Project in regression - practical part

Contents

Exploratory data analysis **1**
Missing data . . . 5

Analysis using SOI phase **7**
Estimating k with a linear regression model . . . 7
Fitting a Tweedie model to the data . . . 11
Estimating probability of zero rain in July . . . 12
Determining k by minimizing AIC . . . 12
Re-estimating with new value of k . . . 14
Model diagnostics . . . 15
Conclusion . . . 18
Bootstrap estimates of k . . . 18

Analysis using SOI directly **21**
Additional predictors . . . 23
Nonlinear effects . . . 23
Model Diagnostics . . . 25
Reporting a final model and interpretation . . . 27
Conclusion . . . 30

In this project we seek to predict the rainfall at Eromanga in Queensland, Australia, during the month of July. We consider a data set that contains total rainfall in July at Eromanga in the period 1905 to 2024 with the exception of a few years. In addition we have measurements of the southern oscillation index (SOI, the standardized difference between the air pressures at Darwin and Haiti, related to el niño) for the same years. The hypothesis is that the SOI is related to the rainfall in Eromanga. We will throughout the project investigate this hypothesis and ultimately use the SOI index to predict the rainfall in Eromanga.

To increase readability we have omitted code chunks on several occasions. In particular when it comes to plots and other standard code parts. When the code and output is more involved and essential for the analysis and understanding of the project we have included it in the project. If anything is unclear all code can be accessed at https://github.com/miclukacova/reg_project.

Exploratory data analysis

First, we load the data set

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

We specify the column classes and print the `head()` and `summary()` of the data set to make sure that the data is read in correctly.

Year	Rain	Month	SOI	Phase
1905	0	7	-19.8	1

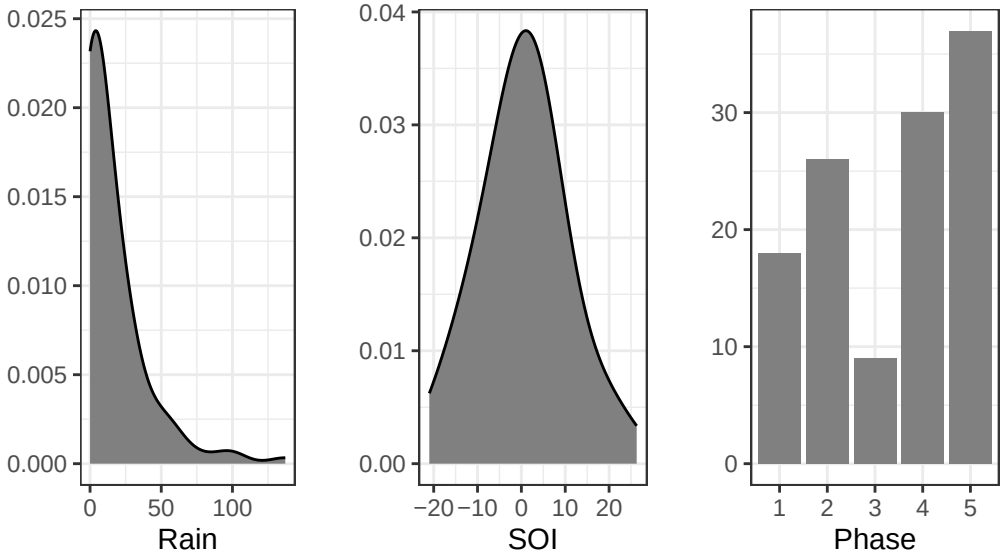
Year	Rain	Month	SOI	Phase
1906	20.8	7	6.3	4
1907	12	7	-5.1	5
1908	NA	7	-3.2	5
1909	NA	7	9.9	2
1910	NA	7	19	2

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

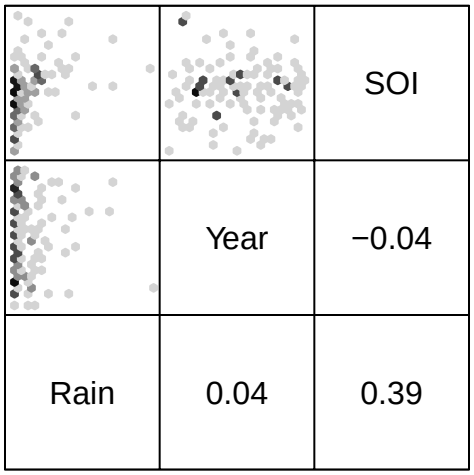
The data set contains five variables: `Year`, `Rain`, `SOI`, `Phase` and `Month`. The `Month` variable is constant and will not be used or analyzed, but keep in mind, that the analysis is carried out for the month of July. The response variable `Rain` represents the total rainfall in July at Eromanga and contains nine missing values, which will be addressed in a subsequent part of the EDA. The `SOI` variable is the southern oscillation index, while `Phase` is a categorical variable indicating the SOI phase on five different levels:

- Phase 1: Consistently negative
- Phase 2: Consistently positive
- Phase 3: Rapidly falling
- Phase 4: Rapidly rising
- Phase 5: Consistently near zero

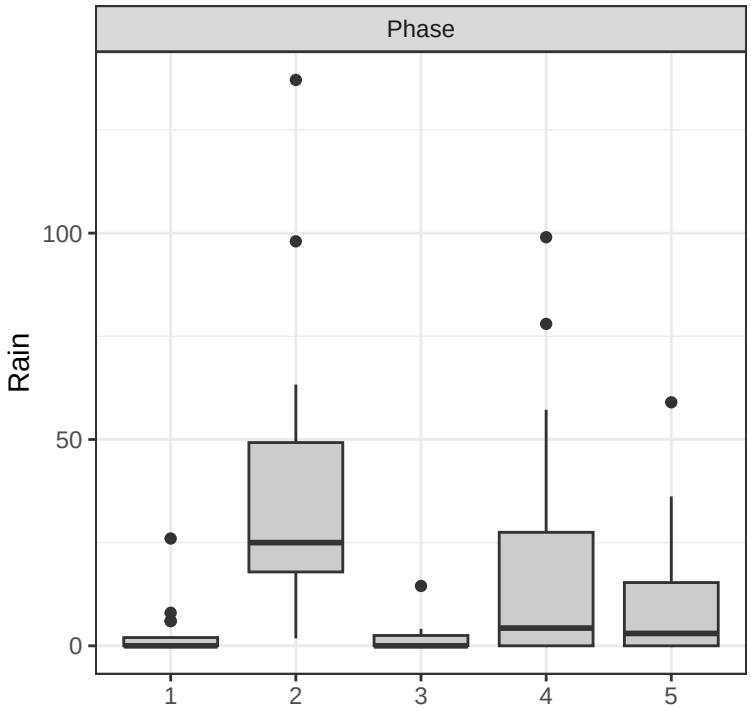
We plot the marginal distributions of the variables to visually explore the data.



We observe that the response variable `Rain` is right-skewed, which should be considered in later analysis. Additionally, Category 3 in `Phase` has relatively few observations, which may also require consideration in subsequent analysis. We proceed to investigate possible co-linearity between the variables in the data set. We assess correlation between the numerical variables `Rainfall`, `SOI` and `Year` with a correlation plot:



`Year` is almost uncorrelated with both `Rain` and `SOI`. There is a weak positive correlation between `Rain` and `SOI`. To investigate the relation between `Rain` and `Phase` we consider the distribution of `Rain` stratified by `Phase`:



The boxplot suggests that the rainfall is larger in phase 2 and 4, while it appears particularly low in phase 1 and 2. The plot display some correlation between `Phase` and `Rain` as there appear to be a difference in rainfall within the different phases. We finally investigate the correlation between `SOI` and `Phase`. Since both variables describe aspects of SOI we expect some correlation between them.

```
pander(summary(lm(SOI ~ Phase, Rain.data)))
```

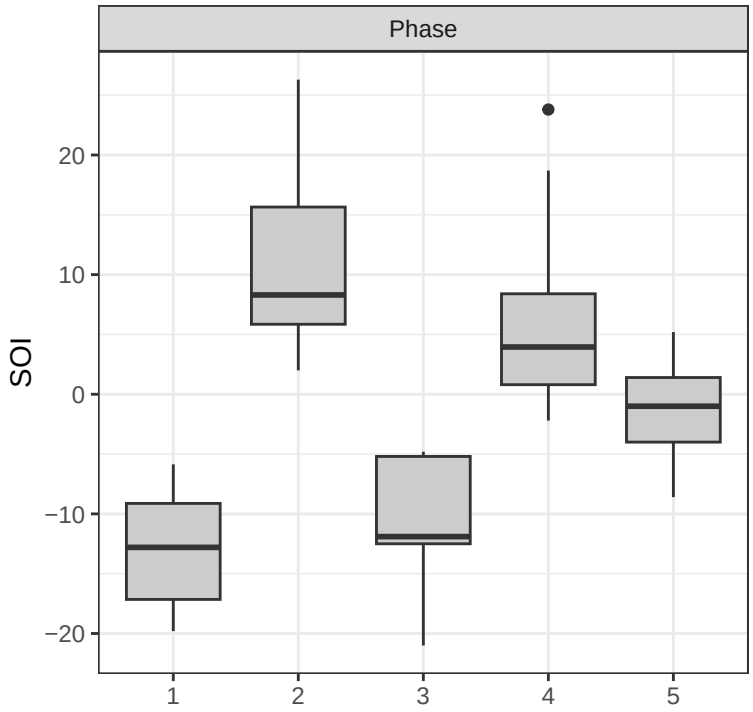
	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-12.61	1.282	-9.832	6.342e-17
Phase2	23.01	1.668	13.8	3.812e-26
Phase3	1.838	2.221	0.8278	0.4095
Phase4	17.98	1.622	11.09	7.127e-20
Phase5	11.24	1.563	7.193	6.903e-11

	Estimate	Std. Error	t value	Pr(>|t|)

Table 4: Fitting linear model: SOI ~ Phase

Observations	Residual Std. Error	R^2	Adjusted R^2
120	5.439	0.6915	0.6808

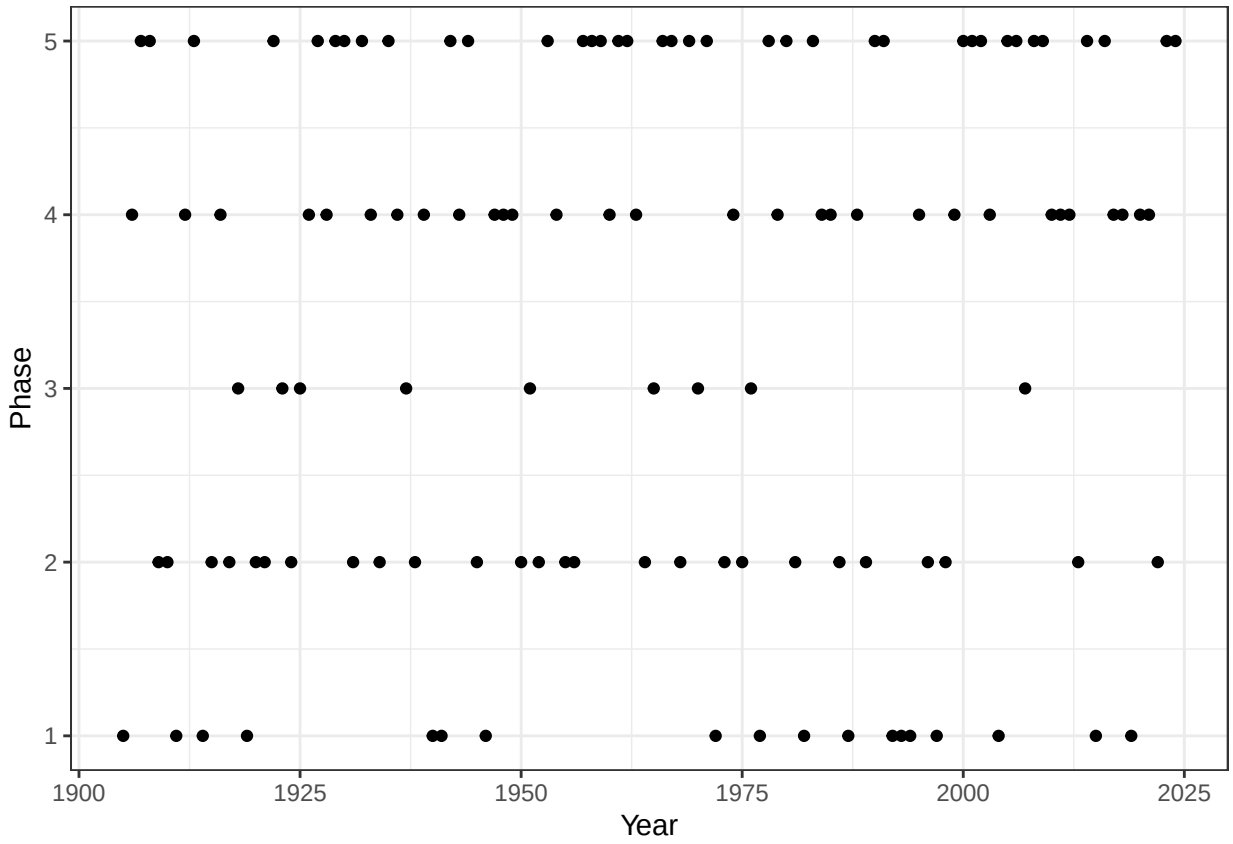
The simple linear model uncovers a high adjusted R^2 implying a high correlation between the two variables. All coefficients except `Phase3` are significant indicating that the `Phase` variable is a good predictor of `SOI`. This discovery is further supported by the following plot



Since the two predictors are collinear to some extent we will be careful, when we consider models with both covariates. It is worth noting though, that while `SOI` is a numeric variable containing exact values of SOI, `Phase` contains information on the 'direction' of the SOI, i.e. whether it is increasing, decreasing or constant. So although the two variables are correlated they both contain information that the other does not, so it could still be valuable to include both variables in a model. We will go into further details on this matter in subsequent sections.

The last two variables for which we have not checked the correlation are the variables `Year` and `Phase`. Since `Phase` is categorical and `Year` ordinal, there is no suitable substitute for the Pearson correlation. We plot the two variables against each other

```
ggplot(Rain.data, aes(x = Year, y = as.numeric(Phase))) +
  geom_point()+
  xlab("Year") +
  ylab("Phase") +
  theme_bw()
```



The plot does not suggest any dependence between `Year` and `Phase`.

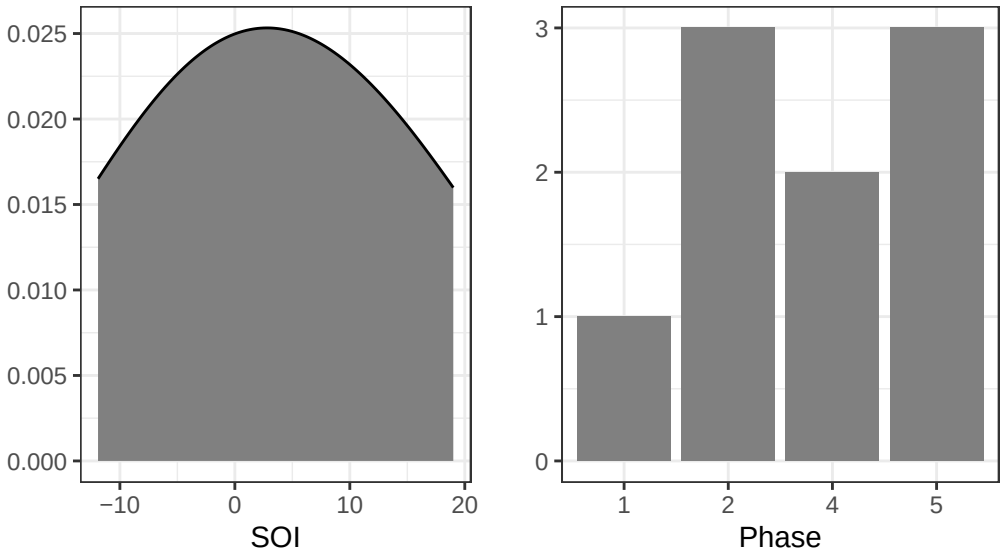
Missing data

With a better understanding of the data we proceed to investigate and handle the missing data. As noted previously the only missing data is in the response variable `Rain` (7.5 pct. missing).

```
pander(Rain.data %>% dplyr::filter(is.na(Rain)))
```

Year	Rain	Month	SOI	Phase
1908	NA	7	-3.2	5
1909	NA	7	9.9	2
1910	NA	7	19	2
1911	NA	7	-11.9	1
1933	NA	7	3.3	4
2021	NA	7	16.26	4
2022	NA	7	7.63	2
2023	NA	7	-3.32	5
2024	NA	7	-5.83	5

We plot the marginal distribution of the variables with missing data to obtain a visual understanding of patterns in the missing data:



Given the modest number of missing observations, it is difficult to identify any clear trends in the missingness. We note that the missingness occurs in consecutive years (except 1933), but there are no distinct pattern beyond that. Without metadata and additional knowledge we are unable to determine if the data is missing completely at random (MCAR), at random (MAR) or not at random (MNAR). It seems unlikely that observations of large rainfall were selectively deleted or that someone was too lazy to record rainfall during heavy rainfalls. A more plausible explanation, given the consecutive years, is that the equipment may be broken for consecutive years or there was a lack of funding these years. The latter explanations would imply that the data is MCAR which implies MAR and we therefore choose to adopt this assumption.

Assuming MAR we can use multiple imputation to impute the missing data. If the MAR assumption holds the imputed values are unbiased and the variation in the data set is preserved. We use the `mice()` function (multiple imputations using chained equations) from the package `mice` to perform the multiple imputations. This procedure uses random draws from the conditional distribution of the target variable given the other variables. That is, we take a bootstrap sample from our data and fit a regression model to this sample to predict the missing values. We use *predictive mean matching* (PMM) to replace the missing values in each imputed data set. This method uses the value of a donor observation to fill in the missing values. The donors are identified by matching the predicted value of the target to the donor value. PMM does not require any distributional assumptions and is therefore fairly robust to different types of data (F. E. Harell, Jr., 2015).

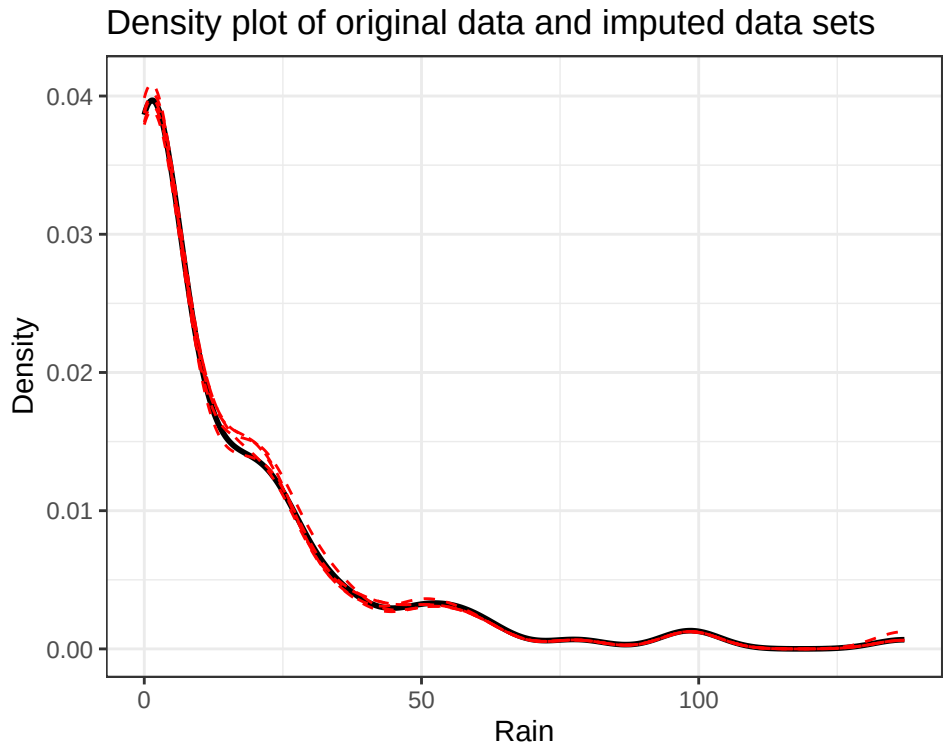
We run the imputations and display the first five rows of the first two imputed data sets:

```
Rain.data.impute <- mice(Rain.data, method = "pmm",
                         m = 5, seed = 10102024,
                         printFlag = FALSE)
pander(head(complete(Rain.data.impute,1)))
pander(head(complete(Rain.data.impute,2)))
```

Year	Rain	Month	SOI	Phase
1905	0	7	-19.8	1
1906	20.8	7	6.3	4
1907	12	7	-5.1	5
1908	0	7	-3.2	5
1909	19.8	7	9.9	2
1910	2.8	7	19	2

Year	Rain	Month	SOI	Phase
1905	0	7	-19.8	1
1906	20.8	7	6.3	4
1907	12	7	-5.1	5
1908	1.2	7	-3.2	5
1909	19	7	9.9	2
1910	19	7	19	2

We plot the density of the five imputed data sets (red dotted) along with the density of the original data set (black solid) to check if the imputed data sets resemble each other and the original data set.



We note that the densities for all the imputed data sets look very similar to the original data set. We therefore proceed with the imputed data sets.

Analysis using SOI phase

In the following part of the project we seek to fit the Tweedie exponential dispersion model to the Eromanga rain data, and predict rainfall as a function of the SOI phase. To fit a Tweedie model we need to estimate the nuisance parameter k assumed to be between 1 and 2.

Estimating k with a linear regression model

We initially try to estimate k using the linear relation $VY = \psi\mathcal{V}(\mu) = \psi\mu^k$. This implies that $\log(VY) = \log(\psi) + k\log(\mu)$ and we can therefore estimate k by a linear regression of $\log(VY)$ on $\log(\mu)$. We compute the empirical variance and mean of the `Rain` variable within each SOI phase for each imputed data set and display the result for the first imputed data set:

```
grouped_imputatations <- list()
for (i in 1:5) {
```

```
  grouped_imputatations[[i]] <- complete(Rain.data.impute, i) %>% group_by(Phase) %>%
    summarise(meanY = mean(Rain), varY = var(Rain))
}
pander(grouped_imputatations[[1]])
```

Phase	meanY	varY
1	2.733	40.17
2	32.7	915.6
3	2.344	22.98
4	18.03	640.5
5	8.359	161

Using the empirical mean and variance of Y in each SOI phase we proceed to fit an additive linear regression to estimate k and the dispersion parameter for each imputed data set:

```
lm.fit.imputed <- list()
for (i in 1:5) {
  lm.fit.imputed[[i]] <- lm(log(varY) ~ log(meanY), data = grouped_imputatations[[i]])
}
pander(lm.fit.imputed)
```

Table 9: Fitting linear model: log(varY) ~ log(meanY)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.129	0.2441	8.721	0.003174
log(meanY)	1.404	0.1055	13.31	0.0009163

-

Table 10: Fitting linear model: log(varY) ~ log(meanY)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.133	0.2505	8.516	0.003402
log(meanY)	1.389	0.1076	12.9	0.001005

-

Table 11: Fitting linear model: log(varY) ~ log(meanY)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.975	0.2331	8.469	0.003456
log(meanY)	1.47	0.09743	15.09	0.0006316

-

Table 12: Fitting linear model: log(varY) ~ log(meanY)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.123	0.2815	7.543	0.004831
log(meanY)	1.388	0.1204	11.53	0.001402

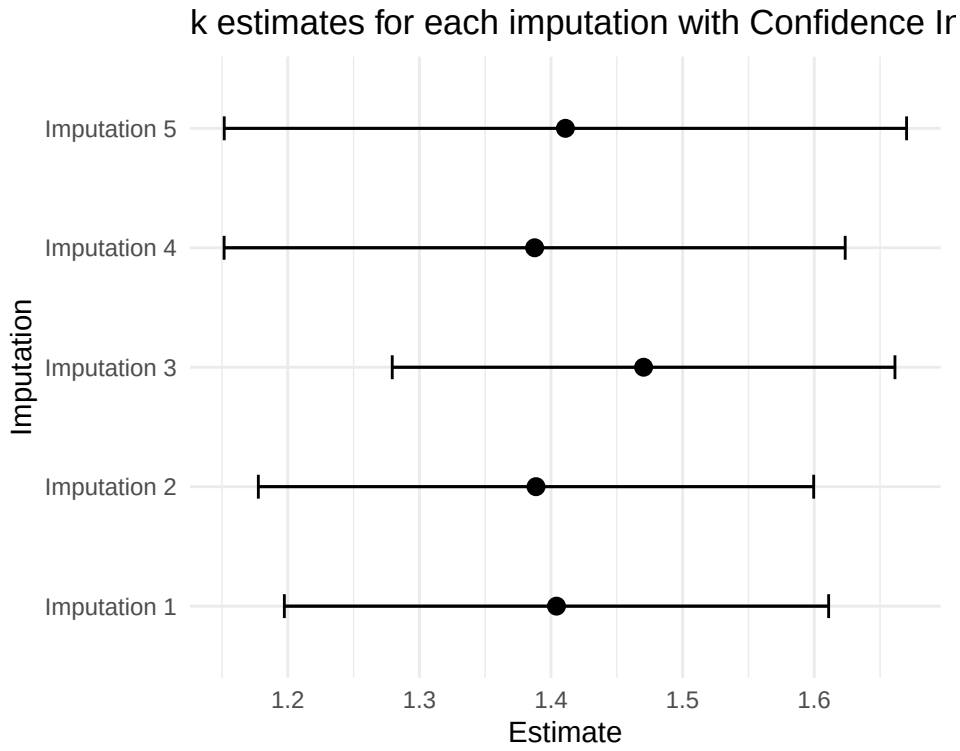
-

Table 13: Fitting linear model: log(varY) ~ log(meanY)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.096	0.3067	6.834	0.006412
log(meanY)	1.411	0.1322	10.67	0.00176

-

We get an initial estimate of k between 1.388 and 1.470 and ψ between $\exp(1.975) = 7.201$ and $\exp(2.133) = 8.440$. Like the density plots the predictions are fairly similar. We compute simple confidence interval (based on the above table output) for the estimates of each imputation and plot them:



As all estimates of k are within the same margin of error (according to these confidence intervals), we choose to merge the imputations into a single data set to simplify future computations and communicate results more clearly. We should however keep in mind that this may artificially reduce the variance of the data slightly and in a more thorough analysis we should keep all five imputed data sets.

We use the `merge_imputations` from the `sjmisc` package which merges multiple imputed data frames from `mice::mids()-objects` into a single data frame by computing the mean or selecting the most likely imputed value.

```
Rain.data.comp <- Rain.data %>%
  mutate(Rain = merge_imputations(Rain.data, Rain.data.impute)$Rain)

Rain.data.grp <- Rain.data.comp %>%
  group_by(Phase) %>%
  summarise(meanY = mean(Rain), varY = var(Rain))

lm.fit <- lm(log(varY) ~ log(meanY), data = Rain.data.grp)
pander(lm.fit)
```

Table 14: Fitting linear model: log(varY) ~ log(meanY)

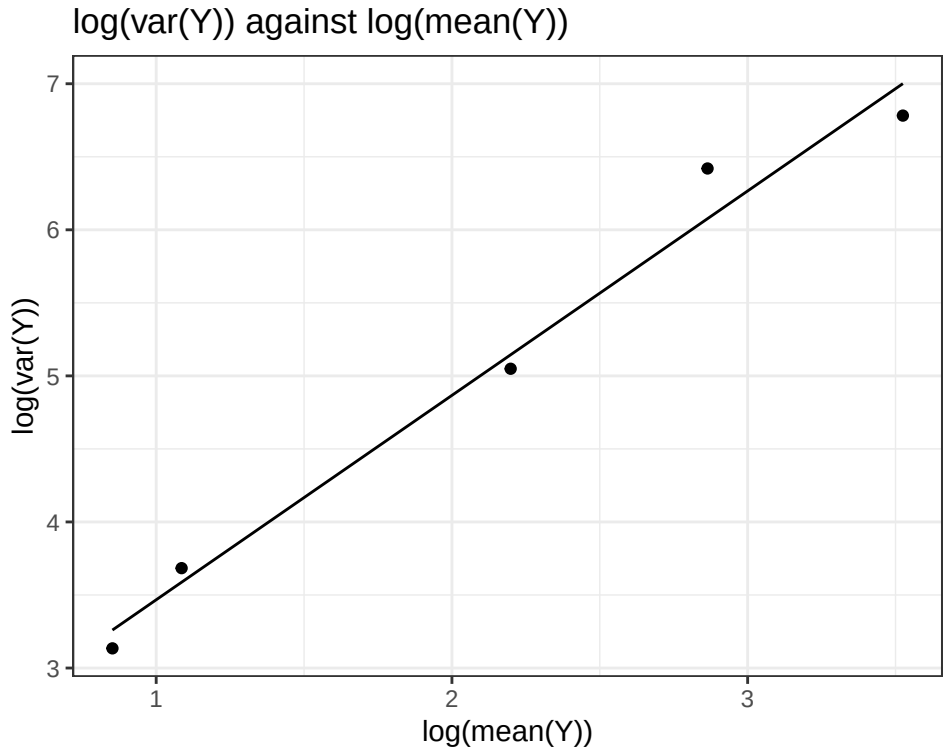
	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.068	0.2647	7.81	0.00437
log(meanY)	1.4	0.1132	12.37	0.001139

We end up with estimates for k and ψ of

```
k_hat_lin <- lm.fit$coef[[2]]
psi_hat_lin <- exp(lm.fit$coef[[1]])

pander(c("Estimate of k" = k_hat_lin, "Estimate of psi" = psi_hat_lin))
```

Estimate of k	Estimate of psi
1.4	7.905

We can visually check that the estimated regression coefficients are reasonably estimated by plotting the fitted regression line on top of the data:



We see no warning signs from the plot so is seems fair to assume, that the estimated values of k and ψ are decent initial estimates of the true values.

Fitting a Tweedie model to the data

We proceed to fit a Tweedie model to the data. We use the `tweedie()` family specification from the `tweedie` package to fit the model with the estimated value of k and `link.power = 0` for the log-link.

```
tweedie.fit <- glm(Rain ~ Phase, data = Rain.data.comp,
                   family = tweedie(var.power = k_hat_lin, link.power = 0))
pander(summary(tweedie.fit))
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.086	0.4875	2.227	0.02786
Phase2	2.439	0.5251	4.645	9.114e-06
Phase3	-0.2339	0.8858	-0.264	0.7922
Phase4	1.778	0.5354	3.321	0.001201
Phase5	1.113	0.5449	2.042	0.04345

(Dispersion parameter for Tweedie family taken to be 8.212105)

Null deviance:	1260 on 119 degrees of freedom
Residual deviance:	910 on 115 degrees of freedom

To interpret the model output we recall the five SOI levels

- Phase 1: Consistently negative
- Phase 2: Consistently positive
- Phase 3: Rapidly falling

- Phase 4: Rapidly rising
- Phase 5: Consistently near zero

Note that Phase 1 is the reference phase (Intercept). The model suggests that the rainfall for Phase 1 is significantly different from 0 with a point estimate of the average rainfall of *2.962* millimeters in July. The model estimates that rainfall for Phase 2 is significantly different from the rainfall in Phase 1 with a point estimate of average rainfall of *33.95* millimeters in July when SOI is in this phase. The model estimates that the rainfall for Phase 3 is insignificantly different from the rainfall in Phase 1 with a point estimate of rainfall of *2.344* millimeters on average in July. The rainfall for Phase 4 is significantly different from the rainfall in Phase 1 according to the model with a point estimate of average rainfall of *17.54* millimeters in July. Finally, the model estimates rainfall for Phase 5 to be borderline significantly different from rainfall in Phase 1 with a point estimate of average rainfall of *9.012* millimeters in July. Hence, the model suggests that Phase 2 and Phase 4, which are when the SOI is consistently positive and rapidly rising respectively, leads to significantly more rain on average than Phase 1. Phase 5, when the SOI is consistently near zero, is predicted to have slightly more rainfall on average, compared to Phase 1 while the model predicts the lowest rainfall on average in SOI Phases 1 and 3 with no significant difference between them.

Furthermore, the model predicts ψ to be *8.212* which is slightly different from the result obtained from the linear regression where ψ was estimated to be *7.905* .

Estimating probability of zero rain in July

We use the estimates obtained in the previous exercises to estimate the probability that it will not rain in July. In the theoretical exercises we derived the probability of zero rain to be

$$\mathbb{P}(Y = 0) = \exp(-\lambda^*) = \exp(-\frac{\mu^{2-k}}{\psi(2-k)})$$

With the two estimates of ψ from the previous exercise, we compute two estimates of the the probability that it will not rain in July. We plug in the estimated values of k and the empirical mean of our data:

```
mu_hat <- mean(Rain.data.comp$Rain)

pander(c("Estimate using psi from linear model"
         = exp(-mu_hat^(2 - k_hat_lin)/(psi_hat_lin*(2 - k_hat_lin))),
         "Estimate using psi from Tweedie model"
         = exp(-mu_hat^(2 - k_hat_lin)/(psi_hat_tweedie.fit*(2 - k_hat_lin)))))
```

Estimate using psi from linear model	Estimate using psi from Tweedie model
0.3406	0.3546

The two results are quite similar and compared to the empirical probability of zero rain in July

```
pander(sum(Rain.data.comp$Rain == 0)/nrow(Rain.data.comp))
```

0.35 we obtain three estimates that are all very similar.

Determining k by minimizing AIC

We now estimate k by minimizing the Akaike Information Criterion (AIC) with a profile likelihood of a model with SOI phase as explanatory variable. That is, we search for the value of $k \in (1, 2)$ that minimizes the AIC. We start by constructing a general profile likelihood function that takes inputs a formula, a family, a data set and an evaluation metric that we wish to optimize.

```
profile_likelihood <- function(form, family, data, eval) {
  model <- glm(form,
               family = family,
               data = data)
  eval_val <- eval(model)
  return(eval_val)
}
```

We define the specific profile likelihood that minimizes the AIC for different values of k of a Tweedie exponential model with rainfall as response and SOI phase as covariate. That is, we specify `form = Rain ~ Phase`, `family = tweedie(var.power = k, link.power = 0)`, `data = Rain.data.comp` and `eval = AICtweedie`:

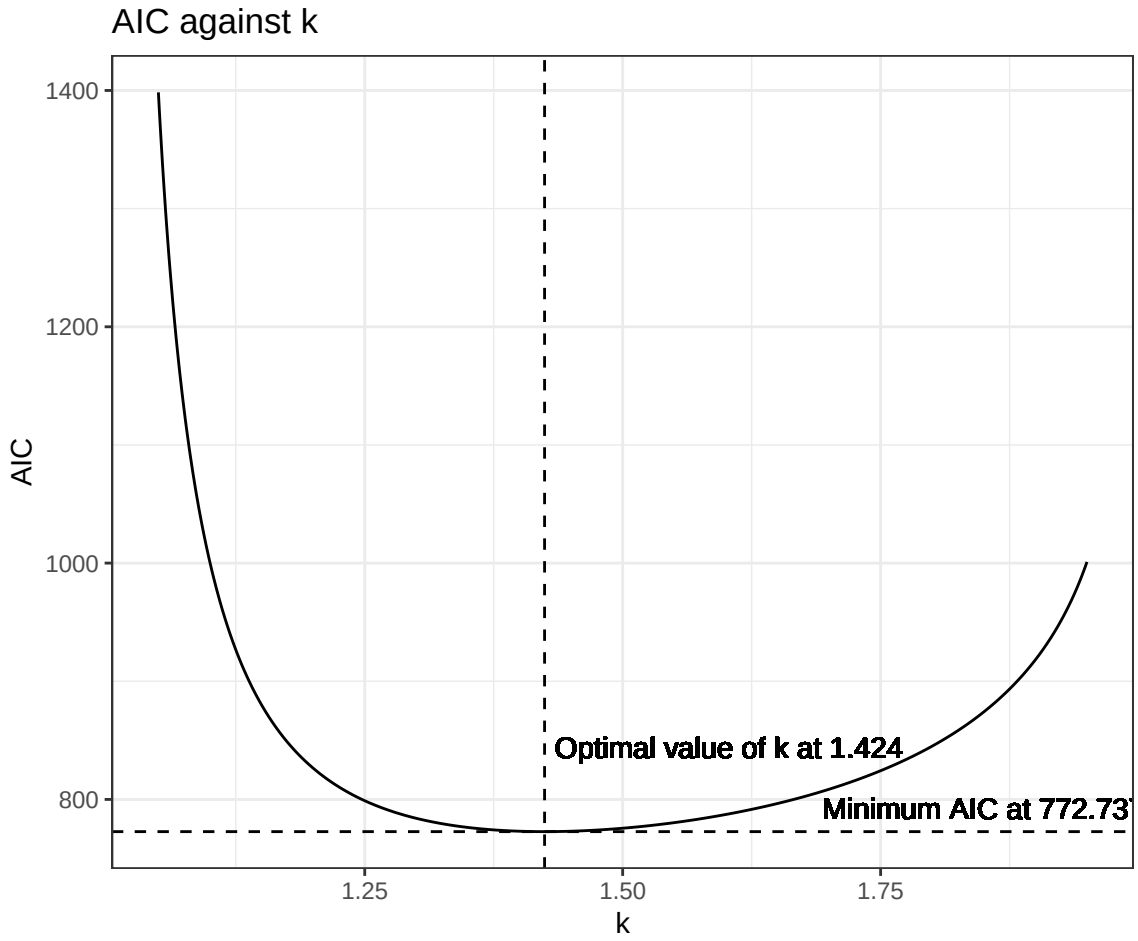
```
tweedie.AIC_profile_likelihood <- function(k) {
  profile_likelihood(form = Rain ~ Phase,
                         family = tweedie(var.power = k, link.power = 0),
                         data = Rain.data.comp,
                         eval = AICtweedie)
}
```

We use `optimize()` to minimize the AIC and find the optimal value of k. We only check values of k between 1.05 and 1.95 to avoid numerical instability:

```
k_hat_AIC <- optimize(tweedie.AIC_profile_likelihood, lower = 1.05, upper = 1.95)$minimum
pander(k_hat_AIC)
```

1.424

We note that the optimal value of k using the AIC profile likelihood method is fairly close to the value of k estimated by the linear model *1.4* . To ensure, that we have found a minimum for $1 < k < 2$ we plot the AIC against k and add a vertical and a horizontal line at the optimal value of k.



The plot confirms, that `optimize()` has found the global minimum of the AIC for $1 < k < 2$. We repeat the calculations from the previous exercises using the optimal value of k found by minimizing the profile likelihood.

Re-estimating with new value of k

As before we initially fit a Tweedie model to the data using the optimal value of k found by minimizing the AIC profile likelihood.

```
tweedie.fit.AIC <- glm(Rain ~ Phase, data = Rain.data.comp,
                       family = tweedie(var.power = k_hat_AIC, link.power = 0))
pander(summary(tweedie.fit.AIC))
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.086	0.4797	2.264	0.02547
Phase2	2.439	0.5189	4.7	7.267e-06
Phase3	-0.2339	0.8699	-0.2689	0.7885
Phase4	1.778	0.5289	3.363	0.001049
Phase5	1.113	0.5377	2.069	0.04076

(Dispersion parameter for Tweedie family taken to be 7.739979)

Null deviance:	1203.2 on 119 degrees of freedom

Residual deviance:	874.5 on 115 degrees of freedom

We note a slight decrease in standard error and p-value for all coefficients and the estimated dispersion parameter *7.74* is slightly different from the previous estimate *8.212* of the Tweedie model. Apart from that the results are identical to the previous model and the interpretation is the same.

We recalculate the estimated probability that it will not rain in July by plugging in the estimated values of k and ψ and the empirical mean of our data:

Estimate using psi from linear model	Estimate using psi from Tweedie model
0.3499	0.3421

Again the estimates are similar to the estimates obtained from the previous estimate of k.

Model diagnostics

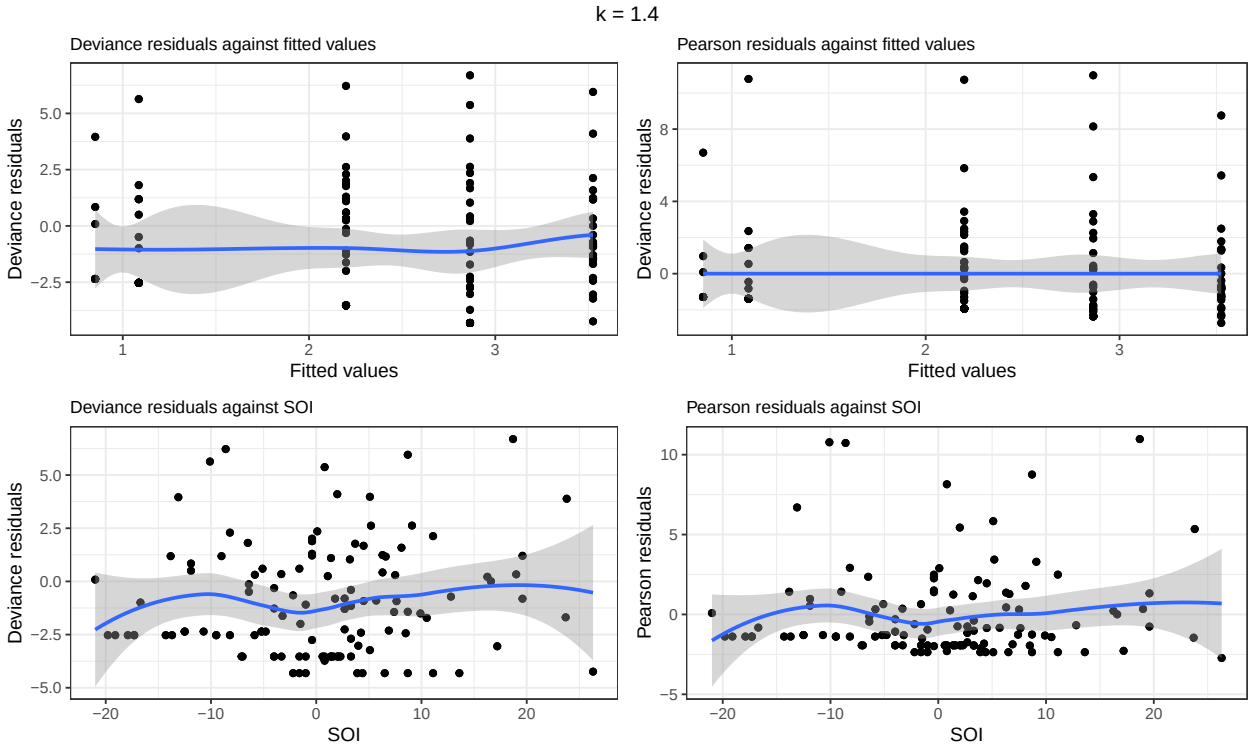
We check the model assumptions for the two Tweedie models fitted in the previous exercise. First we construct a data frame with the relevant diagnostic information for both Tweedie models. We extract the fitted values, Pearson residuals and deviance residuals for both models and add the SOI phase as a variable to the data frame.

Table 22: Table continues below

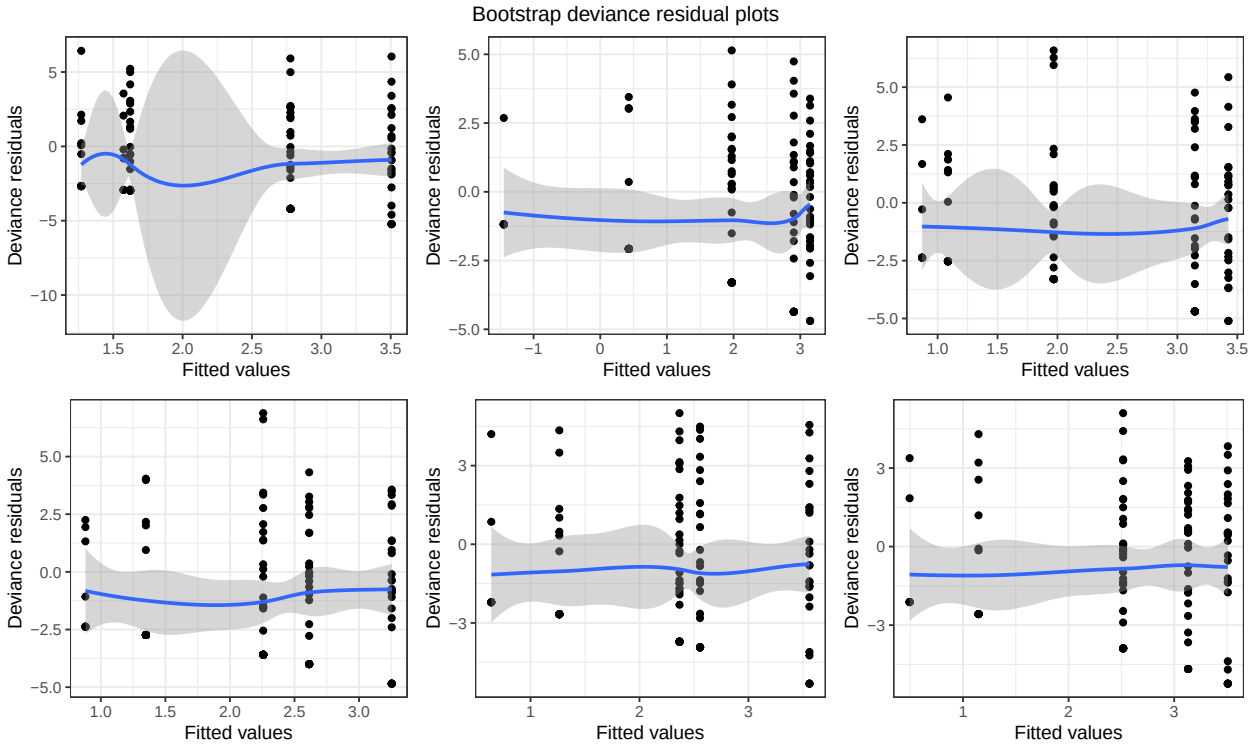
linmod.fitted	linmod.pearson	linmod.deviance	AIC.fitted
Min. :0.852	Min. :-2.729	Min. :-4.3128	Min. :0.852
1st Qu.:2.199	1st Qu.:-1.935	1st Qu.:-2.8228	1st Qu.:2.199
Median :2.199	Median :-1.209	Median :-1.3611	Median :2.199
Mean :2.385	Mean : 0.000	Mean :-0.9051	Mean :2.385
3rd Qu.:2.864	3rd Qu.: 1.010	3rd Qu.: 0.8893	3rd Qu.:2.864
Max. :3.525	Max. :10.977	Max. : 6.6923	Max. :3.525

AIC.pearson	AIC.deviance	SOI
Min. :-2.6120	Min. :-4.2510	Min. :-21.0000
1st Qu.:-1.8830	1st Qu.:-2.7432	1st Qu.: -5.3575
Median :-1.1577	Median :-1.3066	Median : 0.8000
Mean : 0.0000	Mean :-0.9101	Mean : 0.4817
3rd Qu.: 0.9928	3rd Qu.: 0.8718	3rd Qu.: 5.8500
Max. :10.6310	Max. : 6.4126	Max. : 26.3000

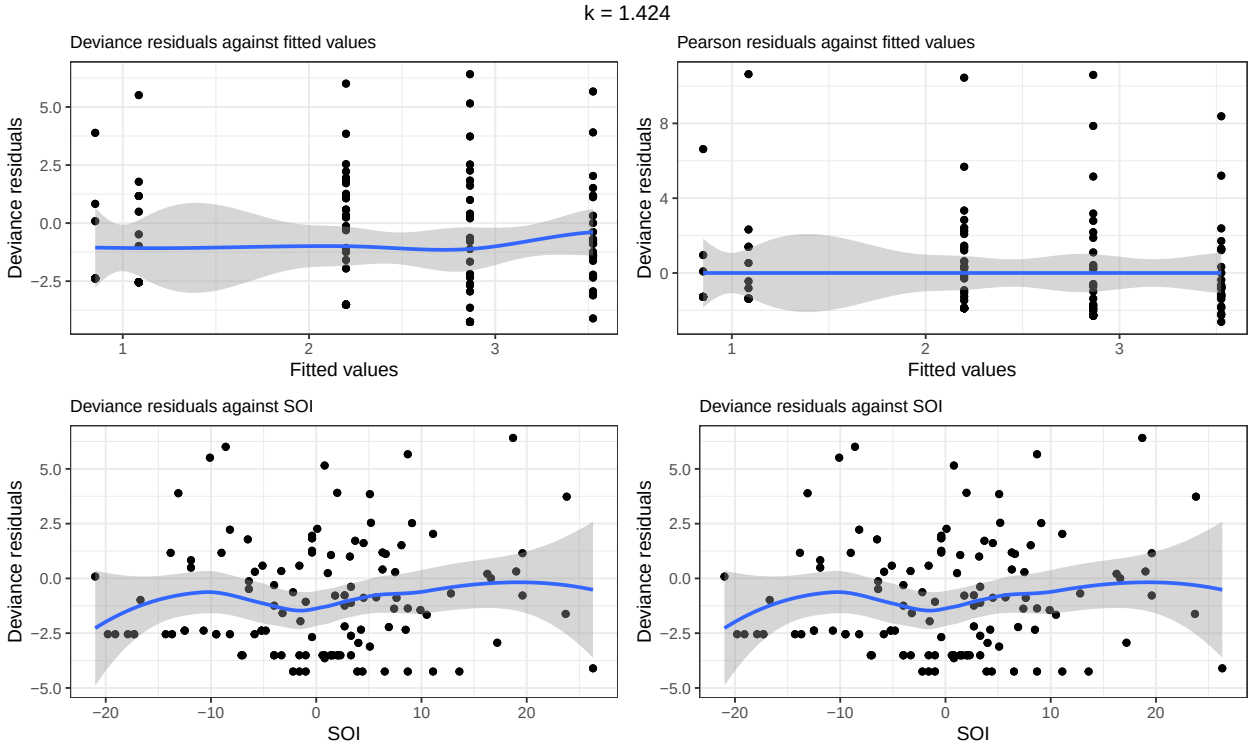
We then plot the residuals against the fitted values, first for the linear model estimate of k:



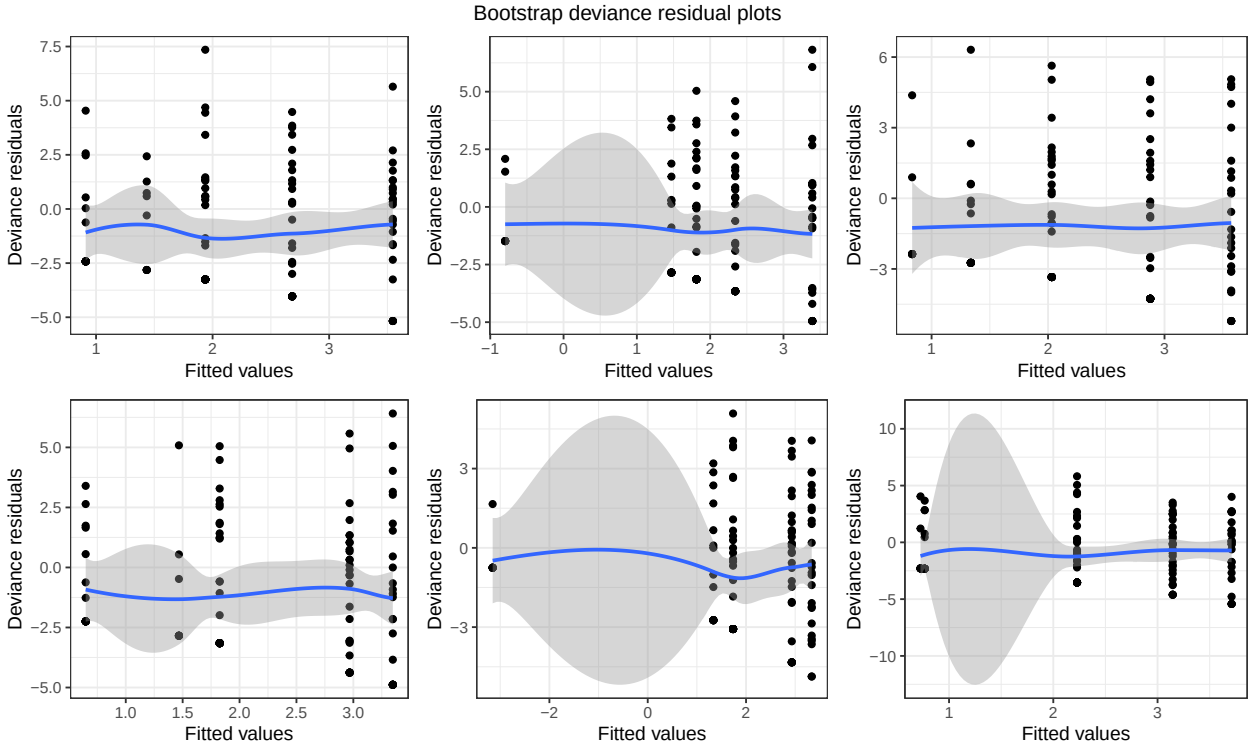
From the plots there is no clear indication of model misspecification. The residuals appear to be randomly scattered around zero, indicating that the model captures the mean and variance structure of the data. We further evaluate the plots with bootstrapping. That is, using the estimated mean, dispersion parameter and k we simulate data from a Tweedie distribution and fit a new Tweedie model to this data. We then plot the residuals against the fitted values. If the original data is from a Tweedie distribution with the estimated parameters, we should observe, that the bootstrapped residual plots resemble the original residual plots.



We see that the plots by and large resemble the residual plots of the original model. This supports the initial diagnostic plots. That is, there is no clear evidence that the model assumptions are violated. We repeat the procedure for the model fitted with the AIC profile likelihood estimated k.



We notice a pattern similar to the previous model and the conclusion is the same.



Again, the pattern appears to be similar to the previous model. There may be concerns with the second and fifth plot where the residual plots diverge slightly from the original plots. It is difficult to say however if this is just noise or if the first estimate of k is better than the second.

Conclusion

The analysis indicates an association between rainfall and SOI phase, a relationship initially suggested by boxplot of rainfall and SOI phase in the EDA and later supported by the models in the subsequent analysis. Based on the model diagnostics, both fitted models seem to adequately capture the mean and variance structure of the data, allowing us to reasonably trust the model conclusions.

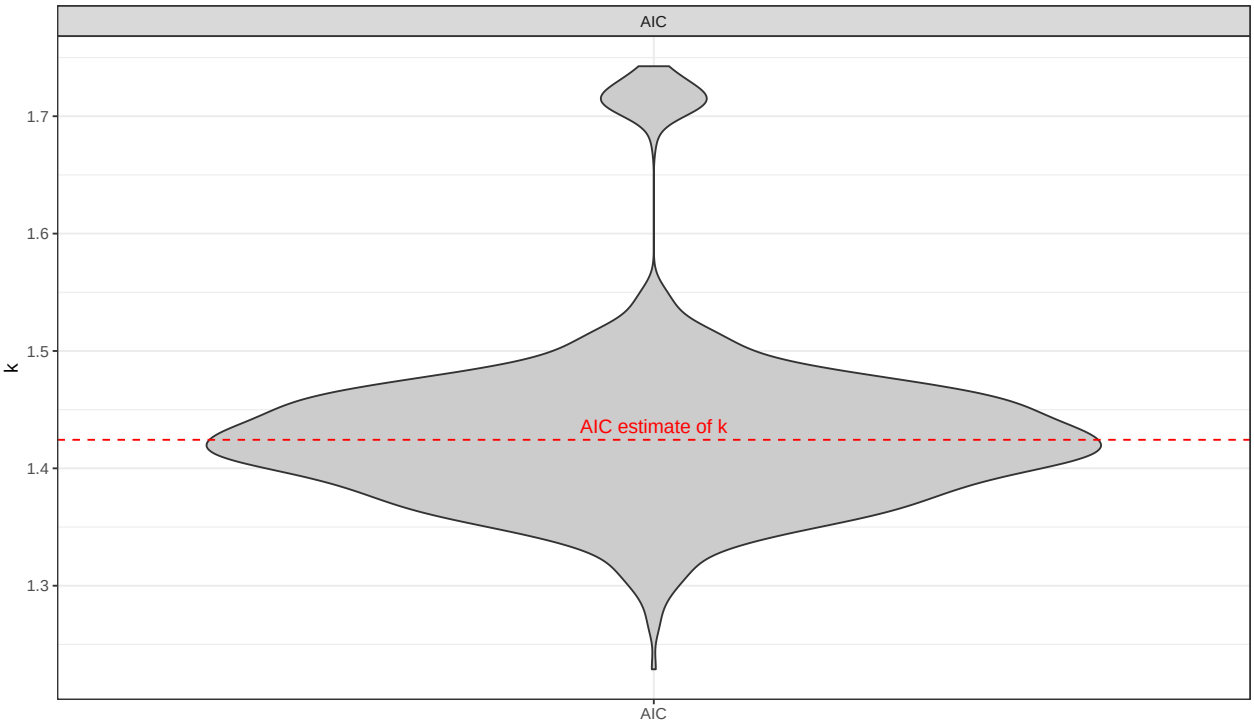
Bootstrap estimates of k

We conclude this part of the analysis with parametric bootstrap to estimate the sampling distribution of k. Using the AIC-based estimate of k and the parameters estimated by the Tweedie model associated to this estimate of k, we simulate data from a Tweedie distribution and fit a new Tweedie model to the simulated data. We repeat this process multiple times to estimate the sampling distribution of k. To carry out the parametric bootstrap we need the strong distributional assumptions GA3 and A5. We have not formally tested these assumptions, so further analysis should check their validity. For now we acknowledge, that the parametric bootstrap may produce too narrow confidence intervals.

Due to the small number of observations in Phase 3 there is a high probability, that some bootstrap samples will contain only zeros in this phase. This implies, that the fitted Tweedie model used in the profile-likelihood will fail to converge. To mitigate this issue, we add a small random value to the observations in the group, in case they are all zeros. While this method introduces some bias, we prefer this method over relying on estimates from a non-converged algorithm which are meaningless.

```
set.seed(10102024)
B <- 1000
parametric_k_AIC <- numeric(B)
```

```r
rain_data_AIC_sample <- Rain.data.comp


for (b in 1:B){
  rain_data_AIC_sample$Rain <- rTweedie(AICmu_hats,
                                        psi_hat_model_AIC,
                                        p = k_hat_AIC)


  # Calculate sum for each phase in order to handle zeros
  rain_data_AIC_sample <- rain_data_AIC_sample %>%
    group_by(Phase) %>%
    mutate(phase_sum = sum(Rain)) %>%
    ungroup()

  # Add small constant to phases where sum of Rain is zero to ensure convergence
  rain_data_AIC_sample <- rain_data_AIC_sample %>%
    mutate(Rain = if_else(phase_sum == 0,
                          Rain + abs(rnorm(n(), 0.0001, 0.001)),
                          Rain))

  tweedieBootstrap.AIC_profile_likelihood <- function(k) {
    profile_likelihood(form = Rain ~ Phase,
                       family = tweedie(var.power = k, link.power = 0),
                       data = rain_data_AIC_sample,
                       eval = AICtweedie)
  }

  boot_k_hat <- optimize(tweedieBootstrap.AIC_profile_likelihood,
                         lower = 1.1, upper = 1.9)$minimum

  parametric_k_AIC[b] <- boot_k_hat
}
```



It appears that the AIC method is fairly robust to changes in the bootstrapped samples with the exception of the spike in the plot that comes from the samples where Phase 3 has only zero observations. We use the parametric bootstrap to estimate the standard error of the AIC estimates of k. The standard deviation is estimated to be

```
se_AIC <- sd(parametric_k_AIC)
pander(se_AIC)
```

0.07367 and the lower and upper bounds of the 95% confidence interval are

```
pander(k_hat_AIC + c(-1, 1) * 1.96 * se_AIC)
```

1.28 and *1.569*

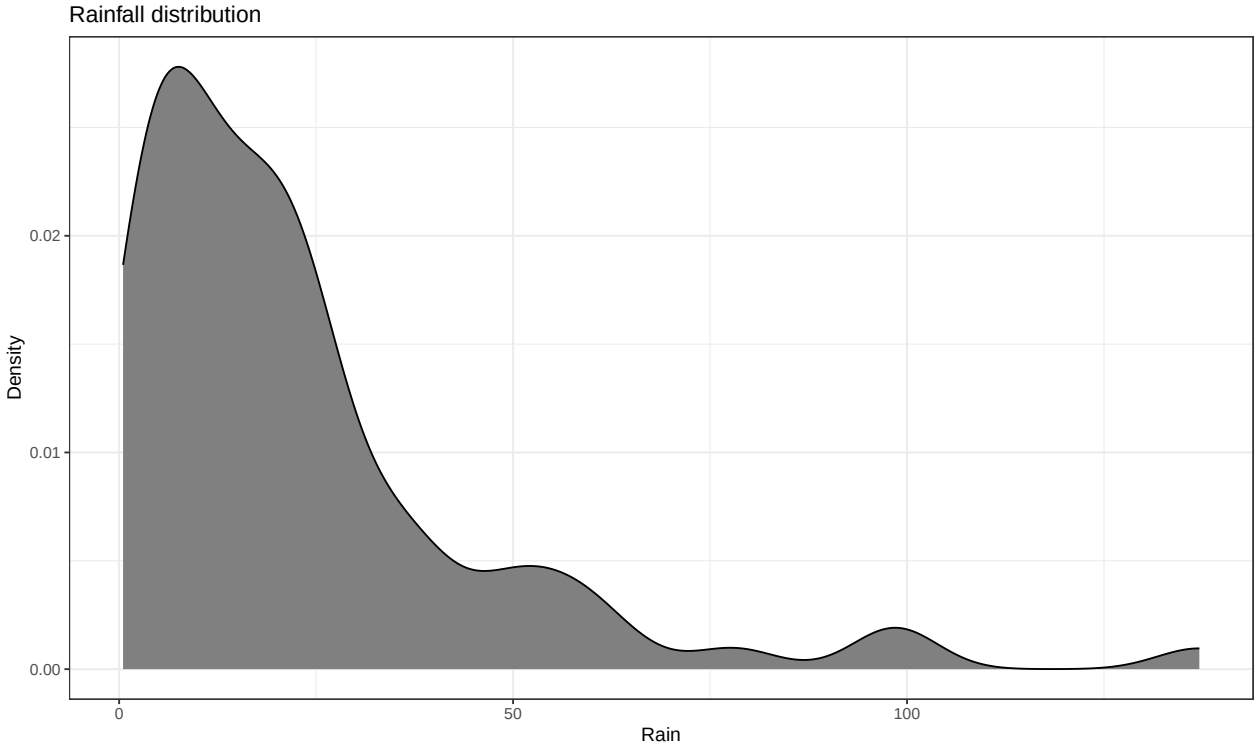
Analysis using SOI directly

We choose to model the rainfall conditionally on having rained. While a Tweedie model could incorporate observations with zero rainfall, this model is not directly comparable to one excluding zero rainfall so to simplify the following analysis and avoid repetition, we chose the latter model. In a more comprehensive analysis, both models should be considered, along with a method for comparing them.

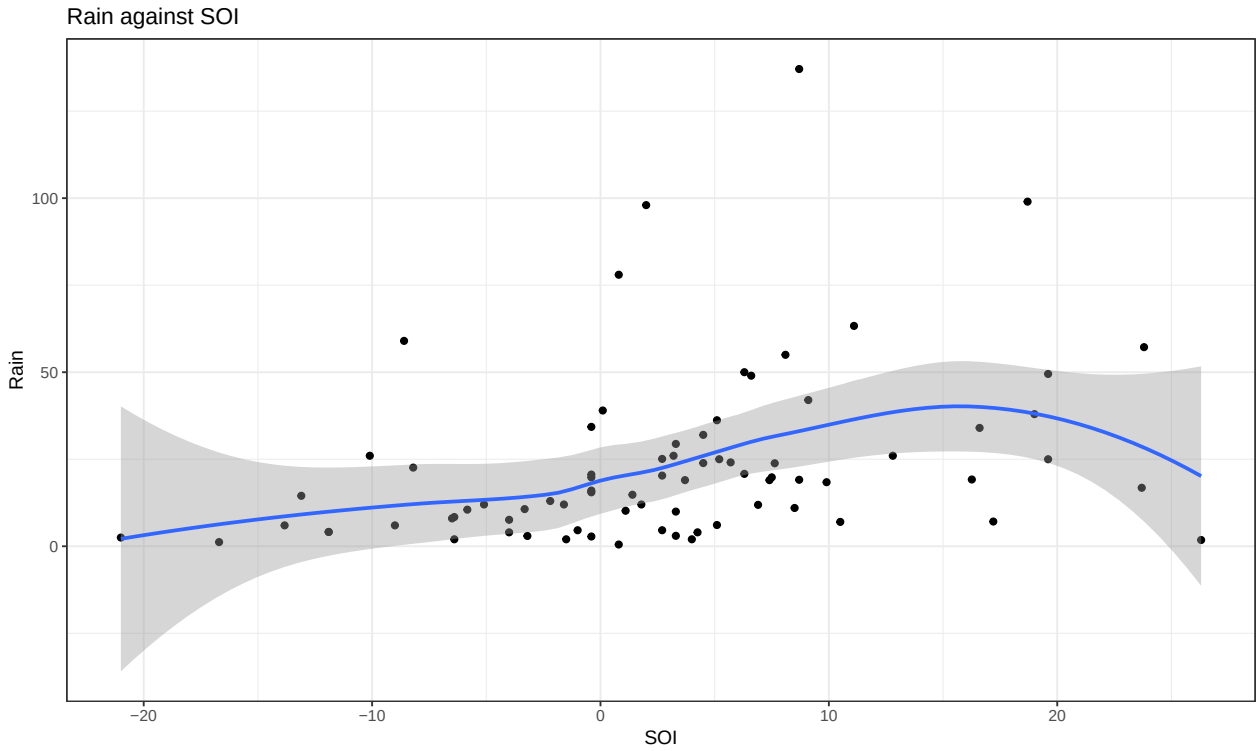
We filter out observations where the rainfall is zero:

```
rain.data1 <- Rain.data.comp %>%
  filter(Rain != 0)
```

We look at the distribution of `Rain`.



The distribution remains highly right skewed. Consider now `Rain` plotted against `SOI`.



There seems to be a positive relationship between `SOI` and `Rain`. The above plots motivates the use of a Gamma exponential dispersion model, which is typically used to fit positive continuous right skewed data. The Gamma exponential dispersion model has a quadratic mean variance relationship, which also seems like a good fit from the scatter plot. The scatter plot also hints at potential non-linear trends. There are a few observations with very large values of `Rain`, but none of which appear extreme enough to be considered outliers.

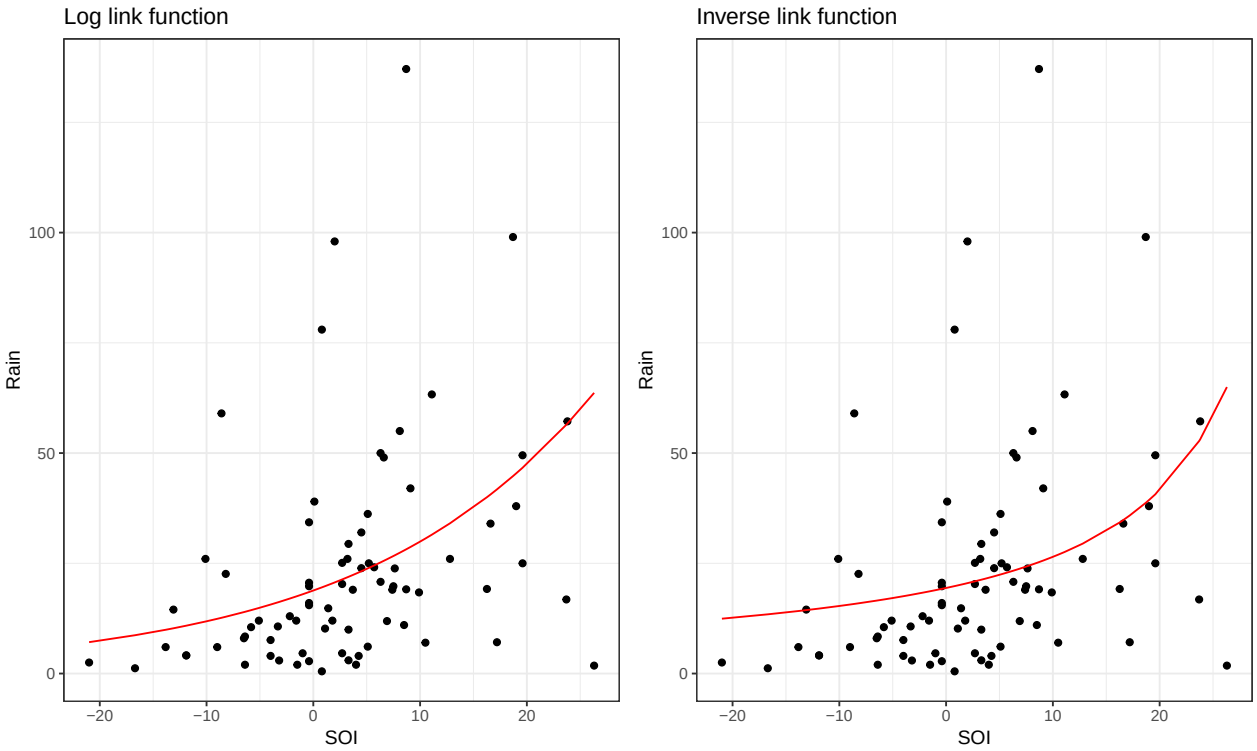
A typical link function choice for the Gamma model is the log link function. If we fit the model using the log link function, we fit the log of the mean of the response variables as a linear combination of the predictors. Other options include the identity link and the canonical link, which is the inverse function. With the identity link it is likely that the model predictions lie outside the support of the gamma-model, i.e. are negative, which is not possible for rain fall (we have observed this when we bootstrapped a model with identity link). This causes issues when we bootstrap this model and for this reason the identity link is disregarded. We therefore fit two models: One with log link and one using the canonical link.

```
glm_gamma_log <- glm(Rain ~ SOI, data = rain.data1, family = Gamma("log"))
glm_gamma_inv <- glm(Rain ~ SOI, data = rain.data1, family = Gamma)
```

Since the models have the same complexity, we can compare them using the training error. The training error based on the squared deviance loss function for the two models is

Link function	Training error
Log	0.8825
Inverse	0.9217

The log link function seems to be the best fit in terms of training error. We plot the model fits



Both models seem to fit the data reasonably well. There is no evidence of overfitting so assume the training error would generalize well. Based on this, we choose to proceed with the the log link gamma model rather than the inverse link model.

Additional predictors

We perform an LRT test to see whether or not we should include additional predictors. We consider to add the predictors `Phase` and `Year`. In the EDA we observed that `Phase` and `SOI` are correlated, and it could be problematic to include both in the model. However, we also discussed that `Phase` contains information that `SOI` does not and vice versa, meaning that the variable could potentially improve our predictions.

```
add1(glm_gamma_log, Rain ~ SOI + Phase + Year, test = "LRT") %>% kable()
```

	Df	Deviance	AIC	scaled dev.	Pr(>Chi)
	NA	68.83829	639.2623	NA	NA
Phase	4	63.70767	642.2971	4.9651410	0.2908930
Year	1	68.22435	640.6681	0.5941376	0.4408236

According to the LRT test there is not evidence in data that suggest that the additional predictors should be added to the model. We therefore choose to proceed with the model that only includes `SOI` as a predictor.

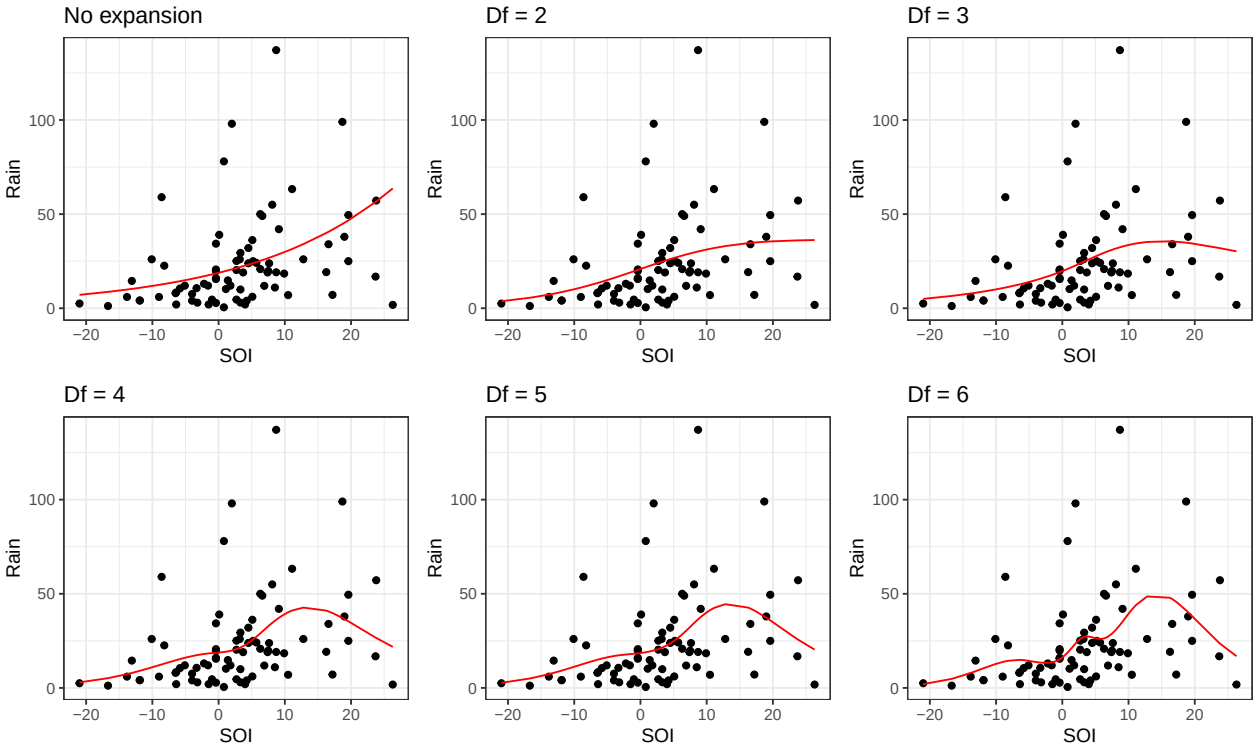
Nonlinear effects

We consider to include a non-linear effect of SOI. In particular we explore the inclusion of a natural cubic spline with 2,3,4,5 or 6 degrees of freedom:

```
form1 <- Rain ~ SOI
form2 <- Rain ~ ns(SOI, df = 2)
form3 <- Rain ~ ns(SOI, df = 3)
form4 <- Rain ~ ns(SOI, df = 4)
```

```r
form5 <- Rain ~ ns(SOI, df = 5)
form6 <- Rain ~ ns(SOI, df = 6)

glm1 <- glm(form1, data = rain.data1, family = Gamma("log"))
glm2 <- glm(form2, data = rain.data1, family = Gamma("log"))
glm3 <- glm(form3, data = rain.data1, family = Gamma("log"))
glm4 <- glm(form4, data = rain.data1, family = Gamma("log"))
glm5 <- glm(form5, data = rain.data1, family = Gamma("log"))
glm6 <- glm(form6, data = rain.data1, family = Gamma("log"))
```

The model fits:



Adding more degrees of freedom to the natural cubic splines adds flexibility to the model, allowing it to fit data better. This comes at the expense of potential overfitting. The model fitted with 6 degress of freedom is quite likely overfitting data. In order to better assess the models in terms of predictions, we do cross-validation to compare the models. We first define the deviance loss, which we will use as error function.

```r
# Error function
dev_loss <- function(Y, muhat) 2 * (log(muhat / Y) + Y / muhat - 1)
```

We define the cross validation function.

```r
cv <- function(data, form, B = 1, k = 8, my_family, error_func){
  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, data = data[group != i, ], family = my_family)
      muhat <- predict(modelcv, newdata = data[group == i, ], type = "response")
```

```
      # !!! change input of error function !!!
      tmp[group == i] <- error_func(data$Rain[group == i],  muhat)
    }
    PEcv[[b]] <- tmp
  }
  mean(unlist(PEcv))
}
```

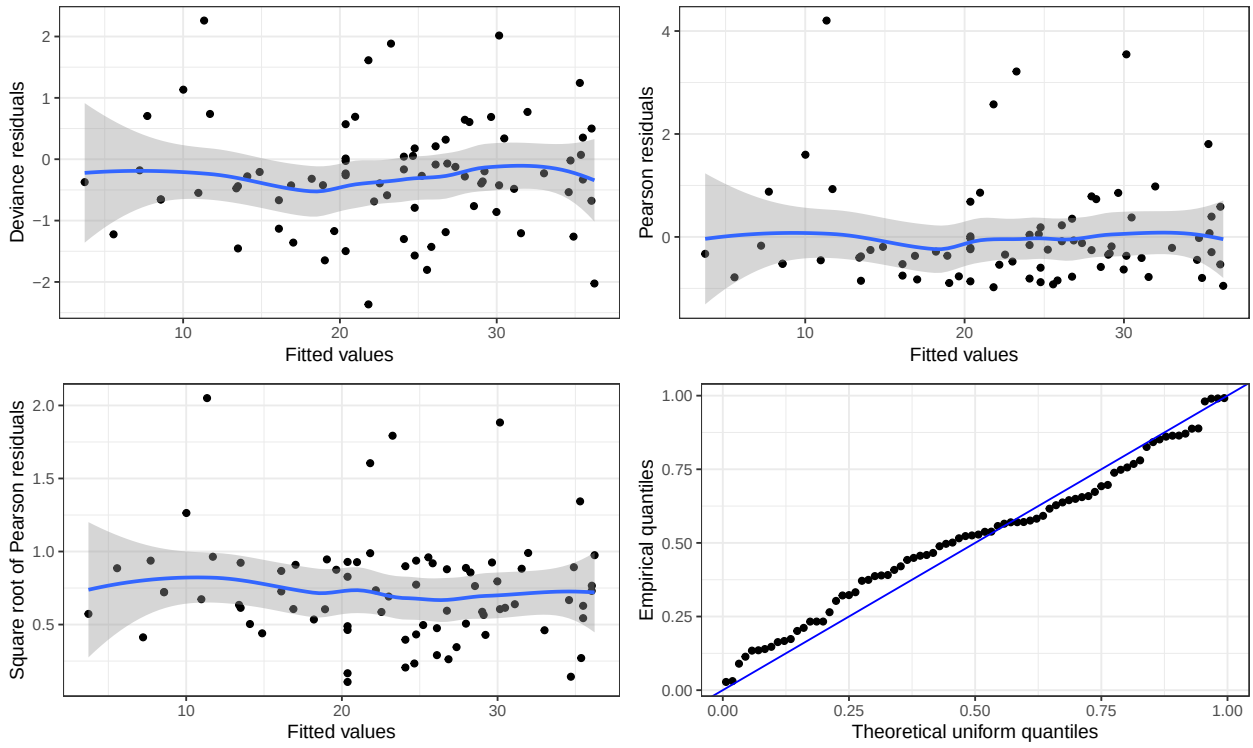
Since the data set is quite small, we perform LOOCV. This is a non random procedure and we therefore set $B = 1$.

Df.	Generalization error
1	0.9393
2	0.9382
3	0.9729
4	0.9915
5	1.02
6	1.036

We proceed with the model with 2 degrees of freedom as it has the smallest cross validation error. But note that the model without natural cubic splines is very close to performing just as well in terms of generalization error. Since we are interested in prediction, we choose the model with 2 degrees of freedom. If interpretability was of higher priority, we would choose the model without natural cubic splines.

Model Diagnostics

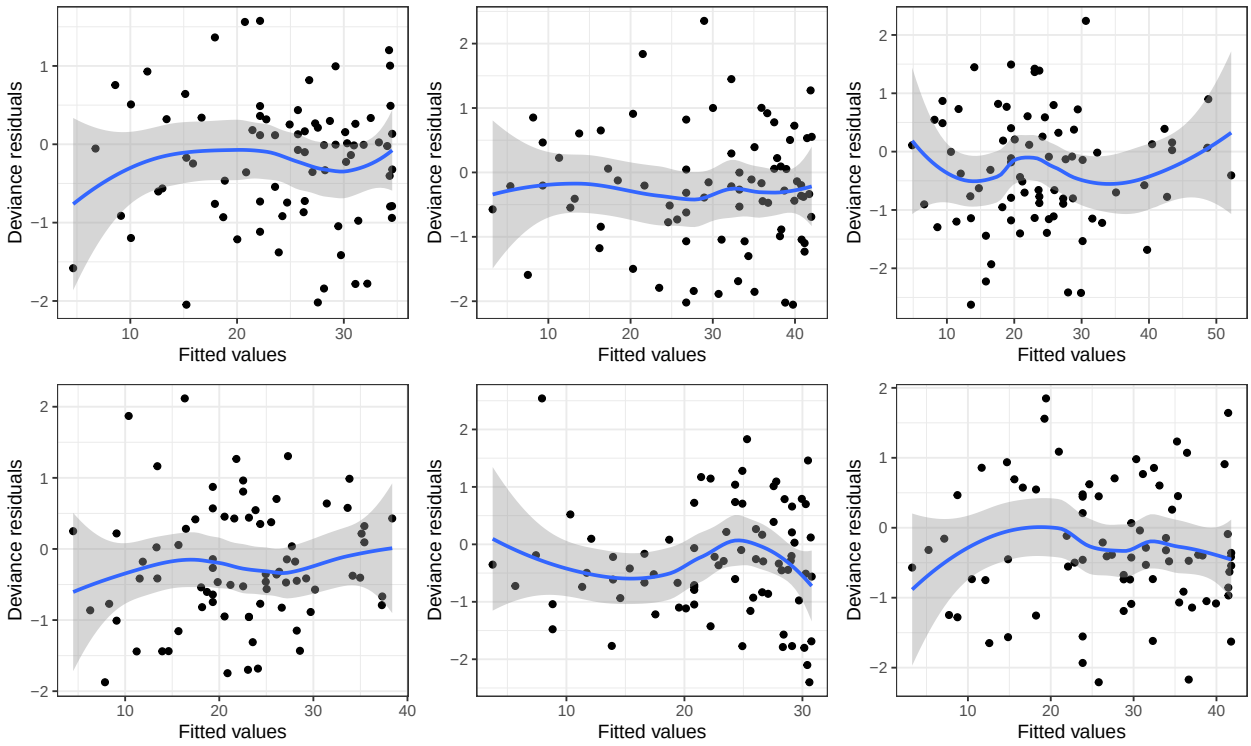
We do model diagnostics for the chosen model.



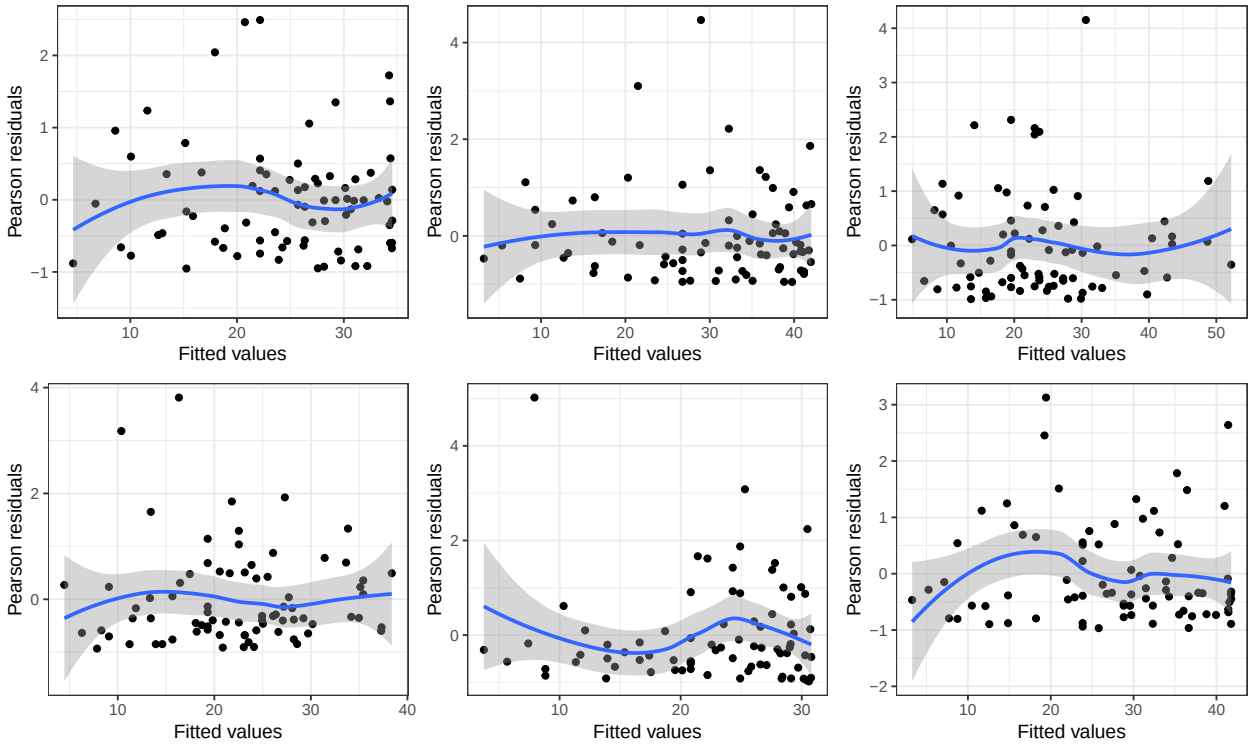
From the plots we see no clear evidence against the weak model assumptions GA1 and GA2. The residuals seem to be randomly scattered around zero with constant variance. There are a few large residuals, but they are not alarming. As the PP-plot displays, the stronger distributional assumption GA3 seem to be slightly off which implies that inference relying on distributional results should be avoided or employed with consideration.

We evaluate the residual plots via bootstrapping. We compare the residuals with simulated residuals under the null hypothesis that our model is correct. We repeat the bootstrap procedure 6 times, and plot the deviance residuals against fitted values and the Pearson residuals against fitted values for each bootstrap sample.

The bootstrapped deviance residual plots



The bootstrapped Pearson residual plots



The bootstrapped residual plots resemble the original residual plots well. This further indicates that GA1 and GA2 are satisfied. Note how the few large residuals that we observed in the original residual plots are also present in the bootstrapped plots supporting the claim that they are not alarming.

Reporting a final model and interpretation

The summary of our final model

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.314	0.5208	2.523	0.01374
ns(SOI, df = 2)1	3.775	1.04	3.629	0.0005163
ns(SOI, df = 2)2	1.263	0.5099	2.476	0.01554

(Dispersion parameter for Gamma family taken to be 1.014447)

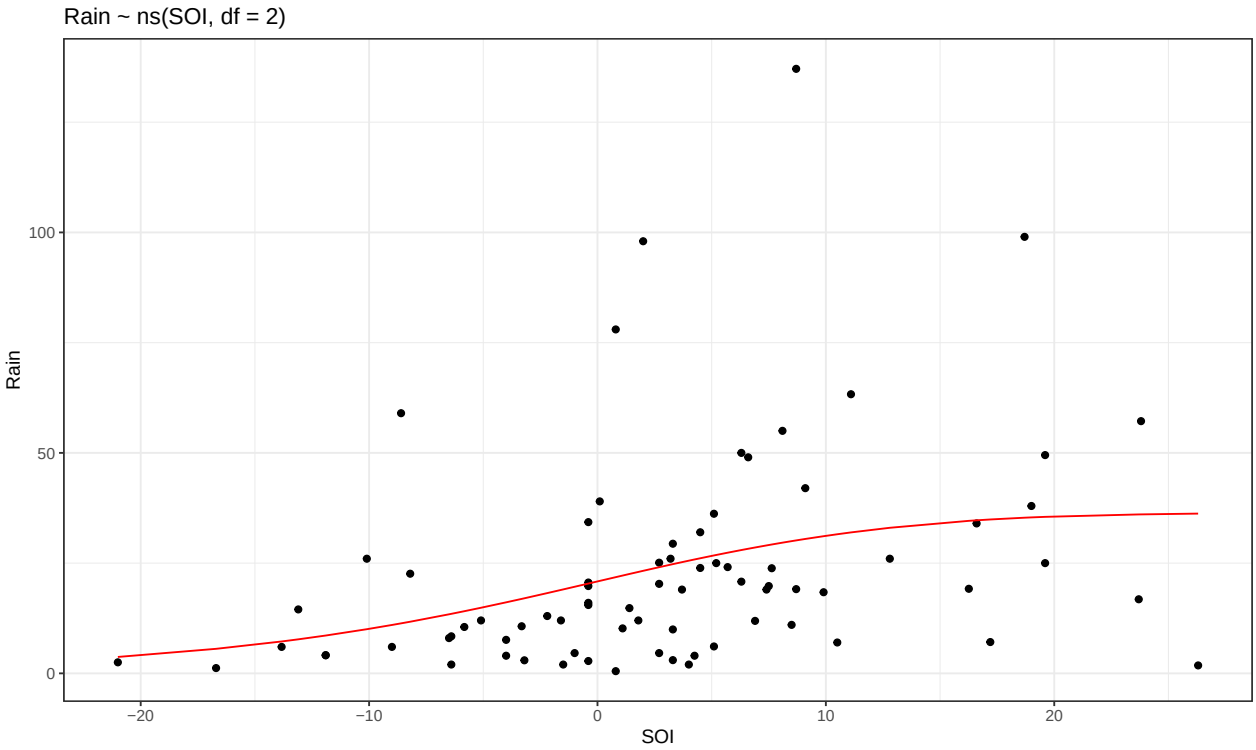
Null deviance:	80.67 on 77 degrees of freedom
Residual deviance:	66.89 on 75 degrees of freedom

From the summary we can conclude that `SOI` is a significant predictor of `Rain`. Since we have used a natural cubic spline with 2 degrees of freedom to fit our model, the coefficients are difficult to interpret. We instead consider the predictions of the model. We print the model predictions for a few values of SOI:

SOI	Rain Fall
-20	4.088
-10	10.1
0	20.85
10	31.18

SOI	Rain Fall
20	35.57

A plot of the model fit is shown below.



The fitted model predicts that rainfall is increasing as a function of SOI. The slope of the model is largest for values of SOI between −10 and 10. For SOI values that are larger or smaller than this, the model is more constant.

We will now turn to the construction of confidence intervals for our model. First we will use nonparametric bootstrap to create a combinant based confidence interval for the model as described on page 220. The code used for the pair sampling can be seen below

```
# Pair sampling
B <- 1000
set.seed(170)
n <- nrow(rain.data1)
boot_pred <- matrix(nrow = n, ncol = B)

for(b in 1:B){
  boot_samp <- sample(n, replace = TRUE)
  boot_mod <- glm(Rain ~ ns(SOI, df = 2),
                  data = rain.data1[boot_samp, ],
                  family = Gamma("log"))
  boot_pred[,b] <- predict(boot_mod, newdata = rain.data1, type = "response")
}

CIs <- matrix(nrow = n, ncol = 2)
for(i in 1:n){
  CIs[i,] <- 2*pred2[i] - quantile(boot_pred[i,], probs = c(0.975, 0.025), na.rm = TRUE)
```

```
}

p1 <- qplot(rain.data1$SOI, rain.data1$Rain) +
  geom_line(aes(y = pred2), color = "black") +
  geom_ribbon(aes(ymin = CIs[,2], ymax = CIs[,1]), alpha = 0.3) +
  xlab("SOI") +
  ylab("Rain") +
  ggtitle("Pair sampling comb. based CI")
```

We will further construct confidence intervals of the form

$$\hat{f} \pm 1.96\hat{\text{se}}$$

Note that these intervals will be symmetric around the point estimate. First we will use residual sampling to estimate standard errors of the model predictions and use these to construct the confidence interval.

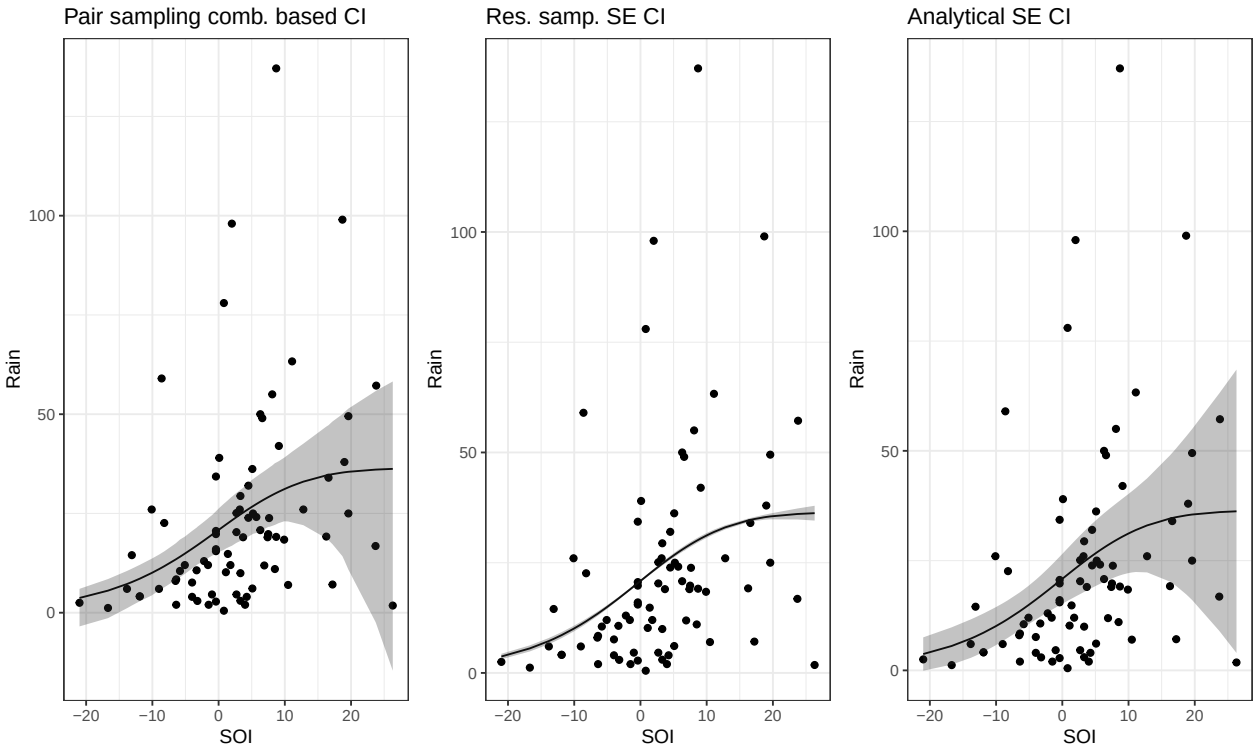
```
# Residual sampling
set.seed(170)
mus <- mod_final$fitted.values
res <- mod_final$residuals
boot_pred2 <- matrix(nrow = n, ncol = B)

for(b in 1:B){
  boot_samp <- sample(n, replace = TRUE)
  boot_data <- data.frame(Rain = mus + res[boot_samp], SOI = rain.data1$SOI)
  boot_mod <- glm(Rain ~ ns(SOI, df = 2),
                  data = boot_data,
                  family = Gamma("log"))
  boot_pred2[,b] <- predict(boot_mod, newdata = rain.data1, type = "response")
}

SEs <- numeric(n)
for(i in 1:n){
  SEs[i] <- sd(boot_pred2[i,])
}

CIs2 <- cbind(pred2 - SEs*qnorm(0.975), pred2 + SEs*qnorm(0.975))

p2 <- qplot(rain.data1$SOI, rain.data1$Rain) +
  geom_line(aes(y = pred2), color = "black") +
  geom_ribbon(aes(ymin = CIs2[,2], ymax = CIs2[,1]), alpha = 0.3) +
  xlab("SOI") +
  ylab("Rain") +
  ggtitle("Res. samp. SE CI")
```

A last confidence interval we will consider is created in the same way as above, but using analytical standard errors of the model predictions.

```
SE_an <- predict(mod_final, newdata = rain.data1, type = "response", se.fit = TRUE)$se.fit
CIs3 <- cbind(pred2 - SE_an*qnorm(0.975), pred2 + SE_an*qnorm(0.975))

p3 <- qplot(rain.data1$SOI, rain.data1$Rain) +
  geom_line(aes(y = pred2), color = "black") +
  geom_ribbon(aes(ymin = CIs3[,2], ymax = CIs3[,1]), alpha = 0.3) +
  xlab("SOI") +
```

```
  ylab("Rain") +
  ggtitle("Analytical SE CI")
```

We will now compare the three confidence intervals.



The first thing that catches the eye is that the confidence interval based on residual sampling SE estimates is extremely narrow, which seems unlikely. Apart from this, we see that the two other confidence interval are narrow for the `SOI` values where we have many observations and wide for the `SOI` values for which we have few observations. This is to be expected. A last thing to point out is the asymmetry that is present in the bootstrap combinant based confidence interval, indicating a certain asymmetry of the distribution of the model predictions.

Conclusion

Because of the right skew of data we decided to use the Gamma model to fit the data. We chose the log link as it ensures that the model predictions are kept within the domain of the distribution and since it performed well in terms of training error. We chose not to include further predictors as they were insignificant according to the LRT test.

The model we have fitted is a Gamma model with a log link function and a natural cubic spline with 2 degrees of freedom. This was the model with the smallest generalization error chosen by cross validation, where we considered models fitted on natural cubic splines with degrees of freedom ranging from 1 to 6.

The model is well fitted to the data and the residuals show no evidence against the model assumptions.

According to the model `SOI` is a significant predictor of `Rain`. The model predicts that `Rain` is increasing as a function of `SOI`. We have constructed confidence intervals for the model predictions, which show that the model is most certain for `SOI` values for which we have many observations, and less certain for `SOI` values where we have few observations.