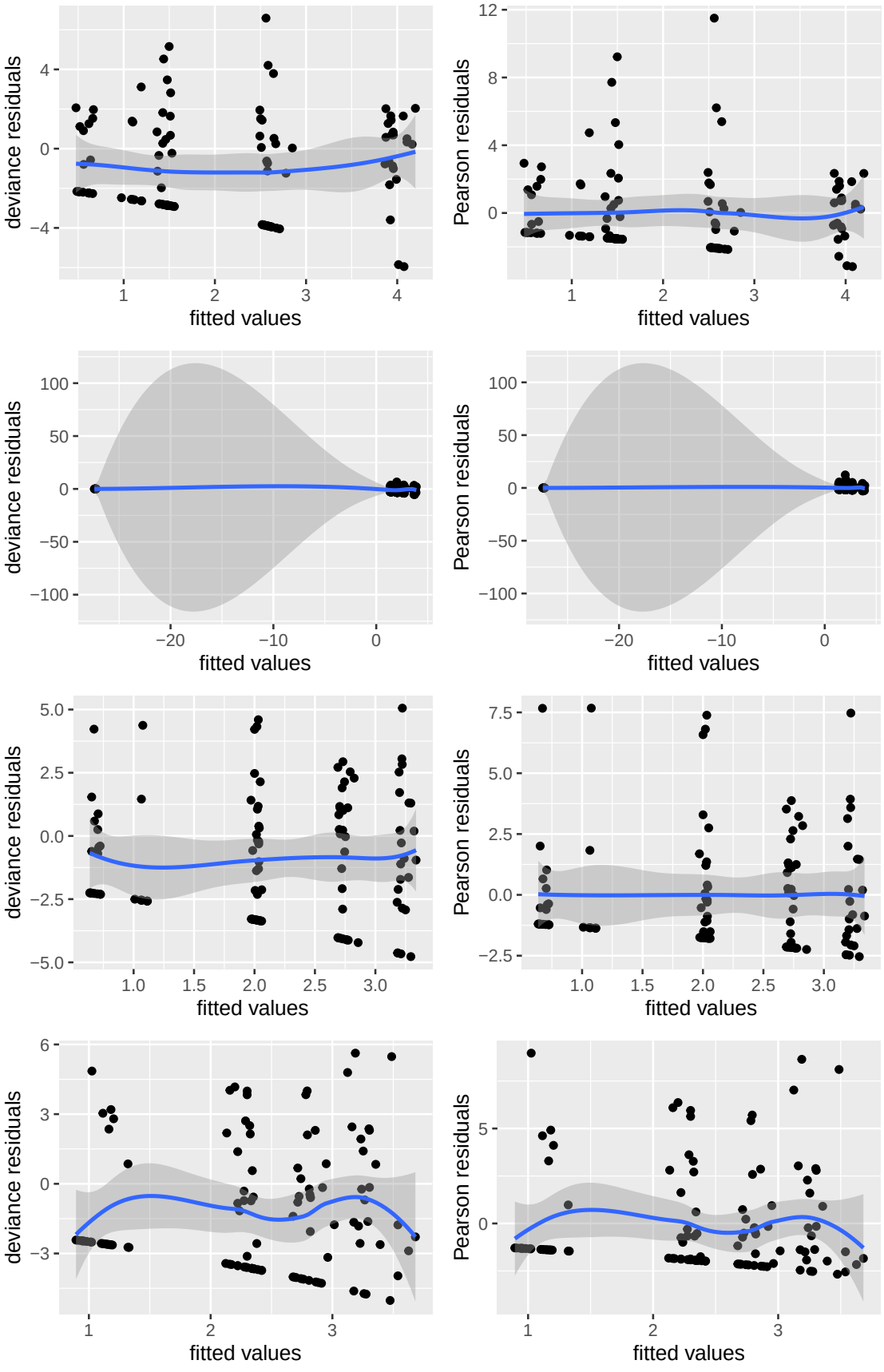
```
## [1] 1.434232
```

```
1/(dim(Rain.data.small)[1]-5)*sum(residuals(SOIglm1, type = "pearson")^2)
```

```
## [1] 7.993578
```

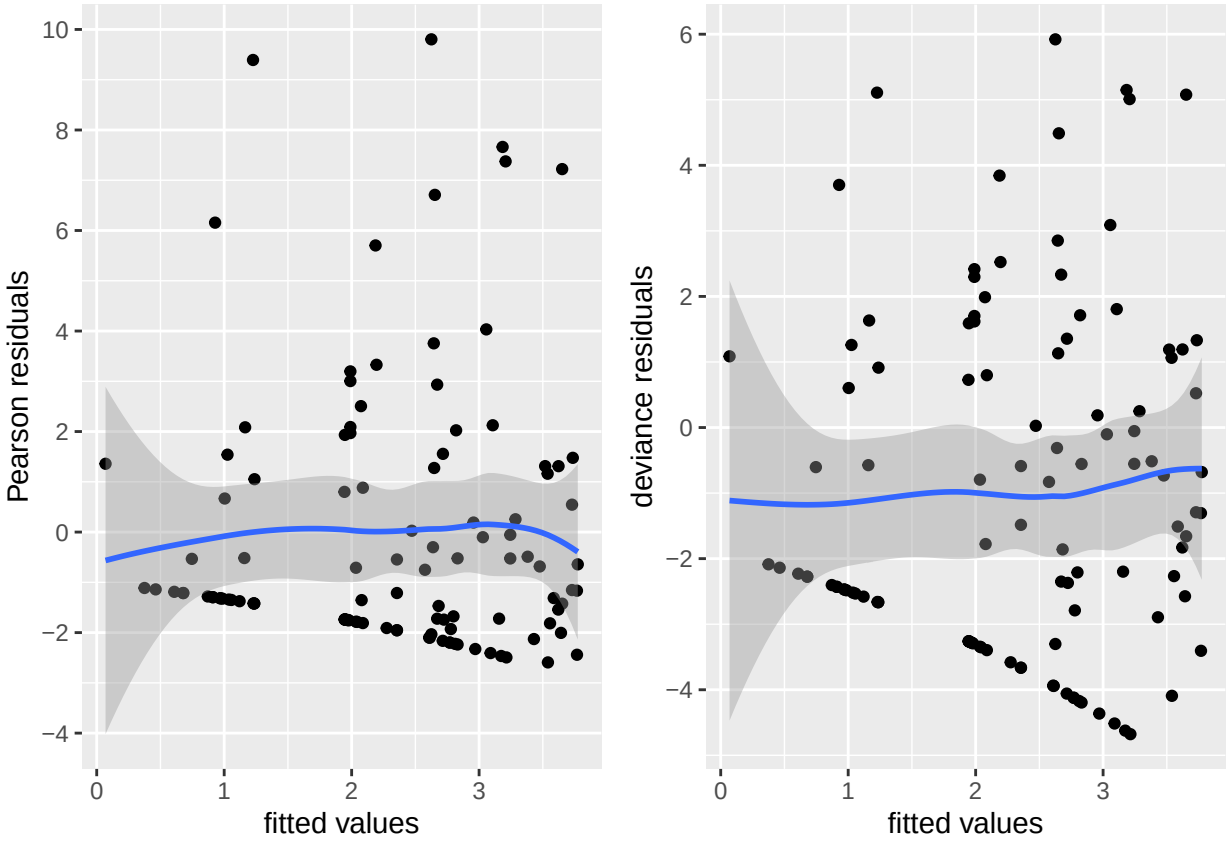
We then get the following plots:

```
set.seed(2024)
yNew1 <- rTweedie(fitted(SOIglm1), p = 1.434232, phi = 7.993578)
yNew2 <- rTweedie(fitted(SOIglm1), p = 1.434232, phi = 7.993578)
yNew3 <- rTweedie(fitted(SOIglm1), p = 1.434232, phi = 7.993578)
yNew4 <- rTweedie(fitted(SOIglm1), p = 1.434232, phi = 7.993578)

simGlmNew1 <- glm(yNew1 ~ Rain.data.small$SOI + Rain.data.small$Phase,
                  family = tweedie(var.power = 1.434232, link.power = 0))
simGlmNew2 <- glm(yNew2 ~ Rain.data.small$SOI + Rain.data.small$Phase,
                  family = tweedie(var.power = 1.434232, link.power = 0))
simGlmNew3 <- glm(yNew3 ~ Rain.data.small$SOI + Rain.data.small$Phase,
                  family = tweedie(var.power = 1.434232, link.power = 0))
simGlmNew4 <- glm(yNew4 ~ Rain.data.small$SOI + Rain.data.small$Phase,
                  family = tweedie(var.power = 1.434232, link.power = 0))
p1 <- qplot(fortify(simGlmNew1)$.fitted, residuals(simGlmNew1, type = "deviance")) +
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")
p2 <- qplot(fortify(simGlmNew2)$.fitted, residuals(simGlmNew2, type = "deviance")) +
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")
p3 <- qplot(fortify(simGlmNew3)$.fitted, residuals(simGlmNew3, type = "deviance")) +
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")
p4 <- qplot(fortify(simGlmNew4)$.fitted, residuals(simGlmNew4, type = "deviance")) +
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")
p1_ <- qplot(fortify(simGlmNew1)$.fitted, residuals(simGlmNew1, type = "pearson")) +
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")
p2_ <- qplot(fortify(simGlmNew2)$.fitted, residuals(simGlmNew2, type = "pearson")) +
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")
p3_ <- qplot(fortify(simGlmNew3)$.fitted, residuals(simGlmNew3, type = "pearson")) +
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")
p4_ <- qplot(fortify(simGlmNew4)$.fitted, residuals(simGlmNew4, type = "pearson")) +
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")

grid.arrange(p1,p1_,p2,p2_,p3,p3_,p4,p4_, ncol=2)
```


deviance residuals
fitted values
Pearson residuals
fitted values
deviance residuals
fitted values
Pearson residuals
fitted values
deviance residuals
fitted values
Pearson residuals
fitted values
deviance residuals
fitted values
Pearson residuals
fitted values

From the data set and the simulated data sets, we see that assuming data follows the model, it is quite typical that there is overdispersion where multiple residuals are close to 6 and -6. It can also be observed with the geom_smooth function that the deviance residuals are around -1 (it is unfortunately a bit harder to see on the second plot because of that one extremely low fitted value) which indicates that our model consistently underestimates the observed values. The reason why the deviance residuals are constantly below 0 is probably because of the assumption GA3 which might not be fulfilled. The Pearson residuals do not depend on this assumptions, only GA1 and GA2, which explains some of their differences, and why the problem with the negative mean structure does not seem to occur for the Pearson residuals. The residual plots also hint that perhaps the mean-variance structure does not hold with the Tweedie exponential dispersion model since they form a slightly trumpet-like shape. We can try to to see if these problems can be handled by adding non-linear effects through natural cubic spline basis expansion on SOI. We will initially try with 4 degrees of freedom.

```
form2 <- Rain ~ ns(SOI, df = 4) + Phase

AICglm2 <- Vectorize(function(k)
  AICtweedie(glm(form2, family = tweedie(var.power = k, link.power = 0),
                 data=Rain.data.small))
)

SOIglm2 <- glm(form2, family = tweedie(var.power = optimize(AICglm2, c(1, 2),
  maximum = FALSE)$minimum, link.power = 0), data=Rain.data.small)

plot1 <- qplot(fortify(SOIglm2)$.fitted, residuals(SOIglm2, type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),labels = seq(-10, 15, by = 2))

plot2 <- qplot(fortify(SOIglm2)$.fitted, residuals(SOIglm2, type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),labels = seq(-10, 15, by = 2))

grid.arrange(plot1,plot2,ncol=2)
```



We see that even with the non-linear effects, both the deviance and Pearson residuals are still quite large and the points still form a slight trumpet shape. According to the geom_smooth function the deviance residuals still tend to be around -1.

We now do cross-validation on the model with different degrees of freedom in the natural spline where the generalization error is computed with the squared deviance residuals. We use jackknifing to increase stability. Note that we have to find the AIC-minimizing k value before sampling in order to fit exactly the same Tweedie model on each sampled data.

```
set.seed(2024)
testCV_ns <- function(data, B = 1, k = 10, df) {

  form <- Rain ~ ns(SOI, df = df)+Phase
  n <- nrow(data)
  PEcv <- vector("list", B)
  tmp <- numeric(n)

  CVAIC <- Vectorize(function(k)
  AICtweedie(glm(form, family = tweedie(var.power = k, link.power = 0),
                 data=Rain.data.small))
  )

  p <- optimize(CVAIC, c(1, 2), maximum = FALSE)$minimum

  for (b in 1:B) {
    ## Generating the random division into groups
    group <- sample(rep(1:k, length.out = n))
```

```
    for (i in 1:k) {
      modelcv <- glm(form, family = tweedie(var.power = p, link.power = 0),
                     data = data[group != i, ])
      muhat <- predict(modelcv, newdata = data[group == i, ], type = "response")

      observed <- data[group == i,]

      ## Calculate the residuals
      Y <- observed$Rain
      ##resids <- (data$Rain[group == i] - muhat)^2
      resids <- 2/(1-p)*(Y^(2-p)-Y*muhat^(1-p))+2/(2-p)*(muhat^(2-p)-Y^(2-p))

      tmp[group == i] <- resids
    }

    PEcv[[b]] <- tmp
  }

  mean(unlist(PEcv))
}

sapply(c(1,2,3,4,8,10,15),function(x) testCV_ns(Rain.data.small, k=nrow(Rain.data.small),
                                                df=x))
```

```
## [1]  8.895055  9.143762  9.305930  9.316223 10.967791 13.545258 14.044843
```

We see that by applying natural splines, we may actually overfit and lose predictive strength in the model.

The Tweedie model above can be used to predict rainfall for any given year when SOI and Phase are known, and Rain not known. However if one knows that in the given year it has already rained in the month of July, but not how much, another model can be used. The gamma-model will for this situation be possible to use. This is because the gamma-distribution requires strictly positive numbers, and with Rain=0 this is not fulfilled. To use the gamma-model we therefore have to make a subset of the data, with only the data where Rain is not 0:

```
Rain.data.wet<-subset(Rain.data.small, Rain>0)
```

We can now fit a gamma-model with the log-link function. We decide to use the log-link such that it is the same as for the Tweedie model. We define the additive GLM based on both SOI and Phase.

$$\log(\mathbb{E}(\text{Rain}_i \mid \text{Phase}_i, X_{i,\text{SOI}})) = \beta_0 + \beta_{SOI} X_{i,\text{SOI}} + \beta_2 1_{\{\text{Phase}_i=2\}} + \beta_3 1_{\{\text{Phase}_i=3\}} + \beta_4 1_{\{\text{Phase}_i=4\}} + \beta_5 1_{\{\text{Phase}_i=5\}}.$$

```
glm_SOIgamma1<-glm(Rain ~ SOI+Phase, family = Gamma(link="log"),
                   data = Rain.data.wet)
```

We now plot the Pearson and deviance residuals for the model to see how well the structure is captured by the model:

```
glm_SOIgamma1d<-qplot(fortify(glm_SOIgamma1)$.fitted, residuals(glm_SOIgamma1,
                                                                type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
```

```
                    labels = seq(-10, 15, by = 2))

glm_SOIgamma1p<-qplot(fortify(glm_SOIgamma1)$.fitted, residuals(glm_SOIgamma1,
                                                                type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

grid.arrange(glm_SOIgamma1d,glm_SOIgamma1p,ncol=2,
             top="Residuals against fitted for deviance and Pearson residuals")
```

The residuals look decently well for the data where the residuals are around 0 and most are between -2 and 2 (and those that are not do not lie too far away). We see that for the deviance residuals the mean-structure might be slightly below 0, which is not the case for the Pearson residuals. As Pearson - cf. p. 144 in the book - are based on weaker assumptions than deviance residuals, we might use too strong assumptions about the model, which leads to a slightly negative mean structure.

However we would also like to find out if it is worthwile to include Phase in the model compared to just having Rain given SOI. We therefore do the same plots as above for a smaller gamma-model:

$$\log(\mathbb{E}(\text{Rain}_i \mid X_{i,\text{SOI}})) = \beta_0 + \beta_{SOI} X_{i,\text{SOI}}$$

```
glm_SOIgamma2<-glm(Rain ~ SOI, family = Gamma(link="log"),
                   data = Rain.data.wet)
```

```
glm_SOIgamma2d<-qplot(fortify(glm_SOIgamma2)$.fitted, residuals(glm_SOIgamma2,
                                                                type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma2p<-qplot(fortify(glm_SOIgamma2)$.fitted, residuals(glm_SOIgamma2,
                                                                type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

grid.arrange(glm_SOIgamma2d,glm_SOIgamma2p,ncol=2,
             top="Residuals against fitted for deviance and Pearson residuals")
```

The residual plots above look quite similar to the residual plots for the model with Phase included. We now investigate the confidence intervals based on parametric bootstrap for the first gamma model (the additive model with SOI and Phase). In doing so we assume GA3 which seems like a plausible assumption based on the residual plots from before. The following code is inspired by Exercise 10.3.

```
set.seed(2024)
B <- 1000
bootModels <- vector("list", B)
boot_sample <- Rain.data.wet

for (b in 1:B){
```

```
  boot_sample$Rain <- simulate(glm_SOIgamma1)[,1]
  bootModels[[b]] <- glm(Rain ~ SOI + Phase, family = Gamma(link="log"), data = boot_sample)
}

bootPred <- sapply(bootModels, function(m) coef(m))

bootse <- apply(bootPred, 1, sd)

cbind(lower = coef(glm_SOIgamma1)-1.96*bootse, upper = coef(glm_SOIgamma1)+1.96*bootse)
```

```
##                    lower      upper
## (Intercept)  1.44215319 3.05954388
## SOI         -0.02174647 0.05142353
## Phase2       0.08919084 2.25842453
## Phase3      -1.37311406 1.18402176
## Phase4      -0.12250140 1.91066386
## Phase5      -0.27867237 1.51777551
```

Since bootstrap confidence intervals can be unstable due to the random sampling, we also compute confidence intervals based on the profile log-likelihood for the model to decide if Phase is worth to include:

```
confint(glm_SOIgamma1)
```

```
## Waiting for profiling to be done...
```

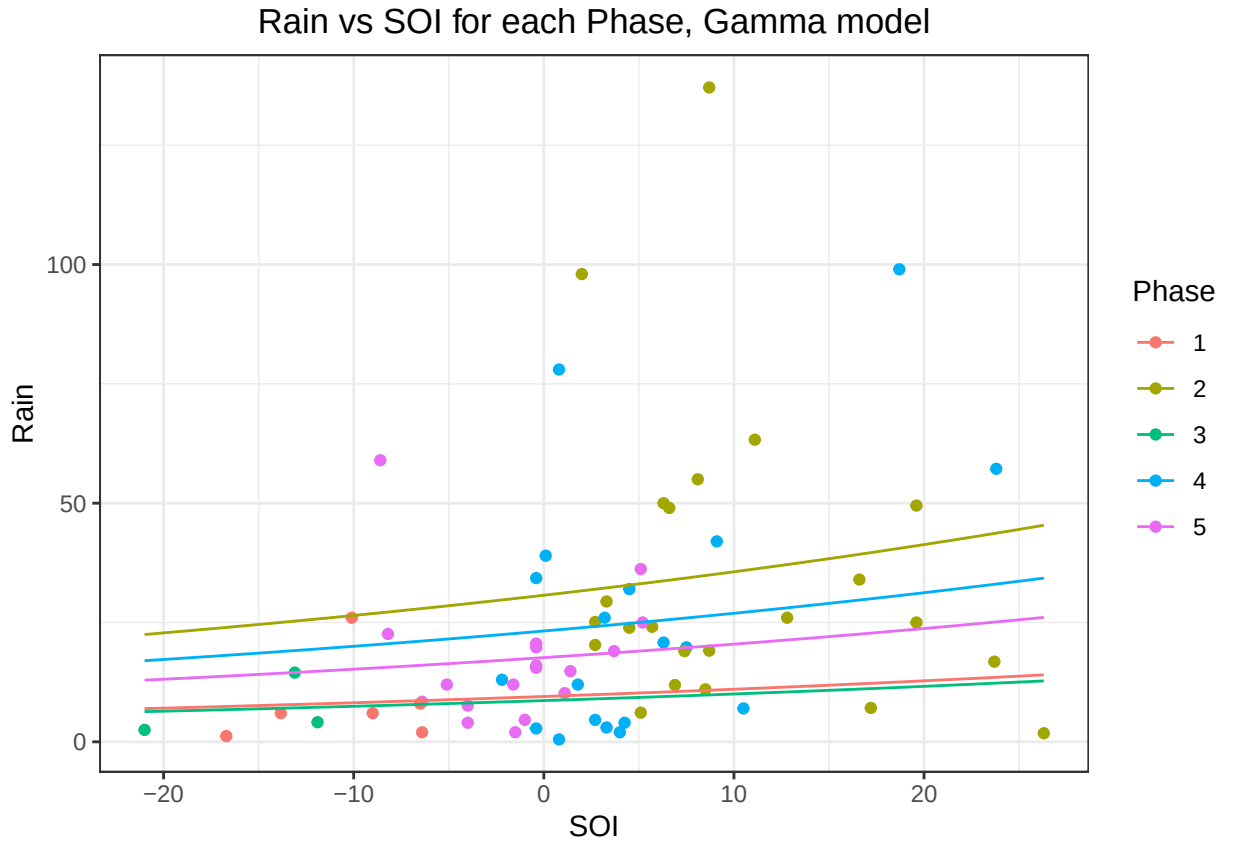
```
##                    2.5 %     97.5 %
## (Intercept)  1.47997531 3.18434586
## SOI         -0.02123390 0.05310184
## Phase2       0.03183226 2.22398163
## Phase3      -1.36277202 1.37032966
## Phase4      -0.23374398 1.95267724
## Phase5      -0.37380382 1.49693604
```

We see that the confidence intervals produced by the two methods look very alike. Even though most of the phases have negative values in the lower limit, the confidence intervals for Phase=2 and the intercept do not include 0 which indicates that it is reasonable to include Phase. The confidence intervals for the other phases are also somewhat skewed, which also support the argument of saving Phase in the model.

We try to plot the GLM for each of the phases to show its effect:

```
q <- ggplot(Rain.data.wet, aes(x = SOI, y = Rain, color = as.factor(Phase))) +
  geom_point() +
  ggtitle("Rain vs SOI by Phase") +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(color = "Phase")

SOI_range <- seq(min(Rain.data.wet$SOI), max(Rain.data.wet$SOI), length.out = 100)
new_data <- expand.grid(SOI = SOI_range, Phase = unique(Rain.data.wet$Phase))
new_data$fitted <- predict(glm_SOIgamma1, newdata = new_data, type = "response")

fit <- data.frame(SOI = Rain.data.wet$SOI, fitted = fitted(glm_SOIgamma1),
```

```
                  Phase = Rain.data.wet$Phase)
q + geom_line(data = new_data, aes(x = SOI, y = fitted, color = as.factor(Phase))) +
  ggtitle("Rain vs SOI for each Phase, Gamma model")
```



It is quite clear from the above plot that for phase 2 and 4 we see a significantly higher expected rainfall than the other groups. Phases 1 and 3 on the other hand seems to predict a much lower rainfall for the same SOI-value. The plot above also supports our argument of saving Phase as a covariate in the GLM.

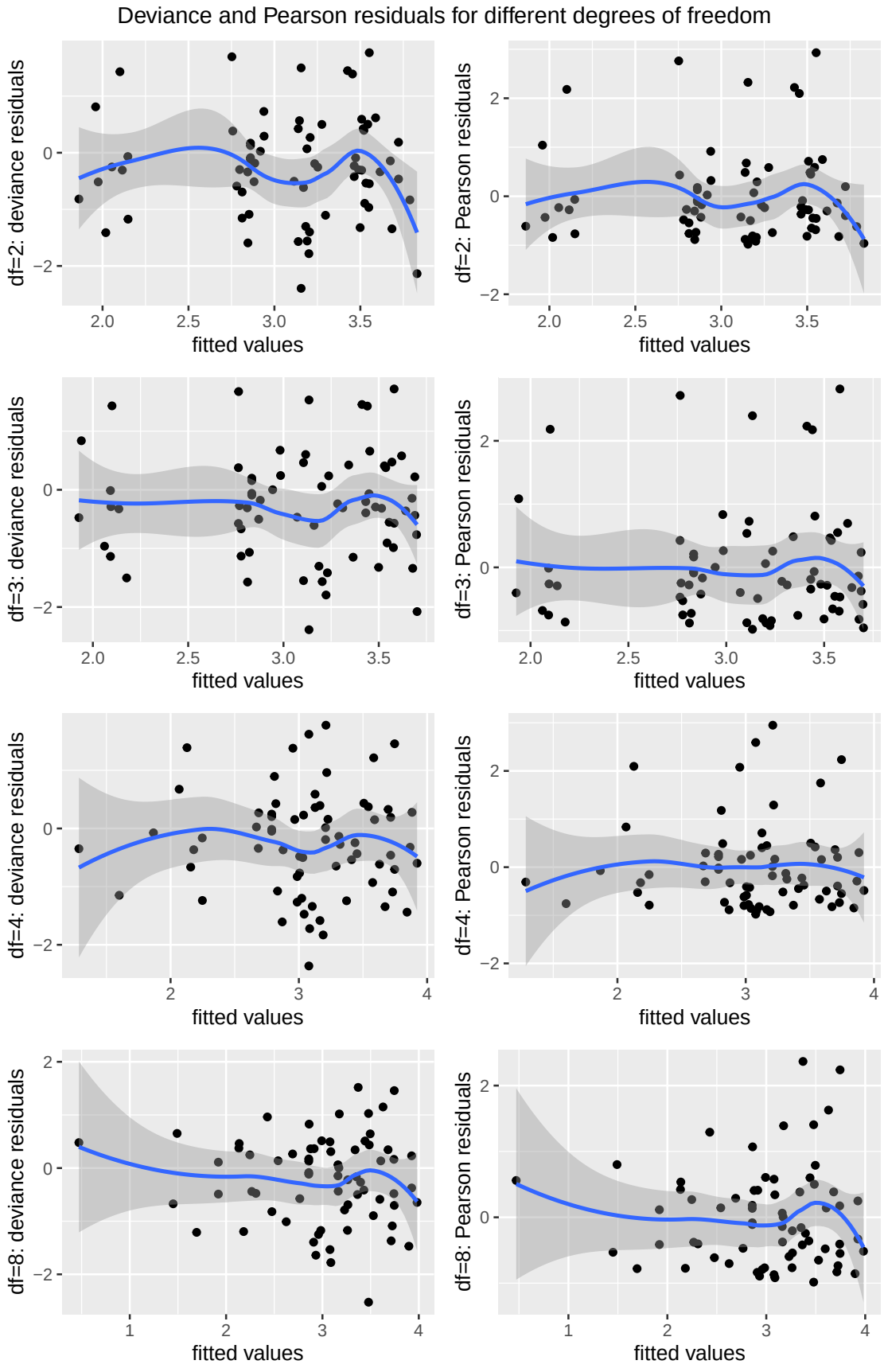
We would like to find out if nonlinear effects help the model. We try to implement splines in order to get a better fit:

```
glm_SOIgamma2df<-glm(Rain ~ ns(SOI, df=2)+Phase, family = Gamma(link="log"),
                data = Rain.data.wet)
glm_SOIgamma3df<-glm(Rain ~ ns(SOI, df=3)+Phase, family = Gamma(link="log"),
                data = Rain.data.wet)
glm_SOIgamma4df<-glm(Rain ~ ns(SOI, df=4)+Phase, family = Gamma(link="log"),
                data = Rain.data.wet)
glm_SOIgamma8df<-glm(Rain ~ ns(SOI, df=8)+Phase, family = Gamma(link="log"),
                data = Rain.data.wet)
```

We now plot the Pearson and deviance residuals for the models to see how well the structure is captured by the model:

```
glm_SOIgamma2dfd<-qplot(fortify(glm_SOIgamma2df)$.fitted, residuals(glm_SOIgamma2df,
                                                                    type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("df=2: deviance residuals")+
```

```
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma2dfp<-qplot(fortify(glm_SOIgamma2df)$.fitted, residuals(glm_SOIgamma2df,
                                                                    type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("df=2: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma3dfd<-qplot(fortify(glm_SOIgamma3df)$.fitted, residuals(glm_SOIgamma3df,
                                                                    type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("df=3: deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma3dfp<-qplot(fortify(glm_SOIgamma3df)$.fitted, residuals(glm_SOIgamma3df,
                                                                    type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("df=3: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))
glm_SOIgamma4dfd<-qplot(fortify(glm_SOIgamma4df)$.fitted, residuals(glm_SOIgamma4df,
                                                                    type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("df=4: deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma4dfp<-qplot(fortify(glm_SOIgamma4df)$.fitted, residuals(glm_SOIgamma4df,
                                                                    type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("df=4: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma8dfd<-qplot(fortify(glm_SOIgamma8df)$.fitted, residuals(glm_SOIgamma8df,
                                                                    type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("df=8: deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

glm_SOIgamma8dfp<-qplot(fortify(glm_SOIgamma8df)$.fitted, residuals(glm_SOIgamma8df,
                                                                    type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("df=8: Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))

grid.arrange(glm_SOIgamma2dfd, glm_SOIgamma2dfp, glm_SOIgamma3dfd, glm_SOIgamma3dfp,
             glm_SOIgamma4dfd, glm_SOIgamma4dfp, glm_SOIgamma8dfd, glm_SOIgamma8dfp,
             ncol = 2, top = "Deviance and Pearson residuals for different degrees of freedom")
```


Deviance and Pearson residuals for different degrees of freedom
df=2: deviance residuals
df=2: Pearson residuals
df=3: deviance residuals
df=3: Pearson residuals
df=4: deviance residuals
df=4: Pearson residuals
df=8: deviance residuals
df=8: Pearson residuals
fitted values

From the above plots, using 3 degrees of freedom seems to give the best fit, however to test which of the models have the best predictive strength, we do Cross validation. We do it with the leave-one-out method, jackknifing. This is because in the data set, we only have very few observations for Phase=1 and Phase=3, hence making the dataset even smaller for say k=10, the probability for us to have observations for each phase is low. By doing jackknifing we only remove 1 observation, hence we will never have problems with observations missing for some phase.

We implement a code to do cross validation, inspired from the code p. 202 in the book. We use the deviance-residuals for gamma-model in the computation, and try for different degrees of freedom. We use 1, 2, 3, 4, 8 and 15 degrees of freedom:

```
testCV_gam <- function(data, B = 1, k = 68, df) {

  form <- Rain ~ ns(SOI,df=df)+Phase
  n <- nrow(data)
  PEcv <- vector("list", B)
  tmp <- numeric(n)

  for (b in 1:B) {
    ## Generating the random division into groups
    group <- sample(rep(1:k, length.out = n))

    for (i in 1:k) {
      modelcv <- glm(form, family = Gamma("log"),
                     data = data[group != i, ])
      muhat <- predict(modelcv, newdata = data[group == i, ], type = "response")

      ## Getting the actual observed Dead and Survived values for the current fold
      observed <- data[group == i,]

      ## Calculate the deviance residuals
      Y <- observed$Rain
      ##resids <- (data$Rain[group == i] - muhat)^2
      resids <- 2*(log(muhat/Y)+Y/muhat-1)

      tmp[group == i] <- resids
    }

    PEcv[[b]] <- tmp
  }

  mean(unlist(PEcv))
}

sapply(c(1,2,3,4,8,15),function(x) testCV_gam(Rain.data.wet, k=nrow(Rain.data.wet), df=x))
```

```
## [1] 1.114915 1.187205 1.251844 1.230014 1.176106 2.554672
```

We see from the models above, that the models with splines all have quite a lot less predictive strength. With an expected generalization error much lower when not having splines introduced, the only argument for using splines would be the better mean and variance structure. However the improvement does not seem significantly enough to justify a worse predictive strength.

From all the above calculations, it seems reasonable to choose the Gamma-model based on SOI and Phase without any splines.

We now have two models, the Gamma model and the Tweedie model, both on the form Rain~SOI+Phase. But which model would be the best to choose when in the same situation.

To find out, we test the Tweedie-model with the data which was used in the Gamma-case, hence only for the years with some rainfall in July.

We first find the optimal k-value:

```
AICopt <- Vectorize(function(k)
  AICtweedie(glm(Rain ~ SOI+Phase, family = tweedie(var.power = k, link.power = 0),
                 data=Rain.data.wet)))

(koptimal_tweedie_wet <- optimize(AICopt, c(1, 2), maximum = FALSE))
```

```
## $minimum
## [1] 1.99993
##
## $objective
## [1] 576.2201
```

Hence the optimal k-value is ≈ 2.

We also plot the residuals for the Tweedie model when in this case.

```
glm_tweedie_wet<-glm(Rain ~ SOI+Phase,
                     family = tweedie(var.power = koptimal_tweedie_wet$minimum,
                                      link.power = 0), data = Rain.data.wet)

d1_glm_tweedie_wet<-qplot(fortify(glm_tweedie_wet)$.fitted, residuals(glm_tweedie_wet,
                                                                      type="deviance"))+
  geom_smooth()+xlab("fitted values")+ylab("deviance residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))


p1_glm_tweedie_wet<-qplot(fortify(glm_tweedie_wet)$.fitted, residuals(glm_tweedie_wet,
                                                                      type="pearson"))+
  geom_smooth()+xlab("fitted values")+ylab("Pearson residuals")+
  scale_y_continuous(breaks = seq(-10, 15, by = 2),
                     labels = seq(-10, 15, by = 2))


grid.arrange(d1_glm_tweedie_wet,p1_glm_tweedie_wet,ncol=2,
             top="Residuals against fitted for deviance and Pearson residuals")
```

These residual plots seem suspiciously close to the plots for the Gamma-GLM. We also calculate the CV-value for the Tweedie-model. However to do so, we must make a modification to the former CV-code for the Tweedie-model; which we do below:

```
set.seed(2024)
testCV_ns2 <- function(data, B = 1, k = 68, df) {

  form <- Rain ~ ns(SOI, df = df)+Phase
  n <- nrow(data)
  PEcv <- vector("list", B)
  tmp <- numeric(n)

  CVAIC <- Vectorize(function(k)
  AICtweedie(glm(form, family = tweedie(var.power = k, link.power = 0),
                 data=Rain.data.wet))
  )

  p <- optimize(CVAIC, c(1, 2), maximum = FALSE)$minimum

  for (b in 1:B) {
    ## Generating the random division into groups
    group <- sample(rep(1:k, length.out = n))

    for (i in 1:k) {
      modelcv <- glm(form, family = tweedie(var.power = p, link.power = 0),
                     data = data[group != i, ])
      muhat <- predict(modelcv, newdata = data[group == i, ], type = "response")
```

```
      observed <- data[group == i,]

      ## Calculate the residuals
      Y <- observed$Rain
      ##resids <- (data$Rain[group == i] - muhat)^2
      resids <- 2/(1-p)*(Y^(2-p)-Y*muhat^(1-p))+2/(2-p)*(muhat^(2-p)-Y^(2-p))

      tmp[group == i] <- resids
    }

    PEcv[[b]] <- tmp
  }

  mean(unlist(PEcv))
}
```

We now calculate the CV-value for the Tweedie model without any splines (df=1).

```
testCV_ns2(Rain.data.wet, k=nrow(Rain.data.wet), df=1)
```

```
## [1] 1.115118
```

We see that this value is almost the complete same as the CV-value for the gamma-model without splines.

All theses values being the same can not be random. But by recalling the first theoretical assignment, we see that for $k = 2$ the Tweedie distribution has same variance function as the gamma-distribution $\Gamma(\lambda, \alpha)$. And for the data for Rain>0 we just found $k = 2$. This means that when modelling the Rain given SOI and Phase, we have a gamma-distribution, and for data with some observations with Rain=0, we get a lower k-value, which is not any known distribution.

Since we get the same results for the gamma-model and Tweedie-model for Rain>0, but the gamma-model does not work when we have observations of Rain=0, the model to prefer is the Tweedie-model. The empirical proportion of zero is around 38% hence it may be a bit extreme to throw out so much data and use a model that can only be used around 62% of the time.

When in the case without observations of no rainfall the Tweedie-model will just be the gamma-model which we found to have a beautiful mean and variance structure, and hence have a good fit on the data. However when years without rain are observed the Tweedie-model does not seems as good. But at least it can be used and give an approximation to the rainfall, even though it might not be a perfect fit. When predicting Rain using this full data set the model to choose is a Tweedie model in the form Rain~SOI+Phase with $k = 1.434232$. We look at the final model for the last time.

```
form1
```

```
## Rain ~ SOI + Phase
```

```
summary(SOIglm1)
```

```
##
## Call:
## glm(formula = form1, family = tweedie(var.power = optimize(AICglm1,
##     c(1, 2), maximum = FALSE)$minimum, link.power = 0), data = Rain.data.small)
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.22307    0.58124   2.104  0.03774 *
## SOI          0.01408    0.02198   0.641  0.52320
## Phase2       2.19320    0.75223   2.916  0.00434 **
## Phase3      -0.20145    0.90114  -0.224  0.82354
## Phase4       1.56068    0.68831   2.267  0.02541 *
## Phase5       1.00359    0.62773   1.599  0.11288
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Tweedie family taken to be 8.069734)
##
##     Null deviance: 1162.27  on 110  degrees of freedom
## Residual deviance:  850.48  on 105  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 6
```

```
confint(SOIglm1)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %     97.5 %
## (Intercept)  0.10848734 2.31024948
## SOI         -0.02725825 0.05517398
## Phase2       0.80807436 3.58894894
## Phase3      -1.97126525 1.52459849
## Phase4       0.24014901 2.89592435
## Phase5      -0.17961849 2.20927851
```

We see that SOI has 0 included in the confidence interval, hence it might not be significant, but this is likely caused by the fact that there are so many observations of 0s in the data set. It might therefore not be a very good predictor of Rain in Eromanga. But Phase seems to be significant. Maybe if we had more variables like temperature or air pressure we would have a better model since we know rainfall is a complex phenomenon, and predicting it perfectly using only two variables is thus close to impossible.