An analysis of the Old Faithful Geyser

End of term report for the lecture
Computational Statistics

Tim Adler

January 28, 2016

1. Introduction

In this report we want to analyse the eruptions of the Old Faithful Geyser. This is a geyser at the Yellowstone National Park, which is known for its predictability, hence the name 'faithful'.

We will use the data used by Azzalini & Bowman [1], which was measured from August 1, 1985 to August 15, 1985. The measurements were taken manually an example page of the notes can be found in [1, p. 2].

The data set consists of two variables with overall 299 measurements. The types of the variables can be found in Table 1. For details please consult Appendix A.1.

Variable	Name	Type	Comment
1	duration	numeric	Eruption duration in min
2	waiting	numeric	Waiting time since last eruption in min

Table 1: Structure of the geyser data set

From the handwritten notes, we take that not all duration measurements were done thoroughly at night times only a classification was given ('short', 'medium', 'long'). Azzalini & Bowman chose to translate those to 2 min, 3 min and 4 min. In this analysis we use an implementation of the geyser data set which is given by the R framwork. Explicitely we use R version 3.2.3 (2015-12-10) on a x86-64-pc-linux-gnu machine and the geyser data set given by the `MASS` library (Version 7.3-45). In this implementation the translation is already incorporated. Figure 1 shows lines at the appropriate durations. We chose to leave the data points in as they do not seem to influence the overall shape of the distribution that much. Additionally we do not want to describe the data set perfectly, but merely want a predictive model.

As stated, in this analysis the main focus will lie on the task of finding a suitable predictive model for the waiting time until the next eruption. Before 1998 the waiting time was predicted using a linear model only taking the last eruption duration into account. According to Cook &

Weisberg [2] this function is given by

$$(\text{next waiting time [min]}) = 30 + 10 \cdot (\text{duration [min]}). \tag{1}$$

This graph can be found in Figure 1 (left, violet line) and we can use it to determine the requirements for our predictive model. The central one is, that we do not so much want to describe the data set perfectly, but instead focus on minimzing the number of missed eruptions while on the same time trying to have a bearable additional waiting time if our model predicts 'too short'.

To achieve this goal we will analyse the shape of the data and propose a proper framework for our model to translate our requirements into statistics.

2. Cluster detection

We have two variables, the (last) waiting time and the duration, which we want to use to predict the next waiting time. Figure 1 shows that our data points cluster into three clouds. These clouds are color coded. We see that there is no obvious linear dependence between any of the two plotted relations. The official predictive function just seems to run parallel to the line through the two centers of duration plot clouds.

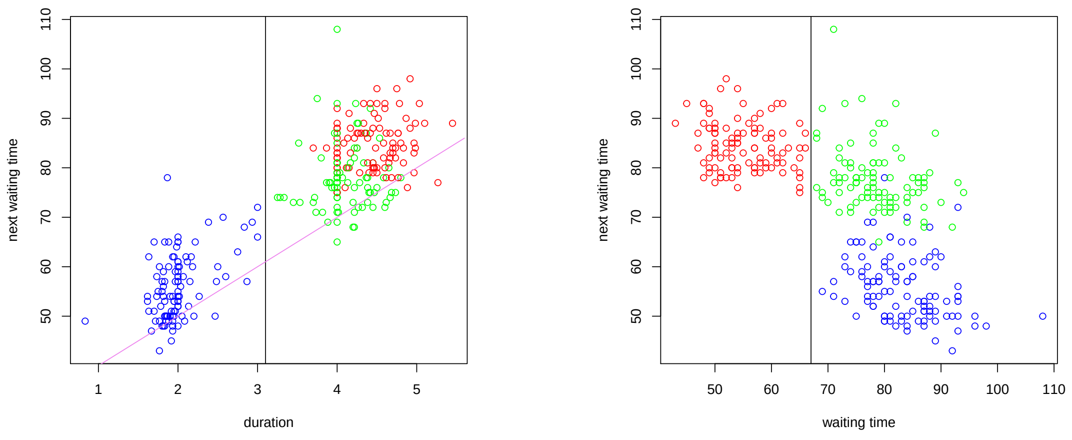

Figure 1: Color coded marginal scatter plots with separators and park model.

We separate the three clouds by the two vertical lines depicted in Figure 1. The duration separator is at 3.1 and the wating time separator is at 67. This is not a perfect seperation as some points of the blue grouping mix with the green grouping and vice versa. Also the exact location of the waiting time split is up to some arbitration. It turns out that this is not a problem, because only one model (and not the best one) uses this separation. There are no other clusters in this data set as can be seen in Figure 2 in Appendix A on page 5.

Azzalini & Bowman [1] state a rough physical model for the evolution of the waiting time, which involves a periodic process with period 2. It does not seem likely to us that we have such a simple periodicity. The details can be found in Appendix A.2.

3. Model Framework

We had a look at four different models and compared their predictions to the official predictive model. For that we set up our own loss function as depicted in Figure 6 on page 8. The procedure to determine its parameter can be found in Appendix B. The four investigated models are:

1. A logistical interpolation of the two clouds shown in Figure 1 (left).
2. A piecewise linear regression separating the two duration clouds.
3. A linear regression taking the duration as well as the last waiting time into account.
4. A piecewise linear regression separating the three clouds in Figure 1 taking the duration and the last waiting time into account.

The exact implementations of the fitting algorithm and the models is described in Appendix C and D. We also used cross validation to determine the predcitive total loss of our models (c.f. Appendix C). The results can be found in Table 2. The residuals of each of the models together with their fitted parameters are stated in Appendix D.

	Number of Param	Fitted Total Loss	Predictive Total Loss	Average Misses [%]	Average Additional Wait [min]
Official	2	4053.83	4053.83	10.07	8.75
Logistic Interpolation	2	3445.20	3488.03	2.35	10.47
Pw Linear (dur)	4	3431.46	3548.98	3.02	10.10
Linear (dur+wait)	3	3533.47	3873.74	1.68	11.09
Pw Linear (dur+wait)	9	3184.27	3354.90	3.36	9.08

Table 2: Summary of the models

4. Conclusion

In conclusion we see that differentiating between the different clusters strongly improves the models. Model 3 (the only one not using clusters) can be discarded immediately. Its predictive total loss is by far the worst, because of the lack of used structure. The last model has the best predictive loss and smallest average waiting time, however the average miss ratio is slightly worse than for the other two. Together with the high amount of parameters necessary I would advise against its use. Perhaps in a later analysis one could consider a piecewise linear model with duration and waiting time as predictors, but only taking the duration separation into account. This would reduce the number of parameters to 6. As a standard choice I would recommend Model 1 as it has the best predictive behaviour, if the additional waiting time is of grave importance one could consider switching to Model 2, although the improvement is nigh negligible.

A. Descriptive Statistics

A.1. Extended summary

In the following section we describe the geyser data set in more detail. In addtion to what was mentioned in Section 1, we want to add that we have to keep in mind that waiting time i of the data set denotes the waiting time *before* eruption i occured. Thus waiting time i incorporates duration $i-1$. We could separate the two. However, as can be seen in Table 3, the duration is much smaller than the waiting time such that the effect on the prediction would be negligible.

	waiting [min]	duration [min]
Min	43	0.83
1st Quantile	59	2
Median	76	4
Mean	72.31	3.46
3rd Quantile	83	4.38
Max	108	5.45

Table 3: Summary of the geyser data set

In Figure 2 we see a 3 dimensional scatter plot of the next waiting time over waiting time and duration. A density estimation has been added and we see, that there are three clouds visible. These correspond exactly to the coloring in Figure 1.

A.2. Periodicity and Recurrence Plots

As Azzalini & Bowman state in [1], they suspect a rough physical model underlying the eruption pattern of the geyser. One prediction of this model were that the waiting time would have a period of 2, i.e. the waiting time would alternate between 'long' and 'short'. Figure 3 (left) seems to back this idea.

To investigate the periodicity of our data set further, we used a recurrence plot as introduced in [3]. Recurrence plots give us a mark at coordinate (i, j) whenever the waiting time at point i is *sufficiently* close to that of j. In our case we compared three consecutive data points starting at i with their counterpart at j and wanted them to be closer together than one standard deviation (13.89), which is a rather lenient restriction.

We see in Figure 3 (right), that we have basically no recurrences at $(i, i+2)$, which we would expect, if we had a periodicity of two. There are quite a few other recurrences, which is not surprising as our closeness constraint is rather lax, however it seems that all in all the period is either too long or too complex to be taken into account in the further analysis.

A.3. Nonparametric and parametric density estimation

Since we want to predict the waiting time it is interesting how the waiting time is distributed overall. For that we utilised a density estimation. We used the `density` method of R together

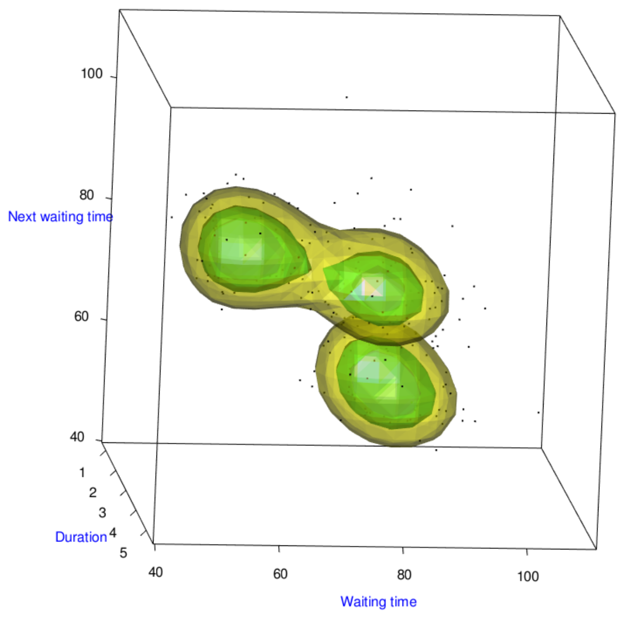

Figure 2: Three dimensional scatter plot with estimated density. We find three clusters corresponding to the clusters marked in Figure 1

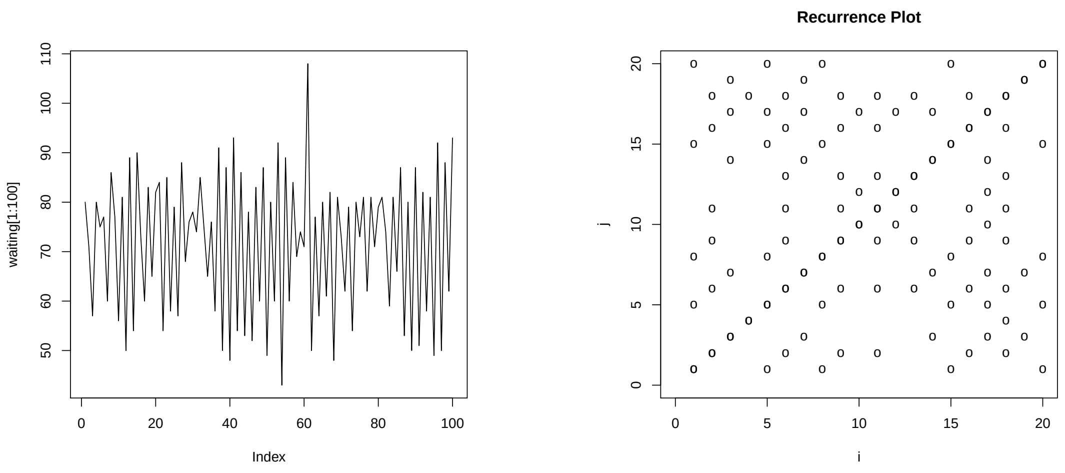

Figure 3: Left: Extract of the waiting time time series. Right: Extract of the waiting time recurrence plot.

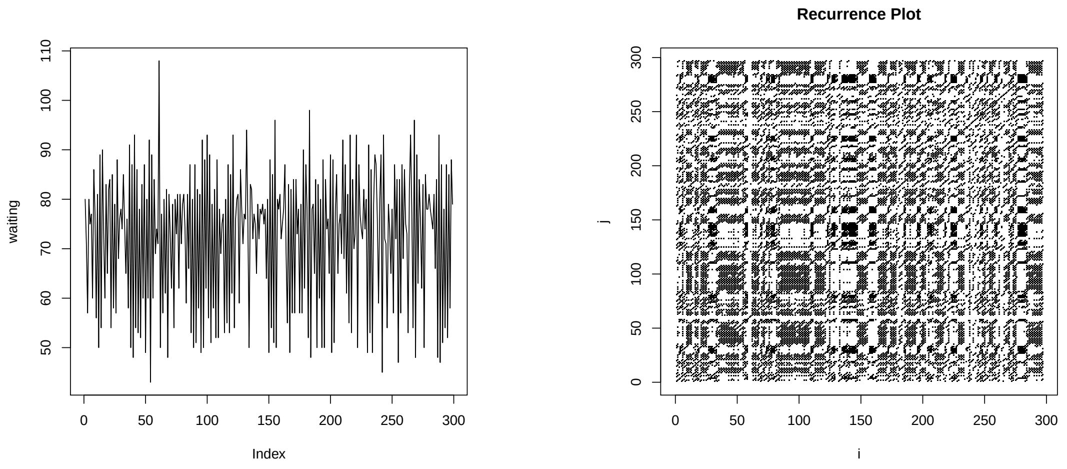

Figure 4: Left: Full waiting time time series. Right: Full waiting time recurrence plot.

with the Sheather-Jones-Algorithm(c.f. [5]) for Bandwidth gauging. The problem is that we wanted to estimate the overall shape without averaging out important structure. To take this into account we also had a look at estimated densities with a far smaller bandwidth and found that the estimated bandwidth is appropriate for our purpose. The result of the estimation can be found in Figure 5 (left, black curve).

We observe that the distribution is strongly bimodal and from the form we deduce that it should be well described by a mixed normal distribution. We did this approximation (blue curve in Figure 5 left) following the procedure described in [6]. Our density candidate is given by

```
> dwait1 <- function(x,p) {
+   p[1]*dnorm(x, mean = p[2], sd = p[3]) +
+      (1-p[1])*dnorm(x, mean = p[4], sd = p[5])
+ }
> dwait1.min <- function(x, p) {
+   e <- dwait1(x,p)
+   if(any(e <= 0)) Inf else -sum(log(e))
+ }
```

We minimize `dwait1.min` using the built in minimizer `optim` of R. As we see in Figure 5 the blue curve is in rather good agreement with the estimated density. However, the left peak is a little off and the overall form of it differs from the nonparametric density.

To remedy this shortcoming we want to introduce (as described in [6]) another parameter and take the duration into account. Looking at Figure 1, it is clear that the duration has a strong impact on the distribution of the waiting time, so it seems sensible to make the density duration dependent. For this we make the 'mixing coefficent' duration dependent. Since this

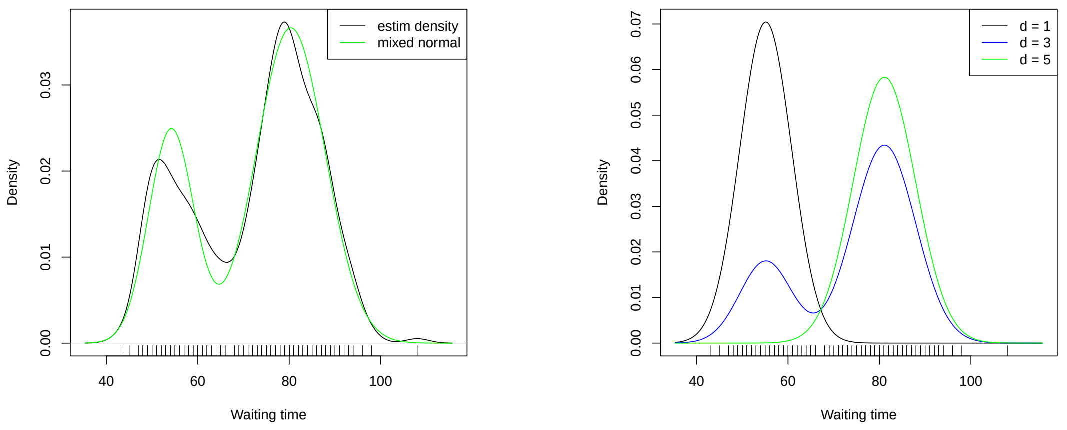

Figure 5: Density estimation of the Waiting time. Left: Duration independet. Right: Duration dependent.

coefficient has to vary between 0 and 1 it is natural to assume a logistic relation to the duration. Implemented this takes the following form:

```
> dwait2 <- function(x,y,p) {
+   q <- exp(p[1] + p[2]*y)
+   q <- q/(1+q)
+   q*dnorm(x, mean = p[3], sd = p[4]) +
+     (1-q)*dnorm(x, mean= p[5], sd = p[6])
+ }
> dwait2.min <- function(x,y,p){
+   e <- dwait2(x,y,p)
+   if(any(e <= 0)) Inf else -sum(log(e))
+ }
```

Analogously to `dwait1` we minimized it using `optim`. We see that for the one additional parameter we get a minimum of 985.32 as compared to 1154.24 for `dwait1`. This is a remarkable improvement. In Figure 5 (right) the waiting time distribution is plotted at different durations. We see that if the duration time is either very high or very low, the bimodal distribution degenerates to a common normal distribution.

Since we now have this very managable duration dependent distribution, we can easily obtain a predictive model for the waiting time. We simply compute the expectation value of the distribution, which depends logistically on the duration. This is the very first predictive model we use as can be seen in Appendix D.

B. The loss function

In order to achieve our goal to find a suitable model to predict the waiting time of the faithful geyser, we need to work with an appropriate loss function.

The standard loss function (i.e. minimizing over the quadratic error) does not seem to be a permissible candidate. A good model should take into account that it is far less annoying for a visitor to wait a couple of minutes at the geyser before the eruption starts than being a few minutes too late. Because of the symmetry of the quadratic error, it cannot distinguish between 'being too early' and 'being late'.

Thus we will use the following loss function

```
> loss <- function(x, q){
+   x[x < 0] <- q
+   x
+ }
```

with parameter q.

As can be be seen in Figure 6, the function attributes a constant high loss, if the prediction is too long and a linear gain of loss, if the predicted waiting time is shorter than the actual waiting time. This fullfills the requirement that a missed eruption weighs heavier than additional waiting time.

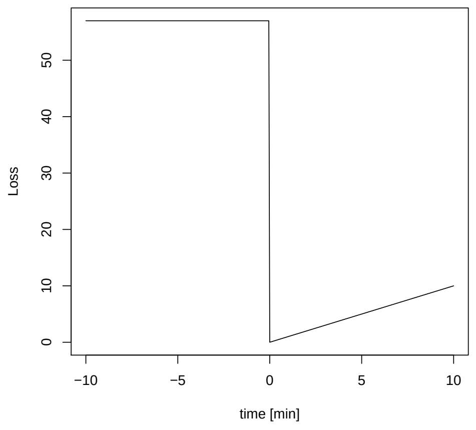

Figure 6: Proposed loss function. Positive time corresponds to a shorter predicted waiting time than the actual waiting time.

The last necessary step is to determine the parameter of this function. At first glance it seems that there are two: The slope of the linear part and the intercept of the constant part. However scaling the function by a (postive) factor does not change the result of a minimization. So we are free to fix the slope to 1 and are left with only the intercept as parameter. In order to determine it, we use the official predictive function of the National Park (as mentioned in [2], c.f. Equation (1) on page 2).

The procedure to determine our parameter q now consists in the following idea: If we have the right q and we fit a linear model using this loss function we want to get the official park function. This idea is sensible, as we can assume that there certainly was a bit of optimization (and rounding) involved in the determination of the park function and we thus can make sure that we roughly apply the same error margin as the park.

Therefore the (manually) applied algorithm consists in systematically choosing different values for q and then fitting a linear model. Aftewards we check the intercept and slope. If both are 'close enough' to 30 respectively 10, we found our parameter.

The fitting procedure used is described in Appendix C. To make sure that we end up in the right local minimum, we choose 30 and 10 as the inital parameters for the fit of the linear model.

For $q = 57$ we yield fit parameters 29.95 and 9.02. We also see that the models are rather robust with respect to small changes in q. So $q = 57$ seems an acceptable choice. q lies in a sensible region, too. A short sanity check: With that choice it is as bad to miss an eruption (and thus having to wait another average waiting time) as it is bad to have to wait another average waiting time (c.f. Table 3 on page 4) to the next eruption after the predictive model said it should be. This sounds reasonable.

With this we get the total loss of the predictive park function as 4053.83. This is from now on our margin to top for alternative models.

C. The fit and cross validation routine

If we are given predictors $x := (x_i)_{i=1,\dots,n}$ and regressors $y := (y_i)_{i=1,\dots,n}$ and a model function f_p with parameter(s) p. We want to choose the parameter(s) in a manner such that the overall loss of the prediction is minimal. That means we want to solve the minimisation problem

$$\min_p g(p, x, y) := \min_p \sum_{i=1}^{n} \operatorname{loss}(y_i - f_p(x_i)).$$

In R this problem takes the following form: The method `genmin` takes the function f_p and generates the function g we want to minimize over. The method `lossfit` takes f_p internally generates g and then uses the R intern minimizer `optim` with the Nelder-Mead-Algorithm (c.f. [4]) and initial parameter p_0 to estimate the parameter p. In code this looks as follows:

```
> # takes a function fn(x,p) and generates a function to minimize over.
> genmin <- function(fn, l = loss, bound = q) {
+   function(p,x,y) {
+     result <- 0
+     if(is.atomic(x)) {
+       for(i in 1:length(y)) {
+         result <- result + l(y[i] - fn(x[i], p), bound)
+       }
+     } else {
+       for(i in 1:length(y)) {
+         result <- result + l(y[i] - fn(x[i,], p), bound)
+       }
+     }
+     #sum(loss(y - fn(x,p), bound))
+     result
+   }
+ }
> # fitting/minimization taking place here
> # p0: initial parameter
> # x: all predictors (as data.frame)
> # y: the (1d) regressor
> # l: custom loss function
> # bound: bound of the loss function
> lossfit <- function(p0, fn, x, y, l = loss, bound = q){
+   fn.min <- genmin(fn, l = l, bound = bound)
+   optim(p0, fn.min, x = x, y = y)
+ }
```

As any numerical optimizer/minimizer, `optim` only finds local minima and therefore p critically depends on the chosen p_0. The method applied to finding p_0 is given for each model separately in Appendix D.

The cross validation routine iterates over the regressor predictor pairs, always leaving one out and computing the fit completely analogously to the fit method. Afterwards it uses the parameter to compute the loss of the left out data point. These losses are added up and returned. The code is as follows:

```
> # p0: initial parameter
> # x: all predictors (as data.frame)
> # y: the (1d) regressor
> # l: custom loss function
> # bound: bound of the loss function
> losscv <- function(p0, fn, x, y, l = loss, bound = q) {
+   fn.min <- genmin(fn, l = l, bound = bound)
+   result <- list(value = 0, warn = c())
+   for(i in 1:length(y)) {
+     o <- optim(p0, fn.min, x = x[-i,], y = y[-i])
+     if (o$convergence != 0) {
+       result$warn <- append(result$warn, values = i)
+     }
+     result$value <- result$value + fn.min(o$par, x[i,], y[i])
+   }
+   result
+ }
```

Again we have the problem of having to choose an initial parameter p_0. Therefore we reused the initial parameters of the fit routine.

D. Model implementation and residues

In the following we show the implementation of each of the used models, the residues after the fit, the strategy for obtaining the initial parameters and the values of the estimated parameters.

D.1. Logistic interpolation with regressor: duration

This model is just a standard implementation of a logistic function interpolating between two plateaus. The parameters of the exponential function, which determine the transition region and speed are fixed and taken from the fit of the duration dependent waiting time distribution (c.f. Appendix A.3). Thus the implementation takes the form:

```
A single object matching ‘lg1.wait’ was found
It was found in the following places
  .GlobalEnv
with value

function(x,p) {
    if(is.atomic(x)) {
      d <- x
    } else {
      d <- x$d
    }
    q <- exp(l[1] + l[2]*d)
    q <- q/(1+q)
    q*p[1] + (1-q)*p[2]
  }
<environment: 0x26191a8>
```

As initial parameters for the plateaus we chose the mean of the two peaks in the waiting time distribution (again, c.f. Appendix A). The result is shown in Table 4. The residue of the fit is depicted in Figure 7 (left) on page 13. All in all there is not much structure in the residues except for a few outliers which are easily identified with the outliers in the waiting time distribution.

	Param1	Param2
Initial	55.14	81.09
Terminal	44.85	71.01

Table 4: Parameters of the logistic interpolation model

D.2. Piecewise linear model with regressor: duration

This model is very close to the official predictive model. The only difference is, that it respects the two clusters in the wating time duration plot. Thus we get two linear models, one for each cluster. Implemented this looks as follows:

```
> plm1.dur <- function(x, p) {
+   if(is.atomic(x)) {
+     d <- x
+   } else {
+     d <- x$d
+   }
+   result <- c()
+   for(i in 1:length(d)) {
+     if(x[i] < 3.1) {
+       result <- append(x = result, values = p[1] + p[2]*d[i])
+     } else {
+       result <- append(x = result, values = p[3] + p[4]*d[i])
+     }
+   }
+   result
+ }
```

As the model is adapted from the official model. We use the parameters of it as initial parameters for our fit. The result of our regression can be found in Table 5 and the residues are in Figure 7 (right). Again we see almost no structure, except for the same outliers as in the model before.

	Param1	Param2	Param3	Param4
Initial	30	10	30	10
Terminal	29.55	9	42.33	6.67

Table 5: Parameters of the piecwise linear (dur) model

D.3. Linear model with predictors: duration & waiting time

This model is a slight adaption of the official predictive model, too. Instead of taking the clustering into account we add the last waiting time as a regressor. This leads to the following implementation:

```
> lm1.durwait <- function(x,p) {
+   d <- x$d
+   w <- x$w
+   p[1] + p[2]*d + p[3]*w
+ }
```

Because of the close relation to the park model, we use its parameters as inital parameters. For the waiting time coefficient we use 0 as we suspect it will only act as a small correction. The results can be found in Table 6 and the residues are depicted in Figure 8 on page 15. As before, there is almost no structure in the residues, except for the known outliers.

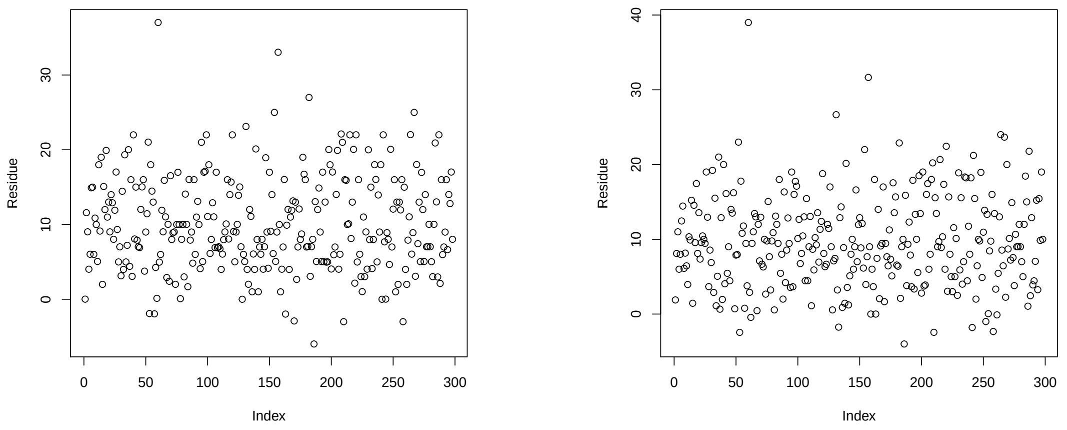

Figure 7: Residuals of Model 1 (left) and Model 2 (right)

	Param1	Param2	Param3
Initial	30	10	0
Terminal	30.49	10.02	−0.05

Table 6: Parameters of the linear (dur+wait) model

D.4. Piecewise linear model with predictors: duration & waiting time

This model is a refinement of the previous one. Again we use duration and waiting time as predictors, yet this time we apply a linear model to each of the three identified clusters separately. This increases the number of parameters to 9 and the code takes the following form:

```
> plm2.durwait <- function(x,p) {
+   d <- x$d
+   w <- x$w
+   result <- c()
+   for(i in 1:length(d)) {
+     r <- 0
+     if(d[i] <= dsep) {
+       r <- p[1] + p[2]*d + p[3]*w
+     } else {
+       if(w[i] <= wsep) {
+         r <- p[4] + p[5]*d + p[6]*w
+       } else {
+         r <- p[7] + p[8]*d + p[9]*w
+       }
+     }
+     result <- append(x = result, values = r)
+   }
+   result
+ }
```

Because of the close relation to the park model. We use its parameters as inital parameters. For the different waiting times we use 0, as we suspect only a small corrective effect. The result is summarized in Table 7 and the residues can be found in Figure 8 (right). In this case there is a little more structure than in the other cases and we have the same outliers. Additionally we see that we miss more eruptions than in the other models. Together with the high number of necessary parameters, these are indicators against the use of this model.

	Param1	Param2	Param3	Param4	Param5	Param6	Param7	Param8	Param9
Initial	30	10	0	30	10	0	30	10	0
Terminal	29.98	10.11	−0.03	32.68	10.11	−0.01	30.01	9.88	−0.04

Table 7: Parameters of the piecwise linear (dur+wait) model

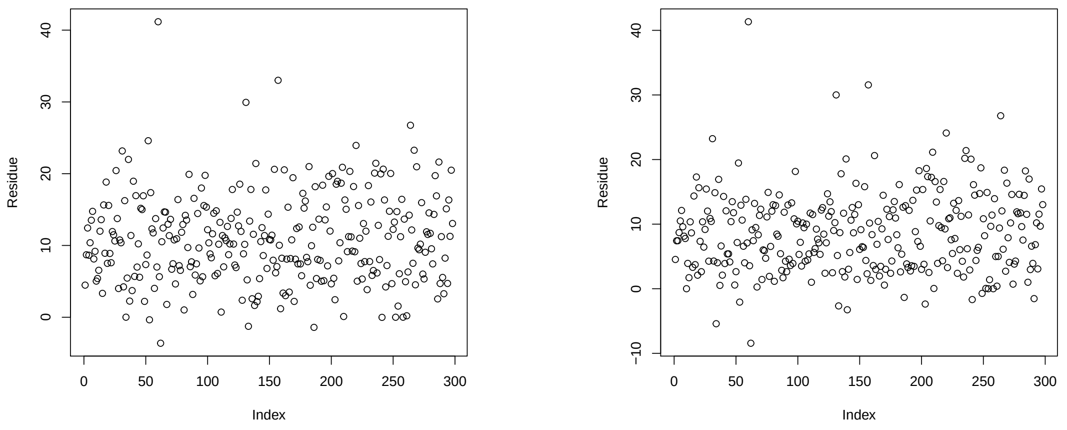

Figure 8: Residuals of Model 3 (left) and Model 4 (right)

References

[1] A. Azzalini and A. W. Bowman. ‘A Look at Some Data on the Old Faithful Geyser’. In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 39.3 (1990), pp. 357–365. ISSN: 00359254, 14679876. URL: http://www.jstor.org/stable/2347385.

[2] R. D. Cook and S. Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[3] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle. ‘Recurrence plots of dynamical systems’. In: *Europhys. Lett* 4.9 (1987), pp. 973–977.

[4] J. A. Nelder and R. Mead. ‘A simplex method for function minimization’. In: *The computer journal* 7.4 (1965), pp. 308–313.

[5] S. J. Sheather and M. C. Jones. ‘A reliable data-based bandwidth selection method for kernel density estimation’. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1991), pp. 683–690.

[6] W. N. Venables and B. D. Ripley. *Modern applied statistics with S-PLUS*. Springer Science & Business Media, 2013.