Data Science 2: Midterm Exam

March, 2018

Please include at the top of your exam whether you are a 109b/121b student, or a 209b student

This exam involves exploring a data set on red wine quality, and how quality relates to physio-chemical features of wine. A description of the study can be downloaded from the **publisher's web site.** The following questions will focus on a subset of data consisting of 1599 red wines. The data are contained in the file `winequality-red.csv`. For each bottle of wine, the following features are measured.

1. fixed acidity
2. volatile acidity
3. citric acid
4. residual sugar
5. chlorides
6. free sulfur dioxide
7. total sulfur dioxide
8. density
9. pH
10. sulphates
11. alcohol
12. quality (score between 0 and 10)

The main goal of this analysis is to predict red wine quality from the physio-chemical features.

Problem 1 [30 points]

Fit an additive model of quality on the physio-chemical variables on all 1599 wines in the data set. Use smoothing splines to fit each predictor variable. No need to explicitly perform cross-validation - please use the default smoothing selections.

(a) [5 points] Plot the smooth of each predictor variable with standard error bands. Which variables seem to have a non-linear contribution to mean quality?

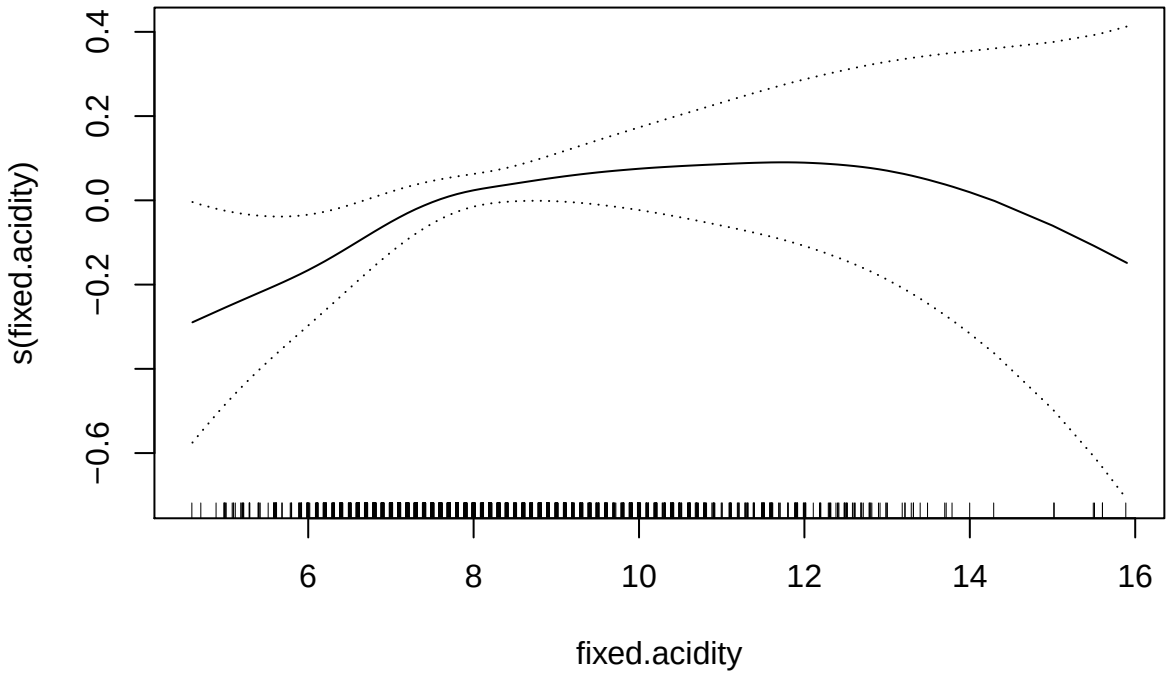
```
x = read.csv("winequality-red.csv")

library(gam)
```

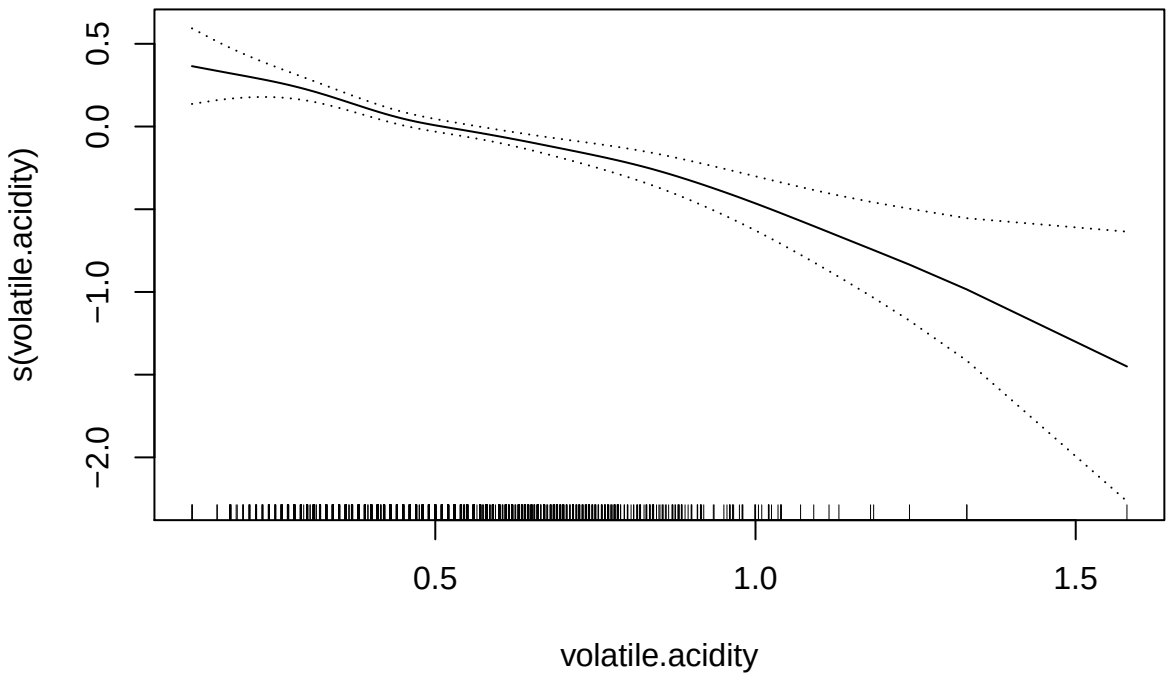
```
## Loading required package: splines
```

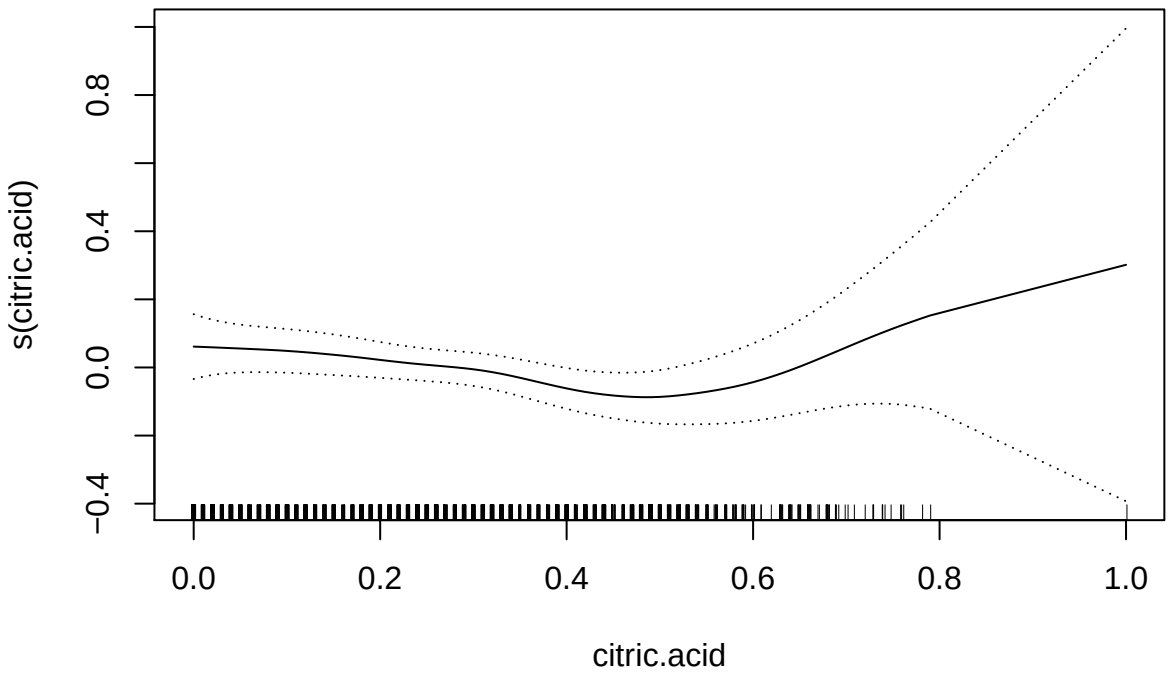
```
## Loading required package: foreach
```

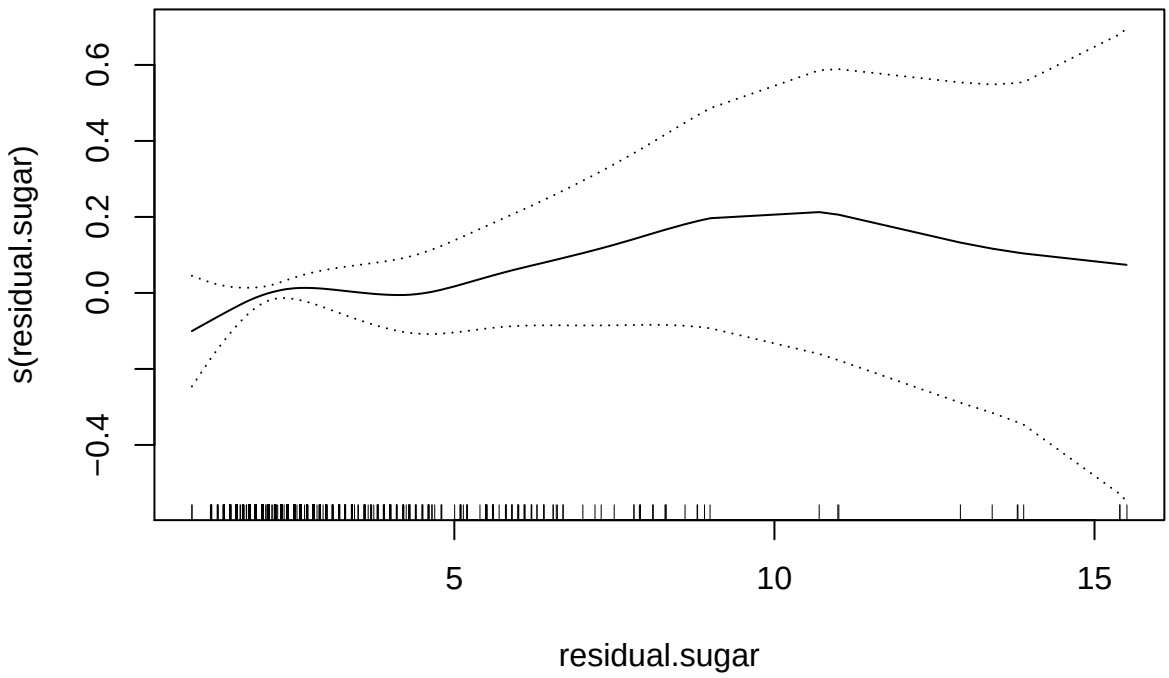
```
## Loaded gam 1.14-4
```

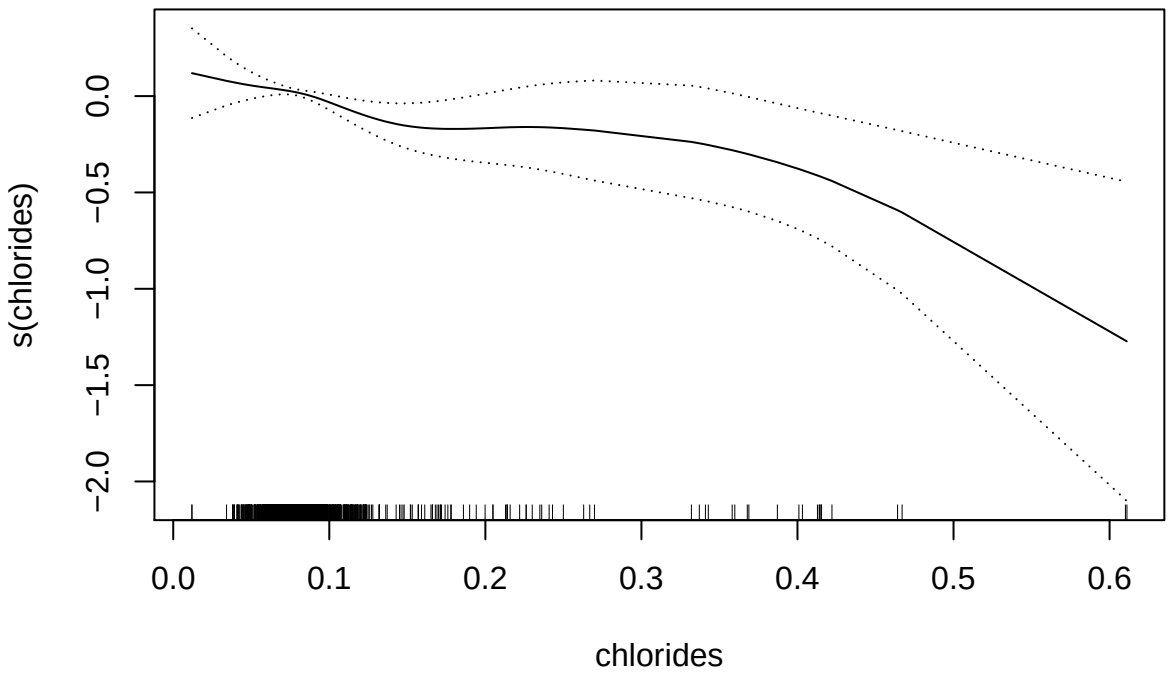
```
wine.gam = gam(quality ~
      s(fixed.acidity)+s(volatile.acidity) +
        s(citric.acid) + s(residual.sugar) + s(chlorides) +
        s(free.sulfur.dioxide) + s(total.sulfur.dioxide) +
        s(density) + s(pH) + s(sulphates) + s(alcohol),
         family=gaussian, data=x)

termnames = paste("s(",names(x)[1:11],")",sep='')
for(name in termnames){
plot(wine.gam,terms=name,rug=T,se=T,residuals=F)
}
```

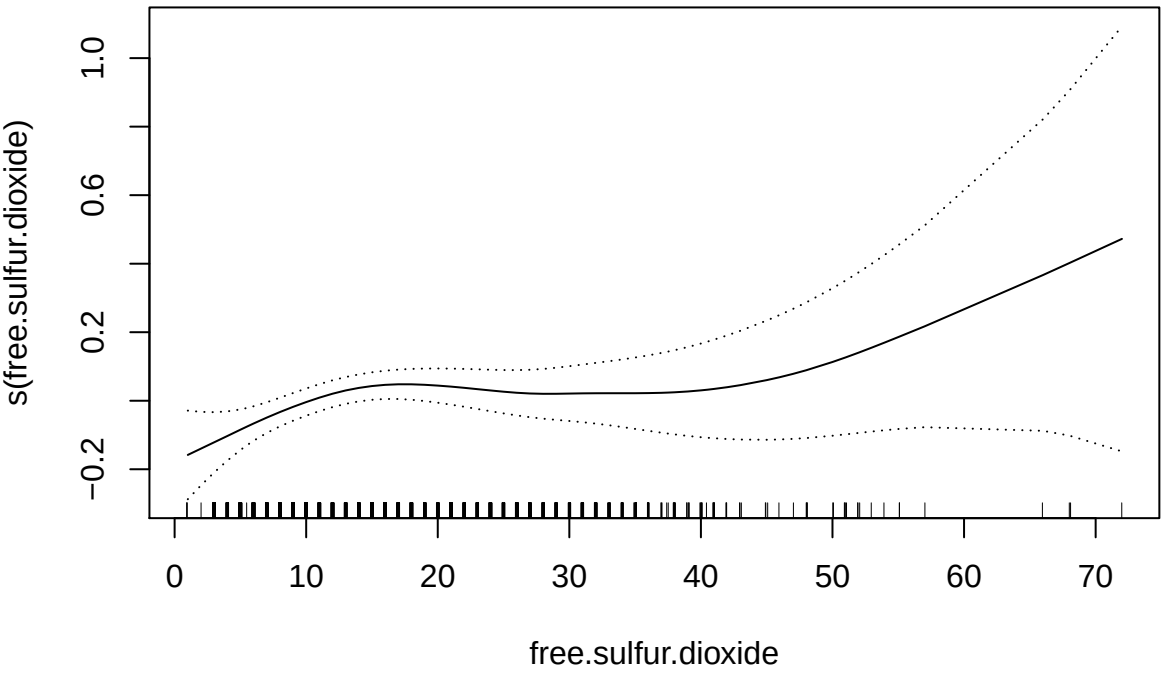


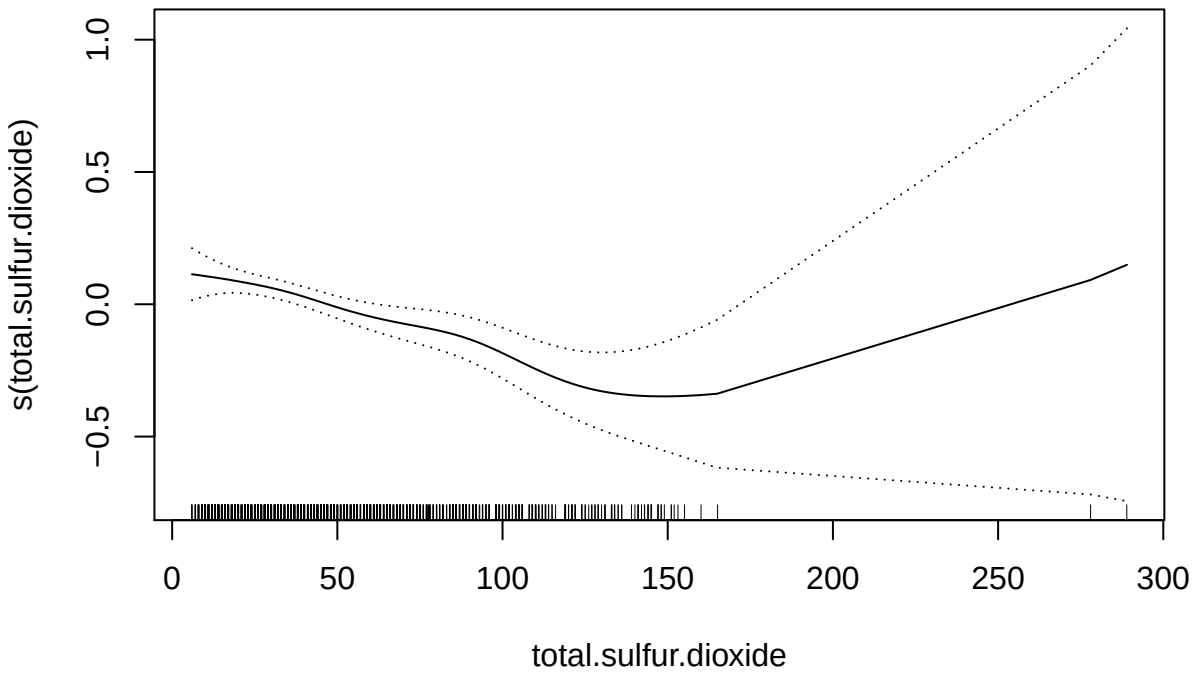

s(volatile.acidity)
0.5
0.0
−1.0
−2.0
0.5
1.0
1.5
volatile.acidity

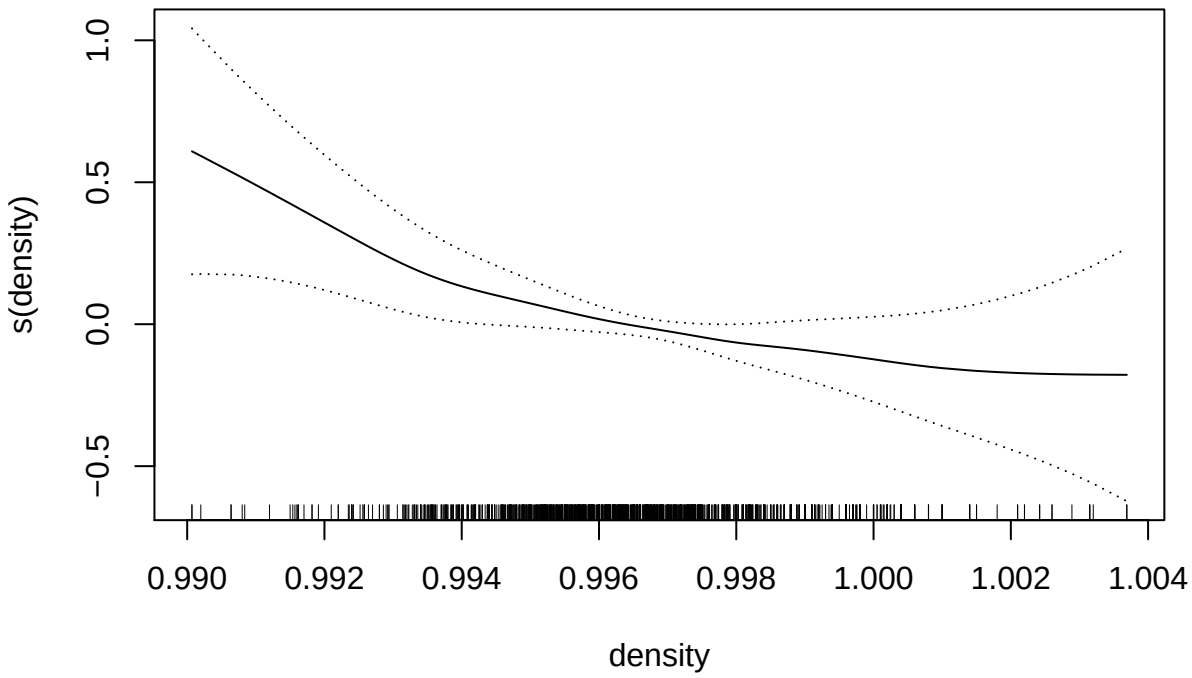

s(citric.acid)
0.8
0.4
0.0
−0.4
0.0
0.2
0.4
0.6
0.8
1.0
citric.acid

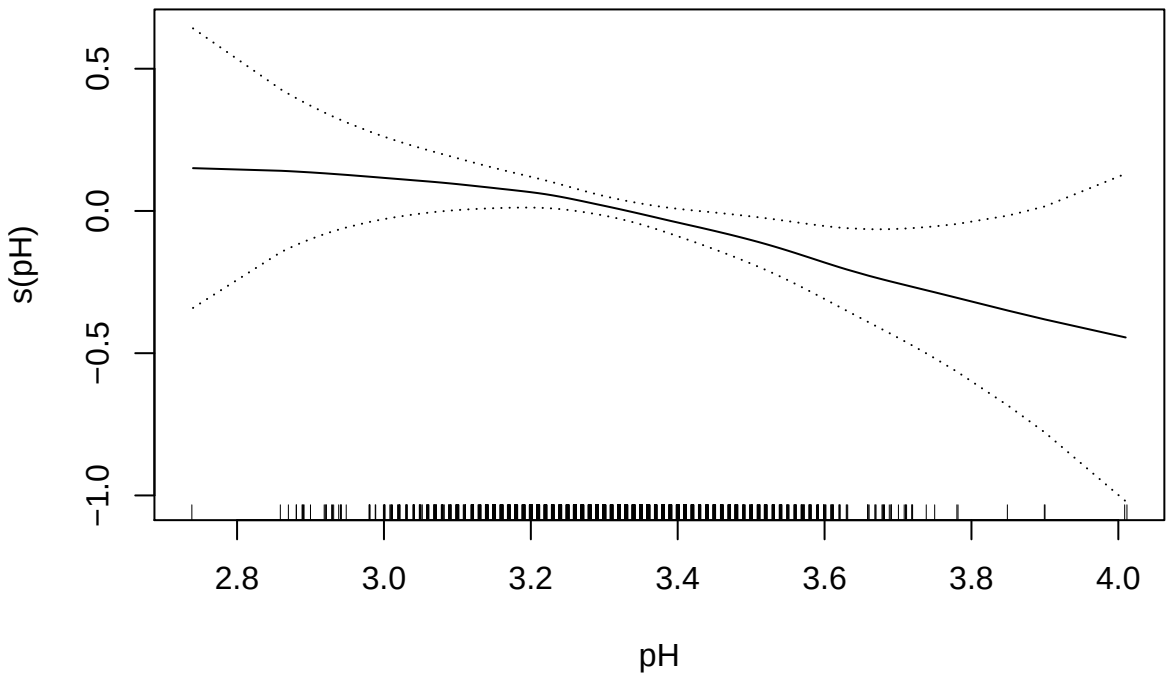

s(residual.sugar)
−0.4
0.0
0.2
0.4
0.6
5
10
15
residual.sugar

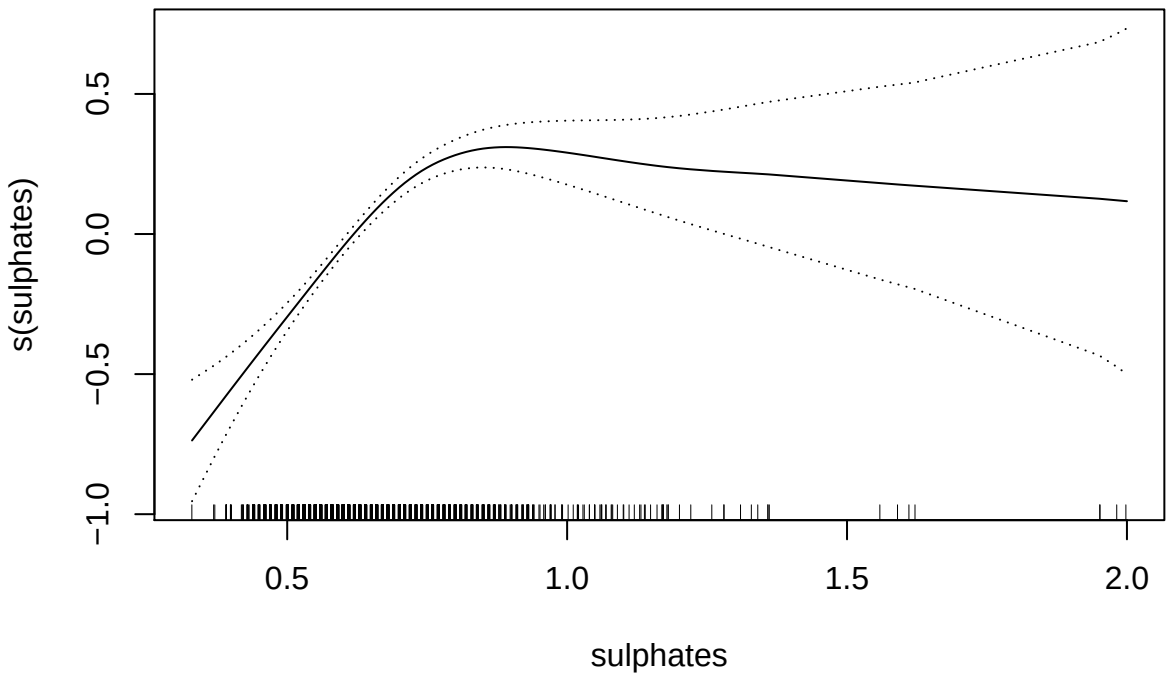

s(chlorides)
0.0
−0.5
−1.0
−1.5
−2.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
chlorides

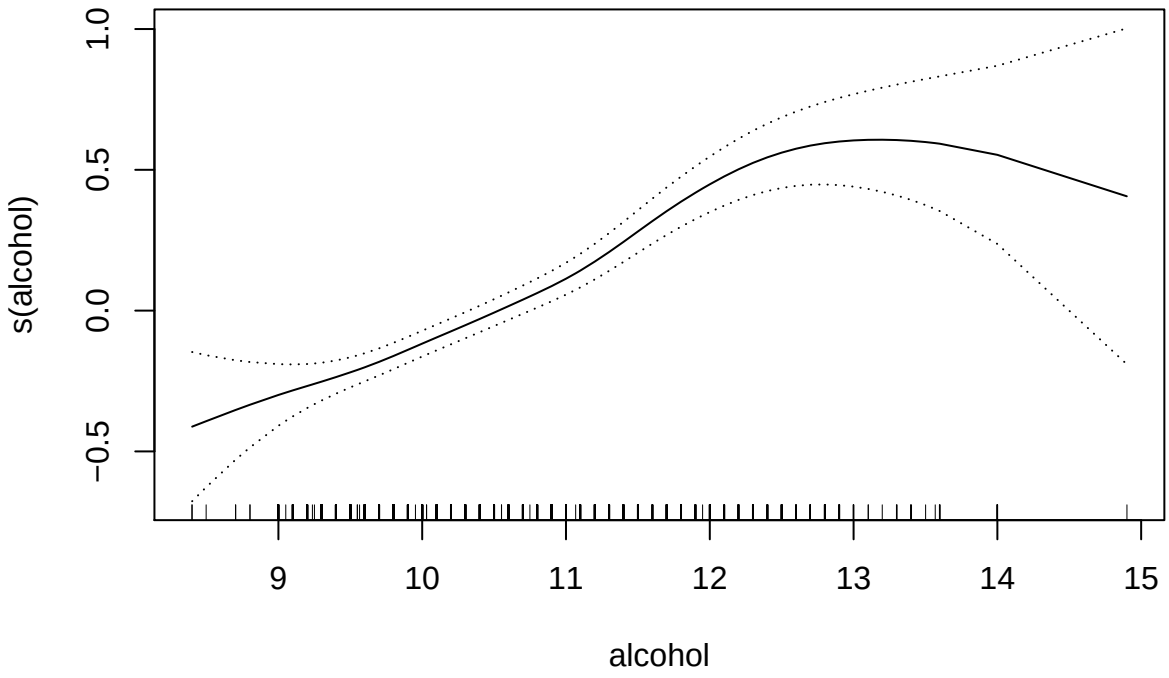

s(free.sulfur.dioxide)
1.0
0.6
0.2
−0.2
0
10
20
30
40
50
60
70
free.sulfur.dioxide


s(total.sulfur.dioxide)
1.0
0.5
0.0
−0.5
0
50
100
150
200
250
300
total.sulfur.dioxide


s(density)
1.0
0.5
0.0
−0.5
0.990
0.992
0.994
0.996
0.998
1.000
1.002
1.004
density


s(pH)
0.5
0.0
−0.5
−1.0
2.8
3.0
3.2
3.4
3.6
3.8
4.0
pH


s(sulphates)
0.5
0.0
−0.5
−1.0
0.5
1.0
1.5
2.0
sulphates



All of the predictor variables seem to evidence a clear non-linear relationship with mean quality, based on the smooths from the GAM fit, with the exceptions of `volatile acidity` (approximately linear), `chlorides` (close to linear), and `pH` (close to linear).

(b) [10 points] Is the overall non-linearity evidenced in the variable-specific smooths statistically significant? Justify your answer with a likelihood ratio test comparing the additive model to a model that includes the features linearly.

```
# Fit linear model as null model
wine.lm = gam(quality ~
      fixed.acidity + volatile.acidity + citric.acid +
        residual.sugar + chlorides + free.sulfur.dioxide +
        total.sulfur.dioxide + density + pH + sulphates + alcohol,
      family=gaussian, data=x)

# Likelihood ratio test for non-linear GAM compared to linear model
anova(wine.lm, wine.gam, test="Chi")
```

```
## Analysis of Deviance Table
##
## Model 1: quality ~ fixed.acidity + volatile.acidity + citric.acid + residual.sugar +
##     chlorides + free.sulfur.dioxide + total.sulfur.dioxide +
##     density + pH + sulphates + alcohol
```

```
## Model 2: quality ~ s(fixed.acidity) + s(volatile.acidity) + s(citric.acid) +
##     s(residual.sugar) + s(chlorides) + s(free.sulfur.dioxide) +
##     s(total.sulfur.dioxide) + s(density) + s(pH) + s(sulphates) +
##     s(alcohol)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      1587     666.41
## 2      1554     608.15 33   58.263 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value for the likelihood ratio test comparing a linear model to the GAM is roughly `2.2e-16`. This is less than any conventional significance level (e.g., 0.05), so we can reject the linear model in favor of the GAM.

(c) [10 points] We now want to investigate how to produce the best expected wine quality based on the physio-chemical content.

- [109b/121b students only] Based on the additive model fit, how might you **approximately** optimize the physio-chemical composition to produce the highest expected wine quality? Use the results from part (a) to answer this question. What is the resulting estimated wine quality? *Hint: For the latter part, use the* `predict` *function.*

From each smooth, we can find (by eye) the maximum smooth value and its corresponding feature value. The actual sets of values are listed in the answer to the 209b problem – see below.

```
# using "predict" to obtain the optimized wine quality
optimal.wine = data.frame(
  fixed.acidity=11.7,
  volatile.acidity=0.12,
  citric.acid=1.0,
  residual.sugar=10.7,
  chlorides=0.012,
  free.sulfur.dioxide=72.0,
  total.sulfur.dioxide=289,
  density=0.99007,
  pH=2.74,
  sulphates=0.89,
  alcohol=13.2)
predict(wine.gam,newdata=optimal.wine)
```

```
##        1
## 9.022752
```

The resulting maximum quality comes out to `9.023`, so anything close to this based on an analysis by eye is fine.

- [209b students only] Based on the GAM fit, determine **numerically** the optimal

combination of physio-chemical features to produce the highest expected wine quality. What are the values of the physio-chemical features corresponding to the optimal wine, and what is the quality rating for the optimized wine? *Hint: The* `preplot` *function applied to a GAM fit produces a list containing as many components as there are smoothed predictors, and each component is a list itself containing the* `x` *and* `y` *values that produce the variable-specific smooths.* It may help that the intercept is the sample mean of the quality scores. Or you can use the values as part of the `predict` function to obtain the estimated value of the optimized wine.

```
z = preplot(wine.gam)
wine.intercept = mean(x$quality)
winemax.mat = NULL
for(i in 1:length(z)){
  tmp.lst = z[[i]]
  y.max = max(tmp.lst$y)
  x.max = (tmp.lst$x[tmp.lst$y==y.max])[1]
  winemax.mat = rbind(winemax.mat, c(x.max,y.max))
}
dimnames(winemax.mat) = list(names(z), c("Xvalue","Yvalue"))
winemax.mat
```

```
##                              Xvalue     Yvalue
## s(fixed.acidity)          11.70000 0.09021988
## s(volatile.acidity)        0.12000 0.36504070
## s(citric.acid)             1.00000 0.30147330
## s(residual.sugar)         10.70000 0.21267067
## s(chlorides)               0.01200 0.11899364
## s(free.sulfur.dioxide)    72.00000 0.47246106
## s(total.sulfur.dioxide) 289.00000 0.14920649
## s(density)                 0.99007 0.60894282
## s(pH)                      2.74000 0.15050825
## s(sulphates)               0.89000 0.31023475
## s(alcohol)                13.20000 0.60686812
```

```
# optimal expected wine quality:
wine.intercept + sum(winemax.mat[,2])
```

```
## [1] 9.022642
```

The wine feature values and their corresponding maximum contribution to the GAM is listed above. The overall expected quality comes out to `9.023` through this optimization procedure.

(d) [5 points] What might be a concern or limitation with optimizing the physio-chemical composition of wine based on the additive model fit? Your answer should be connected to the assumptions underlying additive models.

The assumptions underlying additive models is that each features has a contribution to the mean response that does not interact with the other features. Thus optimizing for each feature

separately does not acknowledge that pairs of features may vary together. For example, it may be that `pH` is optimized at `2.74`, and `alcohol` is optimized at `13.2`, but it may be that it is impossible for wine to have both of these values simultaneous for `pH` and `alcohol`.

Problem 2 [40 points]

Rather than fit a single model to all of the wines, we will fit different models to different subsets of the data (in Problem 3). In preparation, this problem will involve partitioning the data into different clusters/subsets.

(a) [5 points] Explain a reason we might expect different relationships between quality and physio-chemical wine composition by different subsets of the data identified in Problem 1.

It is possible that the different groupings of wine from Problem 1 are sufficiently distinct that evaluations of overall quality may have different relationships with the physio-chemical composition. For example, if one cluster contains wines with low `alcohol` and low `pH`, and another cluster contains wines with high `alcohol` and high `pH`, then it is possible that high-quality wines in the first group would have `alcohol` content as the more relevant feature and that high-quality wines in the second group would have `pH` as the more relevant feature.

(b) [5 points] Prior to performing clustering, you will center each column, and also scale each column so that each transformed feature has a standard deviation of 1.0. Briefly justify the decision to scale the data in this manner. Be specific to the context of this problem.

```
# summary of wines
summary(x[,1:11])
```

```
##  fixed.acidity   volatile.acidity  citric.acid    residual.sugar
##  Min.   : 4.60   Min.   :0.1200   Min.   :0.000   Min.   : 0.900
##  1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900
##  Median : 7.90   Median :0.5200   Median :0.260   Median : 2.200
##  Mean   : 8.32   Mean   :0.5278   Mean   :0.271   Mean   : 2.539
##  3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600
##  Max.   :15.90   Max.   :1.5800   Max.   :1.000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 22.00
##  Median :0.07900   Median :14.00       Median : 38.00
##  Mean   :0.08747   Mean   :15.87       Mean   : 46.47
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 62.00
##  Max.   :0.61100   Max.   :72.00       Max.   :289.00
##     density             pH          sulphates         alcohol
##  Min.   :0.9901   Min.   :2.740   Min.   :0.3300   Min.   : 8.40
##  1st Qu.:0.9956   1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50
```

```
##  Median :0.9968   Median :3.310   Median :0.6200   Median :10.20
##  Mean   :0.9967   Mean   :3.311   Mean   :0.6581   Mean   :10.42
##  3rd Qu.:0.9978   3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10
##  Max.   :1.0037   Max.   :4.010   Max.   :2.0000   Max.   :14.90
```

The scales of the features vary considerably, with `density` ranging from `0.9901` to `1.0037`, and `total.sulfur.dioxide` ranging from `6.0` to `289.0`. Because clustering methods for these data rely on pairwise dissimilarities among the individual wines, then without scaling the data the distances would be mostly determined by the small number of features that have scales of measurement that produce large values. By scaling the data so that each feature has equal standard deviation, all variables will contribute equally to the dissimilarity measure.

(c) [10 points] Suppose we decide to perform partitioning-around-medoids clustering of the observations based only on the physio-chemical features but not using quality. To determine the best number of clusters, optimize based on the gap statistic in the following manner:

1. Set the random number seed to 123 (`set.seed(123)`). Now select a random sample of 200 wines (*hint: use the* `sample` *function*).
2. Set the random number seed to 321 (`set.seed(321)`). Optimize the gap statistic using the method described by Tibshirani (2001) based on the standard error rule, using `d.power=2`.

```
# load libraries - don't need all of them, but cannot hurt to load all
library(ggplot2)
library(factoextra)
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/
```

```
library(cluster)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

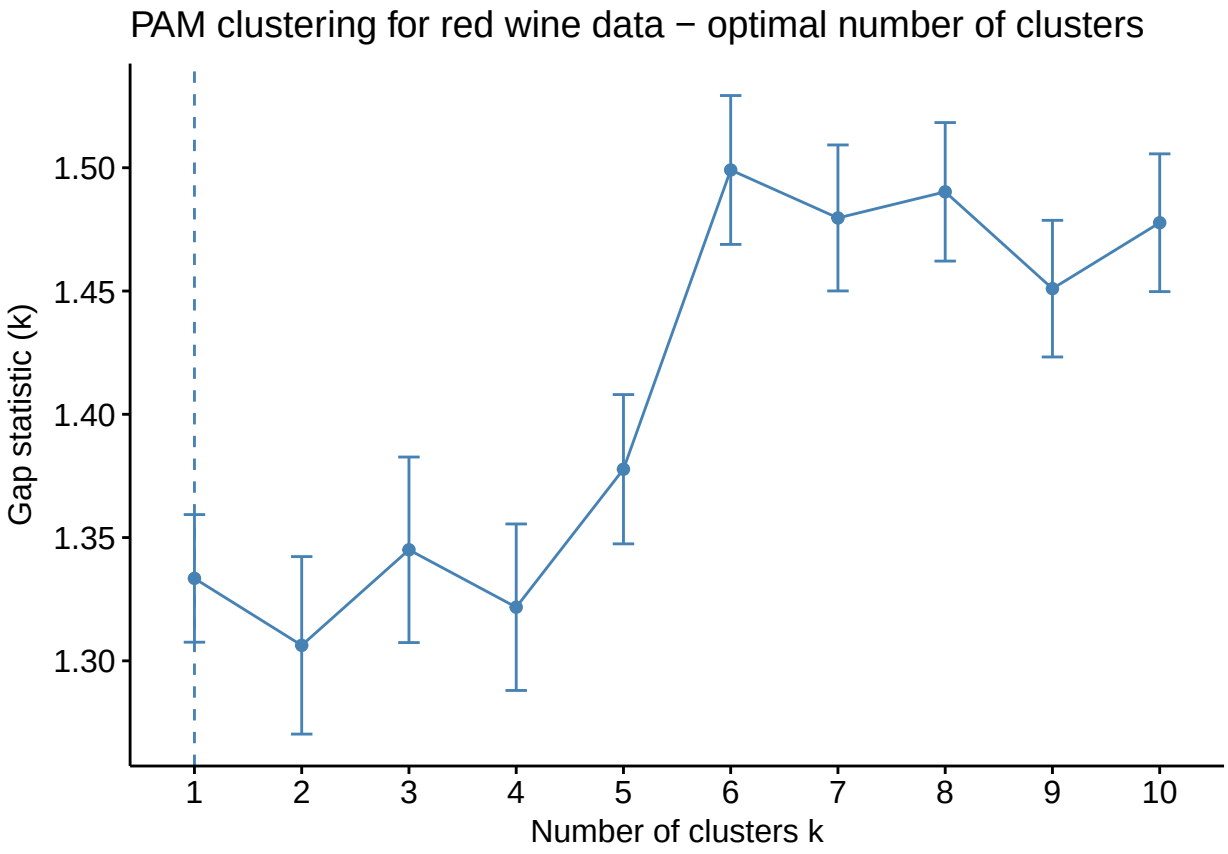
```
library(NbClust)
library(mclust)
```

```
## Package 'mclust' version 5.4
## Type 'citation("mclust")' for citing this R package in publications.
```

```
library(dbscan)
library(MASS)

x.sm = x[,1:11] # no quality variable
set.seed(123)
x.sm.sub = scale(x.sm)[sample(nrow(x.sm),200),]
set.seed(321)
gapstat = clusGap(x.sm.sub,FUN=pam,d.power=2,K.max=10,B=500)
print(gapstat, method="Tibs2001SEmax")
```

```
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = x.sm.sub, FUNcluster = pam, K.max = 10, B = 500,     d.power = 2)
## B=500 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
##  --> Number of clusters (method 'Tibs2001SEmax', SE.factor=1): 1
##          logW   E.logW      gap     SE.sim
##  [1,] 6.972844 8.306292 1.333448 0.02588282
##  [2,] 6.813394 8.119683 1.306289 0.03599735
##  [3,] 6.626267 7.971310 1.345042 0.03763001
##  [4,] 6.551881 7.873632 1.321751 0.03376585
##  [5,] 6.423769 7.801493 1.377724 0.03027421
##  [6,] 6.244083 7.743181 1.499099 0.03019518
##  [7,] 6.213207 7.692842 1.479635 0.02960796
##  [8,] 6.159896 7.650126 1.490230 0.02808761
##  [9,] 6.159868 7.610831 1.450962 0.02770733
## [10,] 6.099090 7.576785 1.477695 0.02793793
```

```
fviz_gap_stat(gapstat,
              maxSE=list(method="Tibs2001SEmax",SE.factor=1)) +
  ggtitle("PAM clustering for red wine data - optimal number of clusters")
```



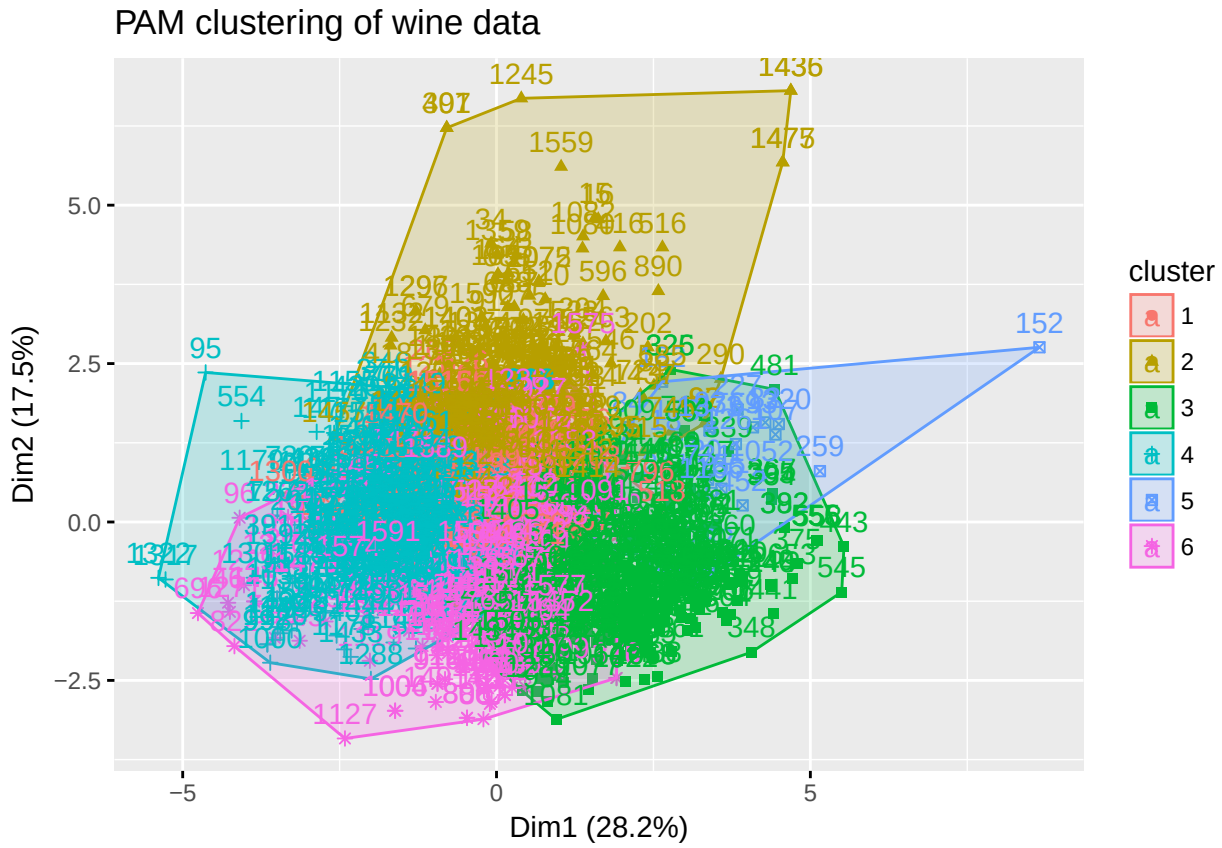
Explain how 1 cluster is the optimal number of clusters according to Tibshirani's rule, even though 6 clusters would be chosen if one were to use the maximum

gap statistic.

Tibshirani's rule is to recursively consider a larger number of clusters only if the gap statistic for the current number of clusters is less than the gap statistic for the next larger number of clusters less its standard deviation. In this case, because the gap statistic for one cluster is already bigger than the gap statistic for two clusters (not even considering the standard deviation), Tibshirani's rule suggests that we stop at one cluster.

(d) [10 points] Partition the full data into six clusters via partitioning-around-medoids on the scaled version of the data. Save the cluster identifiers as a new column in the original data frame (*Hint: the* `clustering` *component of the resulting cluster object contains the IDs*). Plot the first two principal components of the scaled data and visually show the cluster memberships. Show that the proportion of variance in the original data represented by the principal component plot is 45.7%. Use the output of `prcomp` to demonstrate this.

```
wine.pam = pam(scale(x.sm), 6)
fviz_cluster(wine.pam,
             main="PAM clustering of wine data")
```




```
wine.prcomp = prcomp(scale(x.sm))
wine.prcomp.var = wine.prcomp$sdev^2
cumsum(wine.prcomp.var)/sum(wine.prcomp.var) # second is answer:  45.7%
```

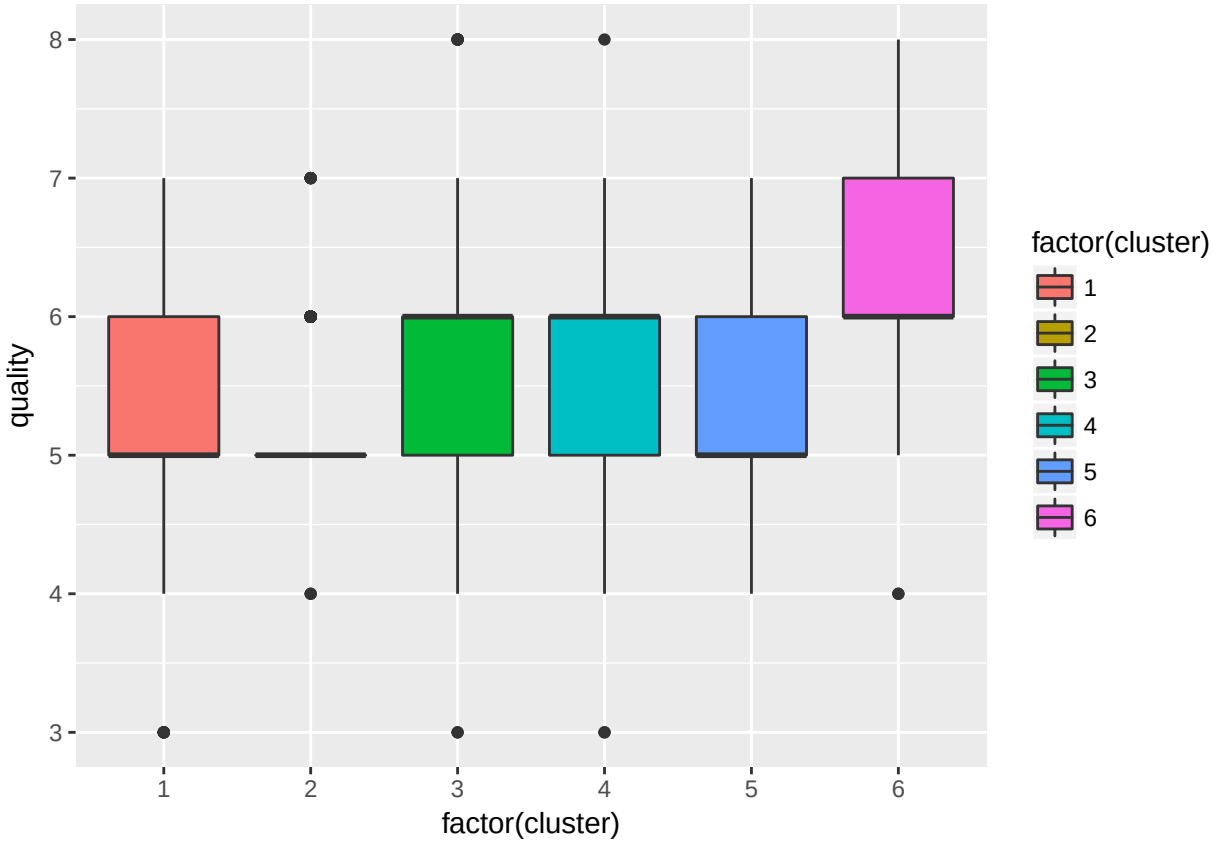
```
##  [1] 0.2817393 0.4568220 0.5977805 0.7080744 0.7952827 0.8552471 0.9083191
##  [8] 0.9467697 0.9781008 0.9945856 1.0000000
```

```
x$cluster = wine.pam$clustering
```

The proportion of variance in the first two principal components is therefore 45.7%.

(e) [10 points] Create a side-by-side boxplot of quality scores by cluster (*Hint: If using* `ggplot` *you should use the* `geom_boxplot` *function – do not forget to make the cluster ID variable a factor in R*). Does the distribution of quality scores differ visually by the clusters you determined? Would you have expected the distribution of quality scores to differ?

```
ggplot(x, aes(x=factor(cluster), y=quality, fill=factor(cluster))) +
  geom_boxplot()
```



The distribution of wine quality scores generally seem comparable, with no clear differences in the distributions (except perhaps in cluster 6 that tend to have higher quality scores, though with a median of 6.0 like clusters 3 and 4).

It is not terribly surprising that the distributions of quality scores do not differ much. The clustering procedure identified clusters of wines according to their physio-chemical content, and not quality. Only if the quality scores varied by physio-chemical features in exactly the way the clustering procedure partitioned the data would we expect the boxplot of quality

scores to show differing distributions. But we are not so lucky.

Problem 3 [30 points]

We will now fit a normal hierarchical linear model for quality scores against the physio-chemical predictors nested in the formed clusters from the previous problem.

(a) [10 points] Implement a normal hierarchical linear model in Stan (called from R) to fit the model. Make sure you let all the linear model coefficients vary by cluster. *Hint: You may find the Stan code supplied with the lecture notes helpful.* You may assume that the intercepts across the six clusters have a normal prior distribution with a mean which is the average of the quality scores across the whole data set, and with an unknown standard deviation. The 11 physio-chemical coefficients across the six clusters can be assumed to be normally distributed centered at 0 with different standard deviations. Finally, you may assume that all the standard deviation parameters have a prior uniform distribution with a minimum of 0 and maximum of 100 (which is sufficiently large).

(b) [10 points] Briefly report on the details of your model implementation (number of iterations of burn-in and the number of iterations of saved parameters, number of parallel samplers, and any assurances that the sampler converged). (*Hint: If you saved the Stan fit of your model in the R object* `wine.fit`, *you can access the matrix of model summaries from* `summary(wine.fit)$summary`.) Do not be concerned about warnings of divergent transitions after warm-up if you have evidence that the sampler converged for the feature coefficients.

(c) [10 points] Create a visualization that demonstrates the variation of coefficients across clusters. One natural way would be to display side-by-side boxplots of the posterior simulated draws for the relevant coefficients. (*Hint: Use the* `extract` *function applied to the fitted Stan model to obtain simulated coefficient values.*) Based on these results, do you think that the hierarchical model by formed clusters was helpful in explaining the variation in quality scores? Briefly justify.

```
# stan analyses
library("rstan")
```

```
## Loading required package: StanHeaders
```

```
## rstan (Version 2.16.2, packaged: 2017-07-03 09:24:58 UTC, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## rstan_options(auto_write = TRUE)
## options(mc.cores = parallel::detectCores())
```

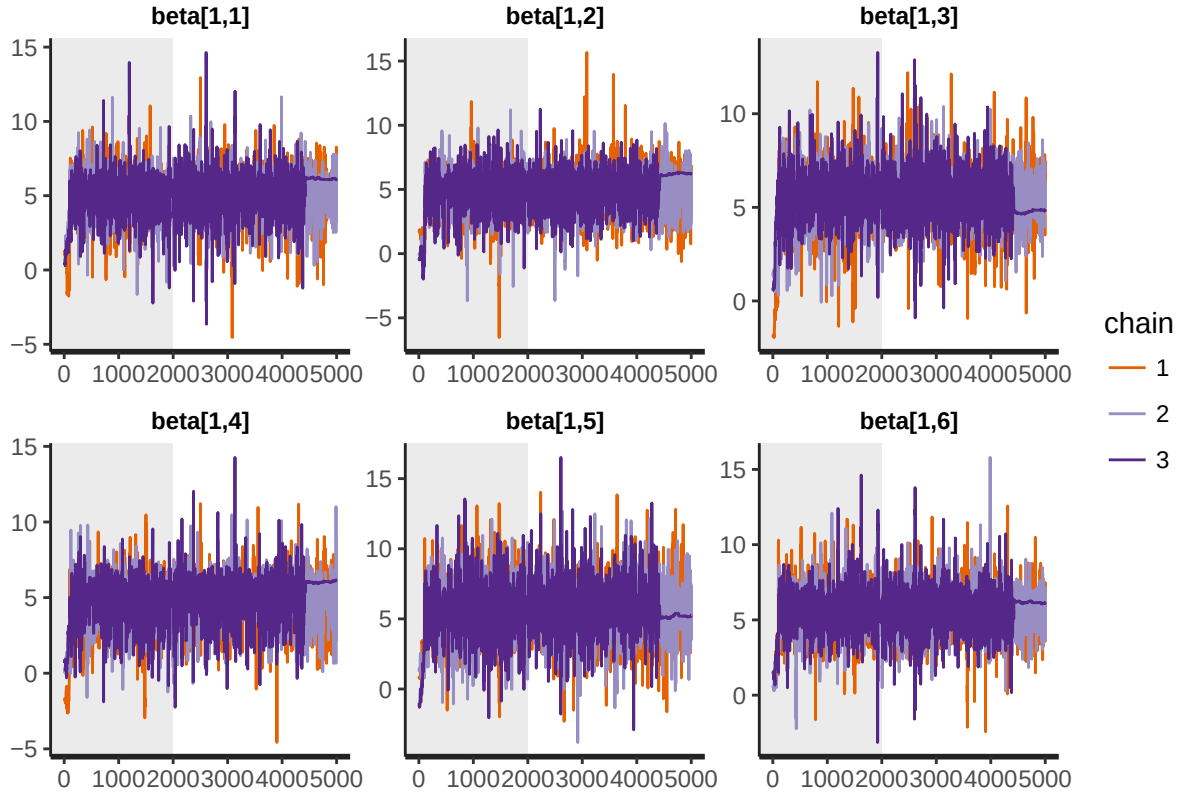
```
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

##
```

```
## Contents of wine.stan:
##
## data {
##    int<lower=0> N;  // number of obs
##    int<lower=0> N_ID; // number of clusters
##    int<lower=0> N_vars; // number of predictors (incl intercept)
##    real y[N];  // quality scores
##    matrix[N,N_vars] X;  // predictor matrix
##    real y_avg;  // average quality score (used for intercept prior mean)
##    int cluster[N];  // cluster IDs
## }
##
## parameters {
##    matrix[N_vars,N_ID] beta;
##    real<lower=0,upper=100> sig;
##    real<lower=0,upper=100> tau_beta[N_vars];
## }
##
## transformed parameters {
##    matrix[N,N_ID] linpred_mat;
##    vector[N] linpred;
##    linpred_mat = multiply(X, beta);  // N x N_ID
##    for (i in 1:N){
##      linpred[i] = linpred_mat[i,cluster[i]];
##    }
## }
##
## model {
##    sig ~ uniform(0,100);
##    for(k in 1:N_ID){
##      tau_beta[k] ~ uniform(0,100);
##      }
##    for (k in 1:N_ID){  // prior on beta[j,k]
##      beta[1,k] ~ normal(y_avg, tau_beta[1]); // intercepts
##      for(j in 2:N_vars){
##       beta[j,k] ~ normal(0, tau_beta[j]);  // other coefs
##      }
##    }
##    for (i in 1:N){
##      y[i] ~ normal(linpred[i], sig);
##      }
## }

wine.dat = list(
```

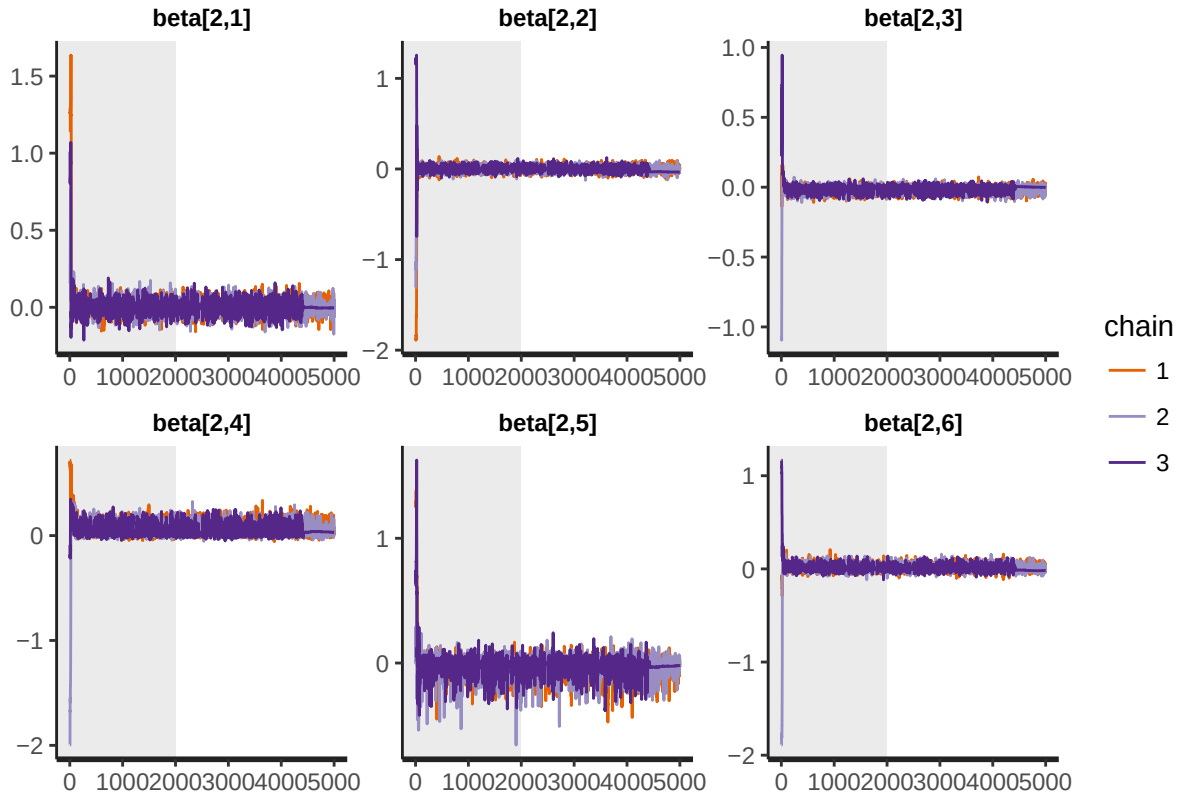
```
  X = cbind(1,x[,1:11]),
  y = x$quality,
  cluster = x$cluster,
  N = nrow(x),
  N_ID = max(x$cluster),
  N_vars = 12, # number of predictors, including intercept
  y_avg = mean(x$quality)
)

wine.fit = stan(file="wine.stan", data=wine.dat,
    iter = 5000, warmup=2000, chains = 3)
```

```
## Warning: There were 676 divergent transitions after warmup. Increasing adapt_delta ab
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

## Warning: There were 28 transitions after warmup that exceeded the maximum treedepth.
## http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded

## Warning: Examine the pairs() plot to diagnose sampling problems
```

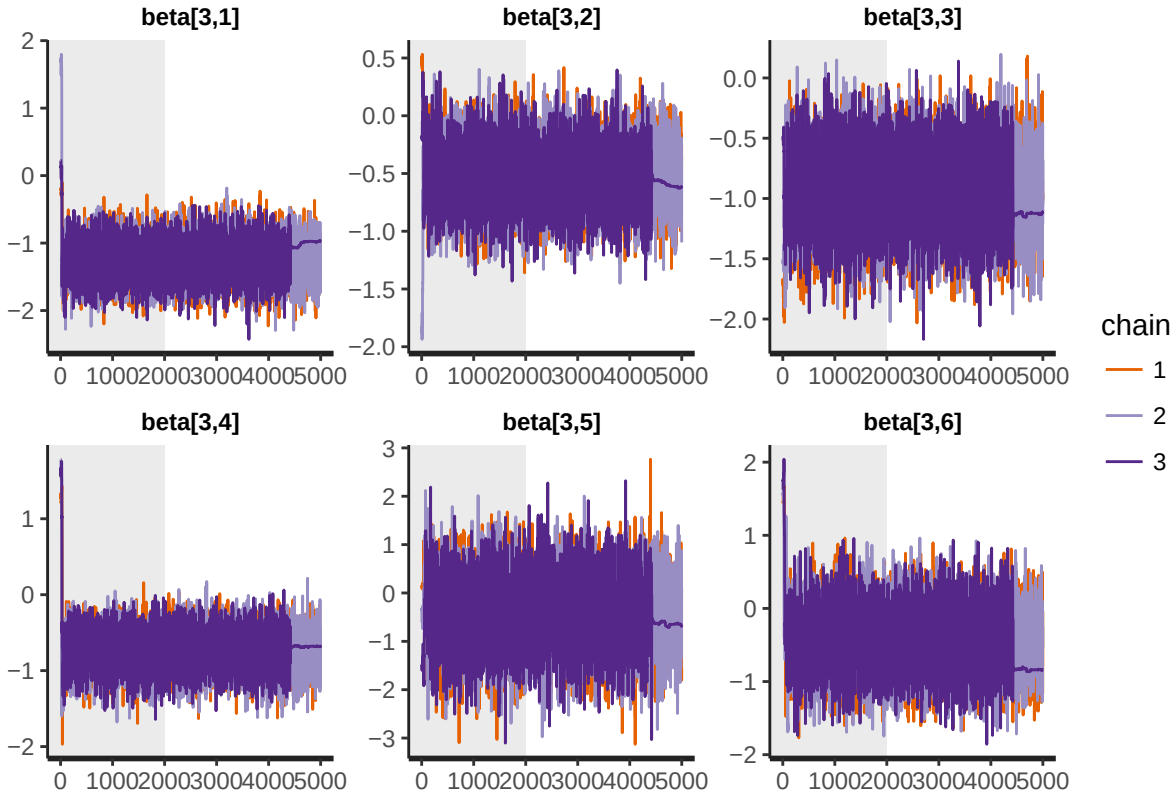
```
# create traceplots

traceplot(wine.fit,pars=c("beta[1,1]","beta[1,2]","beta[1,3]","beta[1,4]","beta[1,5]","b
```



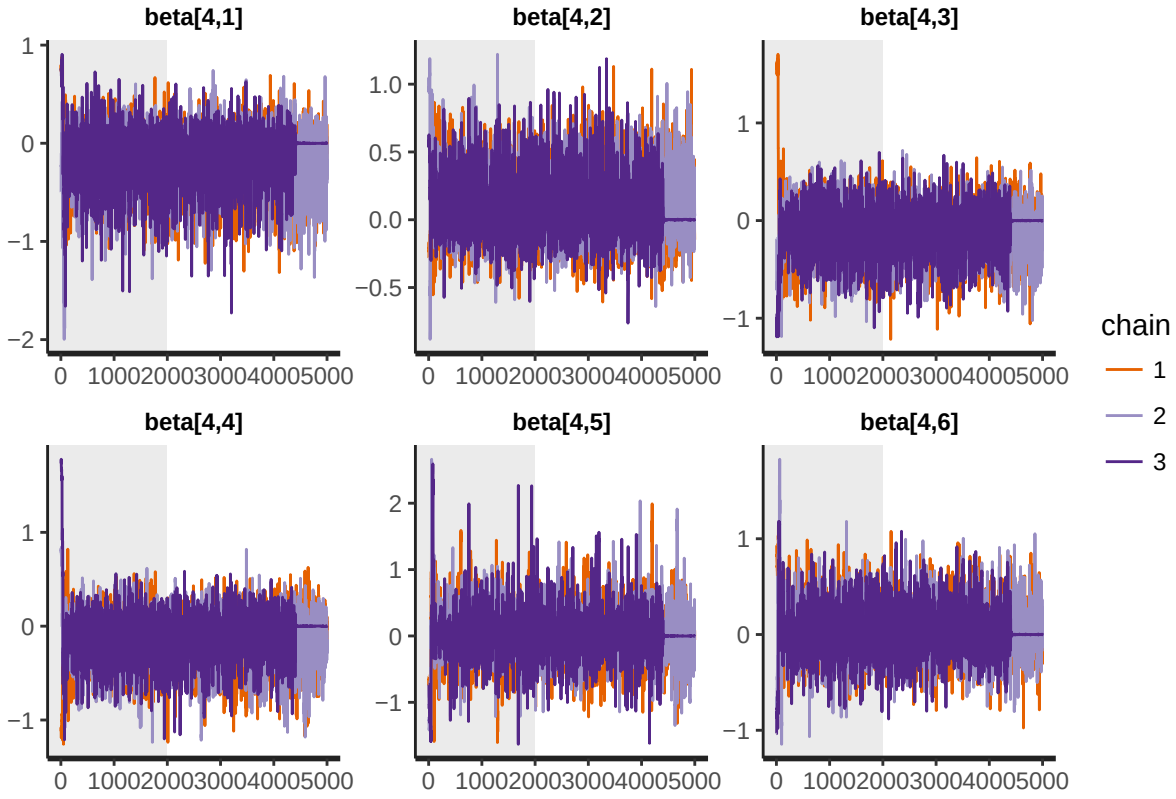
```
traceplot(wine.fit,pars=c("beta[2,1]","beta[2,2]","beta[2,3]","beta[2,4]","beta[2,5]","b
```



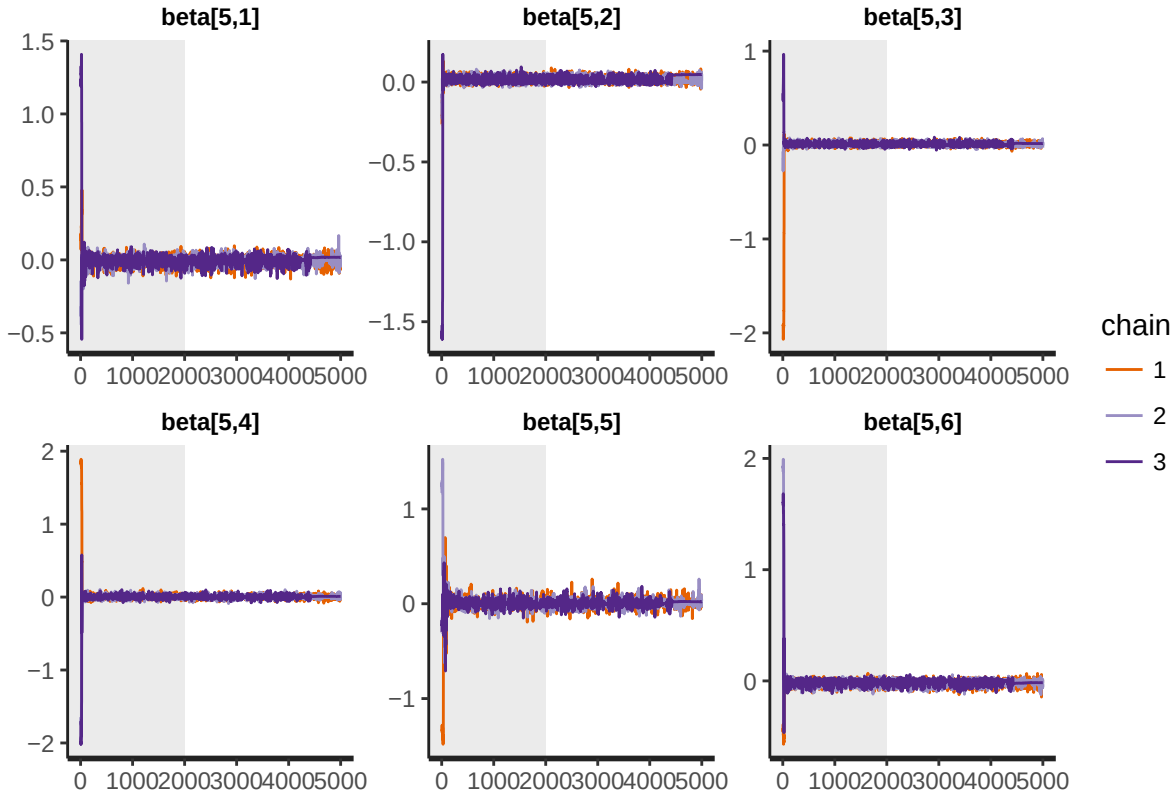

```
traceplot(wine.fit,pars=c("beta[3,1]","beta[3,2]","beta[3,3]","beta[3,4]","beta[3,5]","b
```



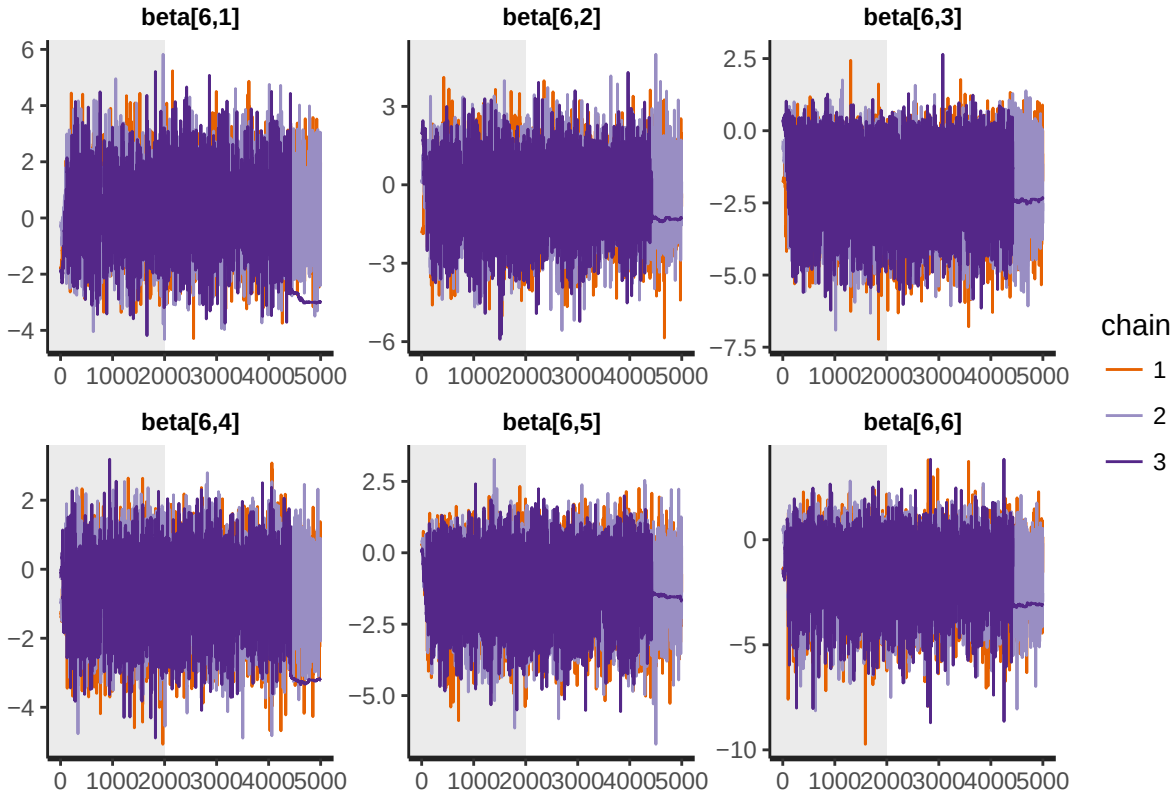

```
traceplot(wine.fit,pars=c("beta[4,1]","beta[4,2]","beta[4,3]","beta[4,4]","beta[4,5]","b
```



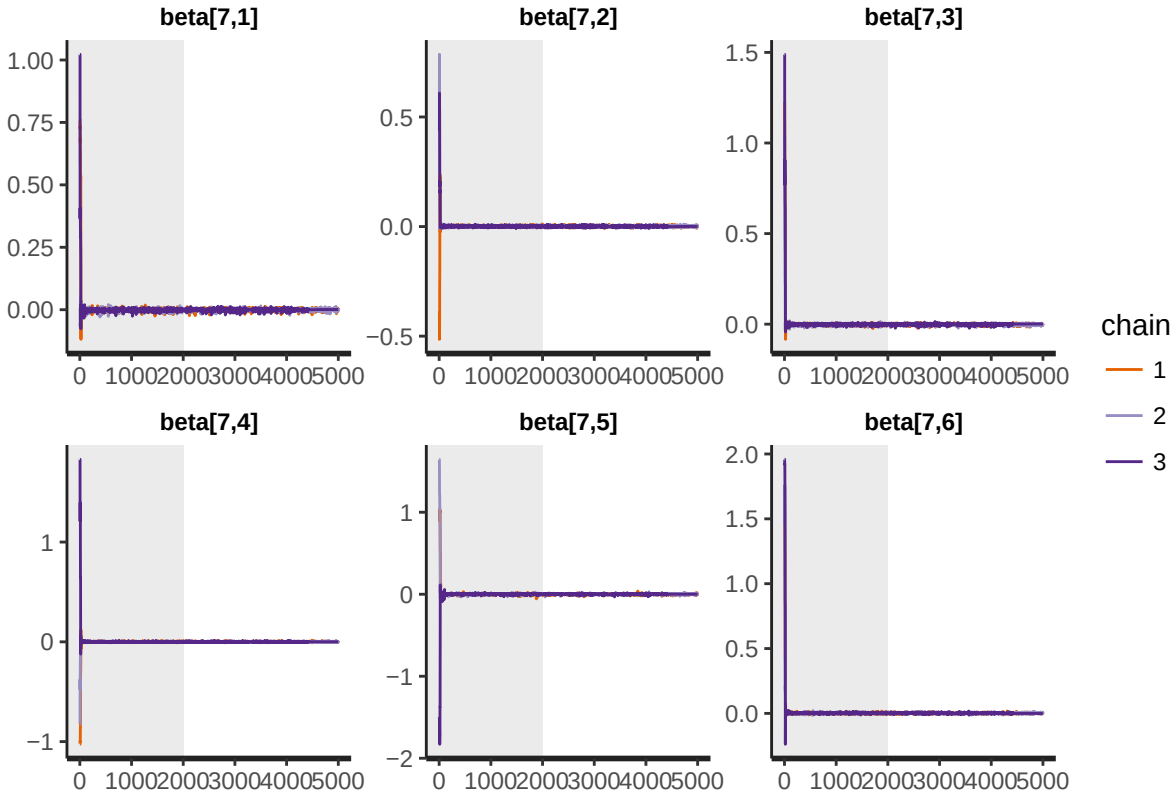

```
traceplot(wine.fit,pars=c("beta[5,1]","beta[5,2]","beta[5,3]","beta[5,4]","beta[5,5]","b
```



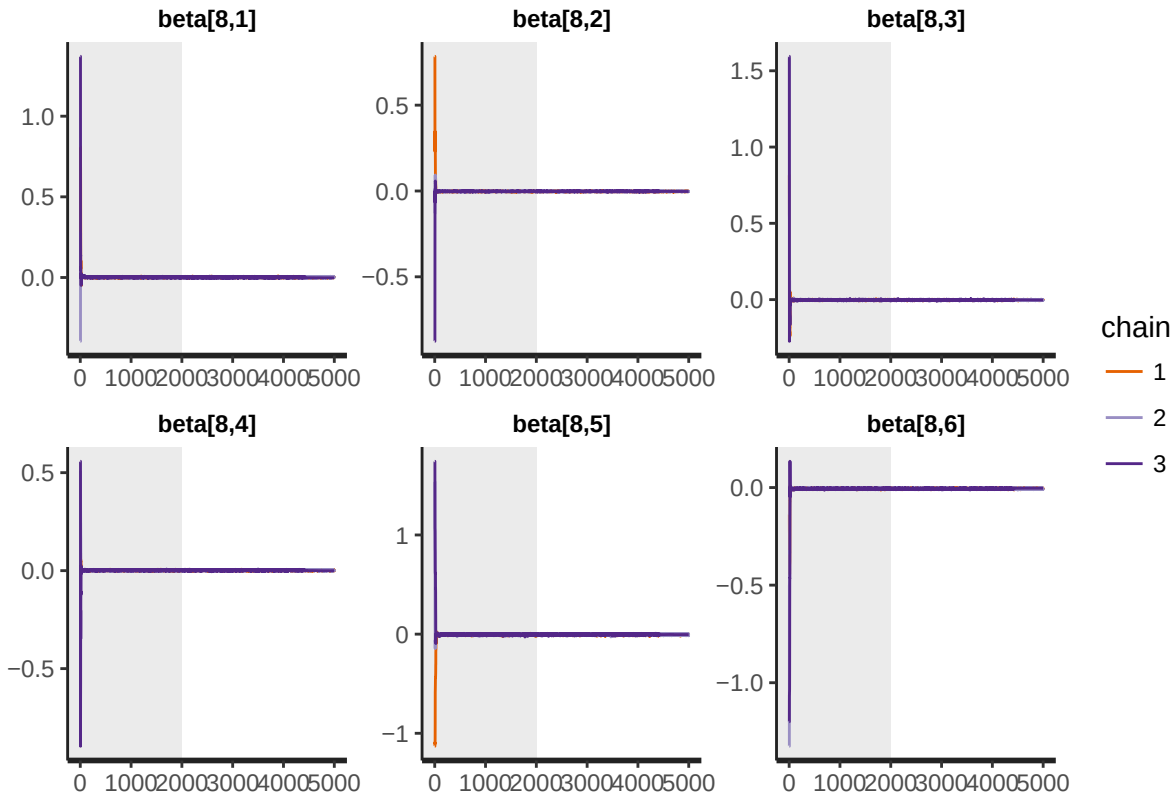

```
traceplot(wine.fit,pars=c("beta[6,1]","beta[6,2]","beta[6,3]","beta[6,4]","beta[6,5]","b
```



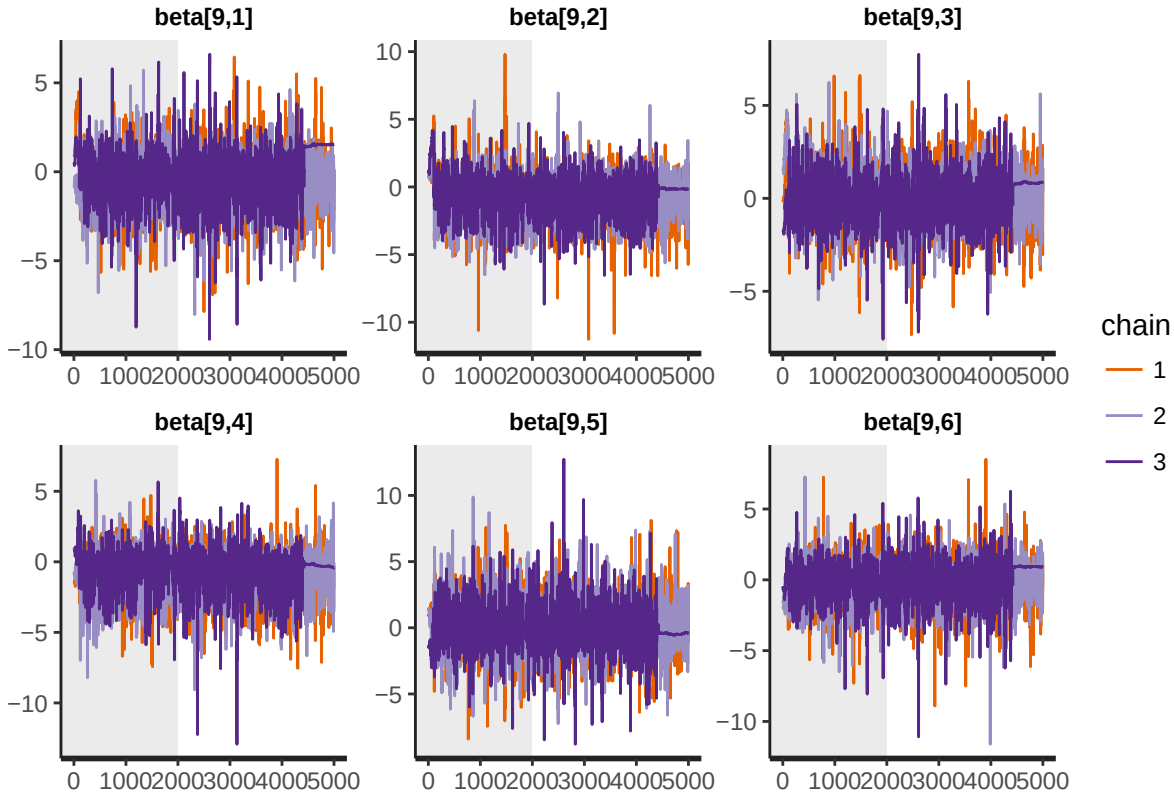

```
traceplot(wine.fit,pars=c("beta[7,1]","beta[7,2]","beta[7,3]","beta[7,4]","beta[7,5]","b
```



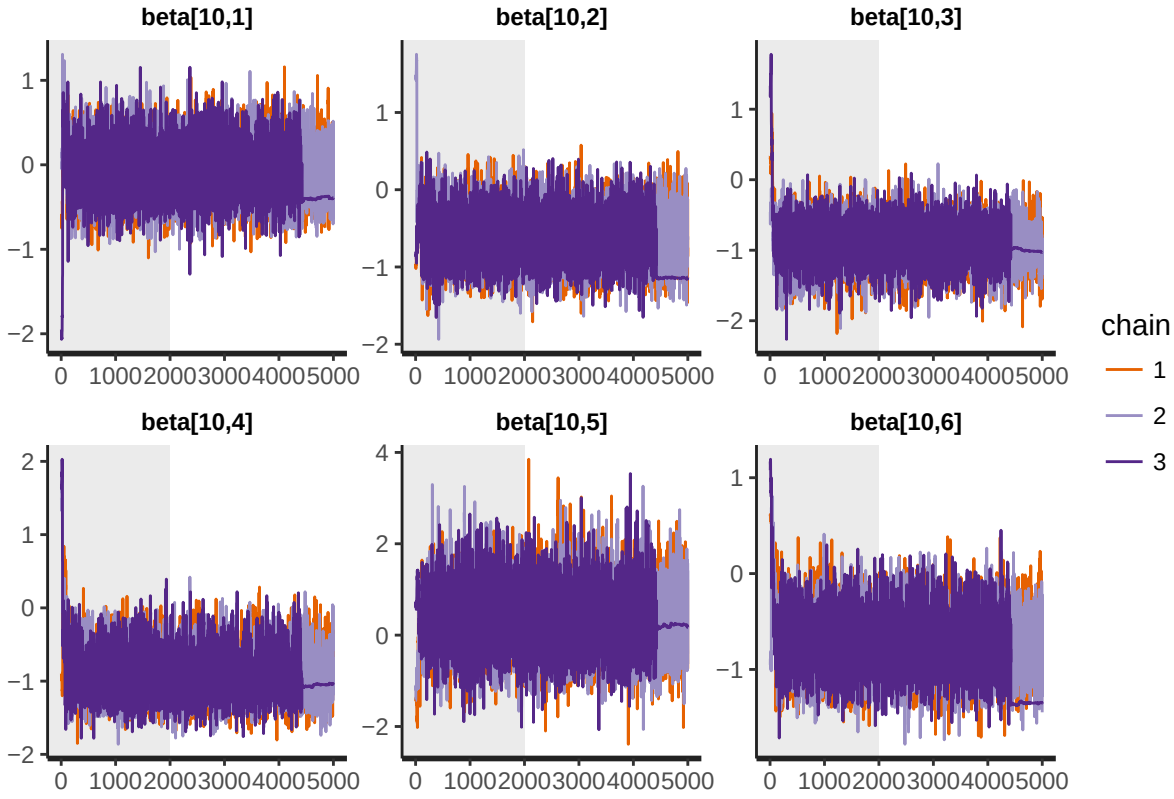

```
traceplot(wine.fit,pars=c("beta[8,1]","beta[8,2]","beta[8,3]","beta[8,4]","beta[8,5]","b
```



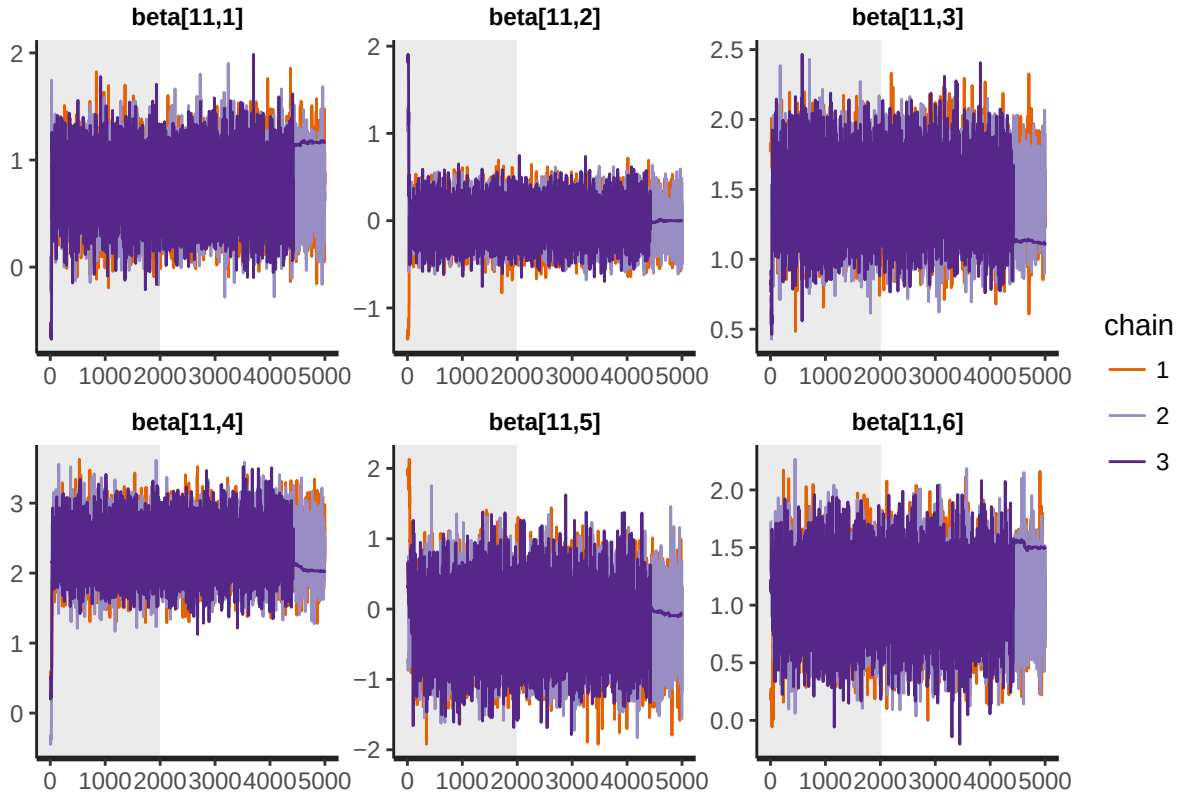

```
traceplot(wine.fit,pars=c("beta[9,1]","beta[9,2]","beta[9,3]","beta[9,4]","beta[9,5]","b
```



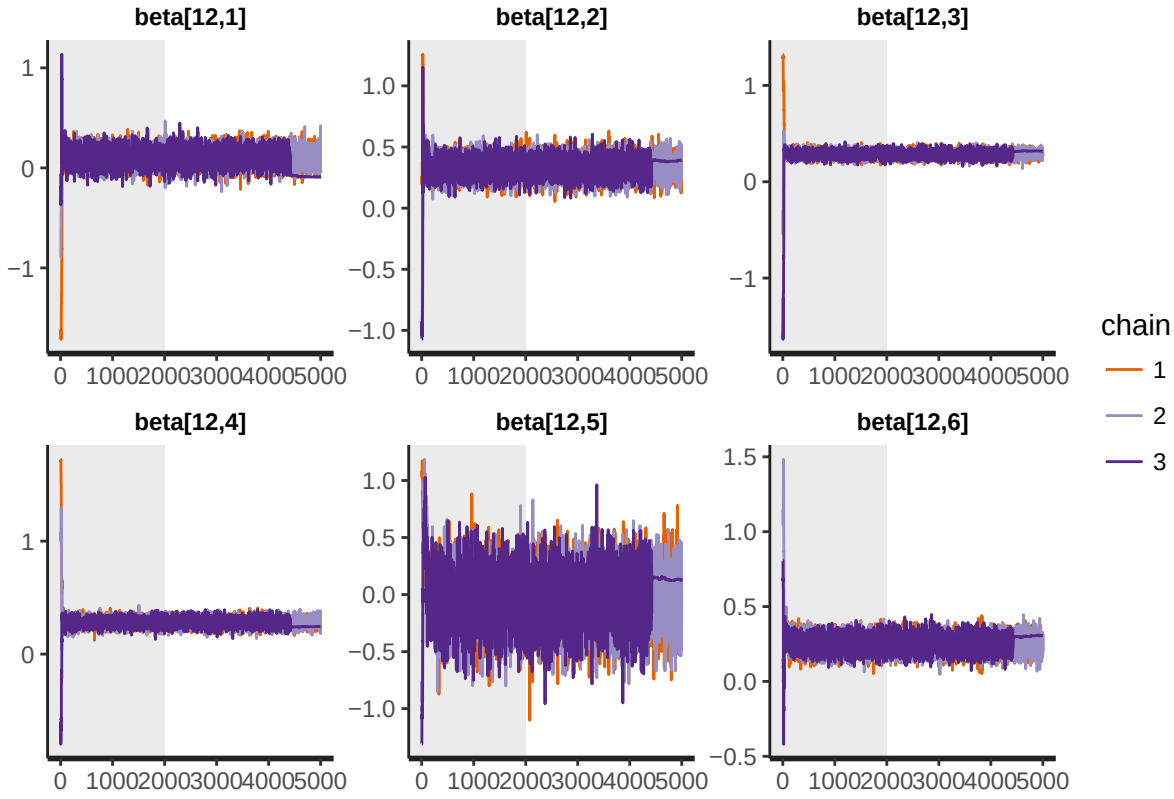

```
traceplot(wine.fit,pars=c("beta[10,1]","beta[10,2]","beta[10,3]","beta[10,4]","beta[10,5
```



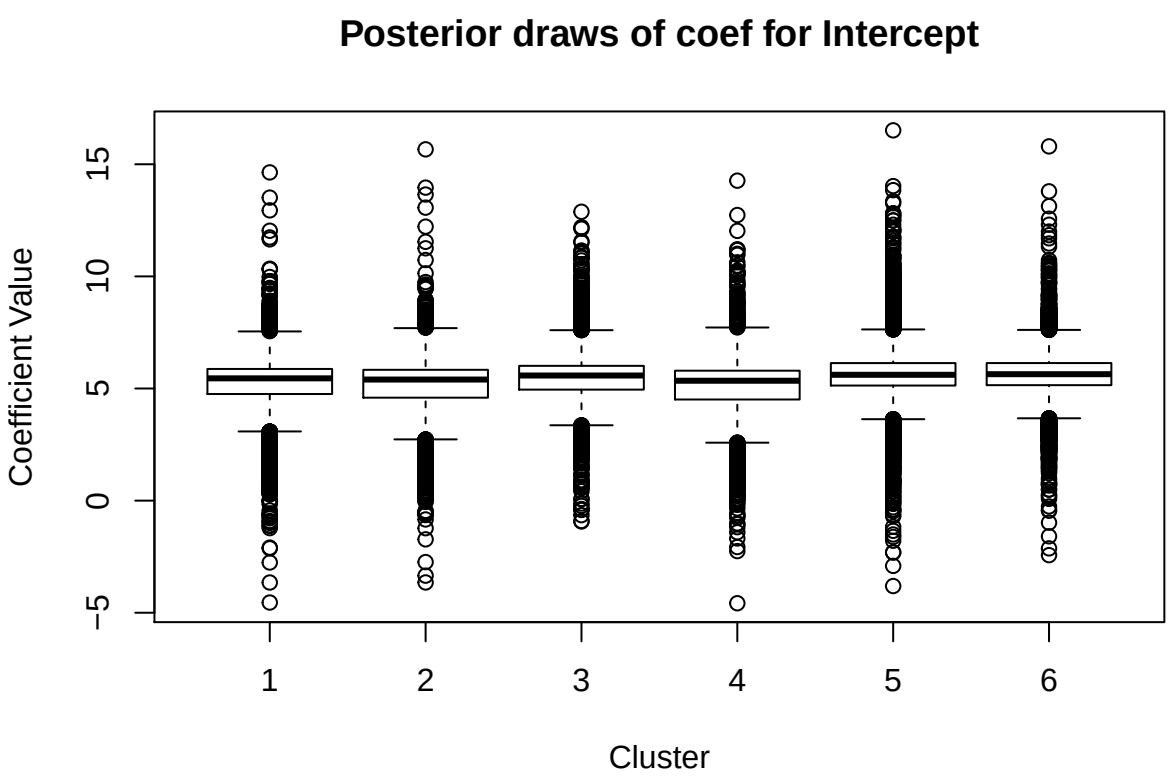

```
traceplot(wine.fit,pars=c("beta[11,1]","beta[11,2]","beta[11,3]","beta[11,4]","beta[11,5
```

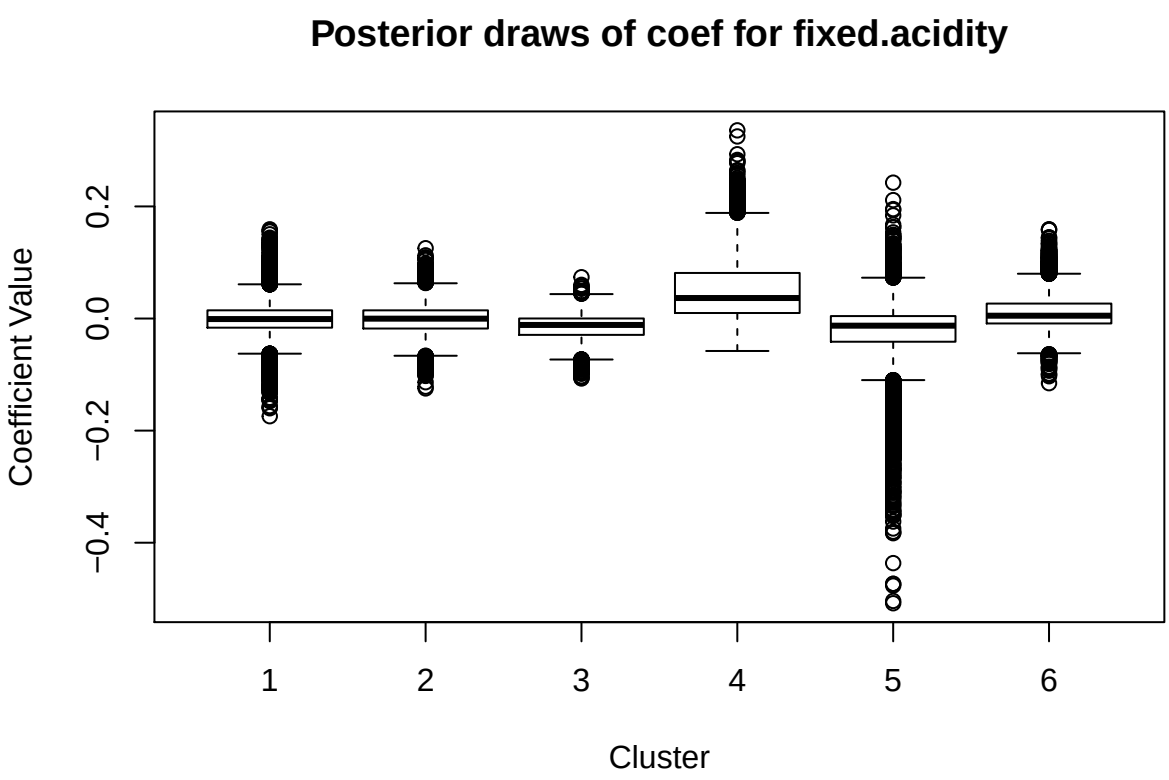


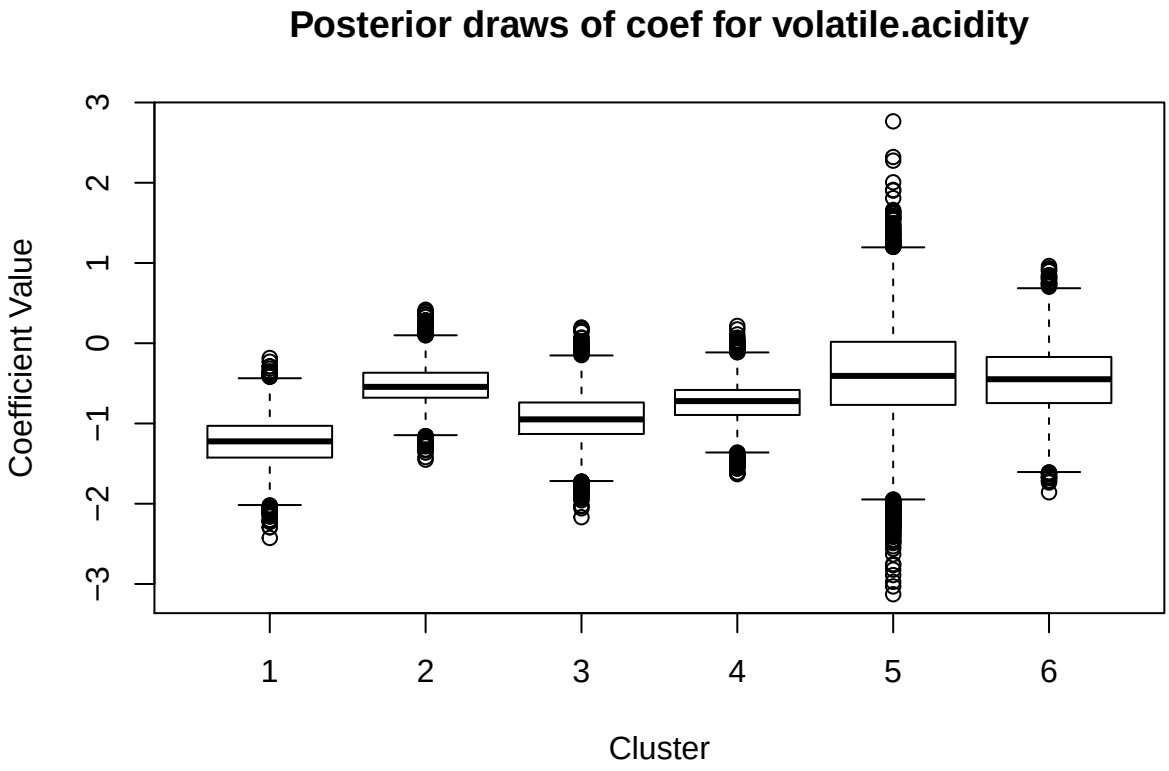

```
traceplot(wine.fit,pars=c("beta[12,1]","beta[12,2]","beta[12,3]","beta[12,4]","beta[12,5
```

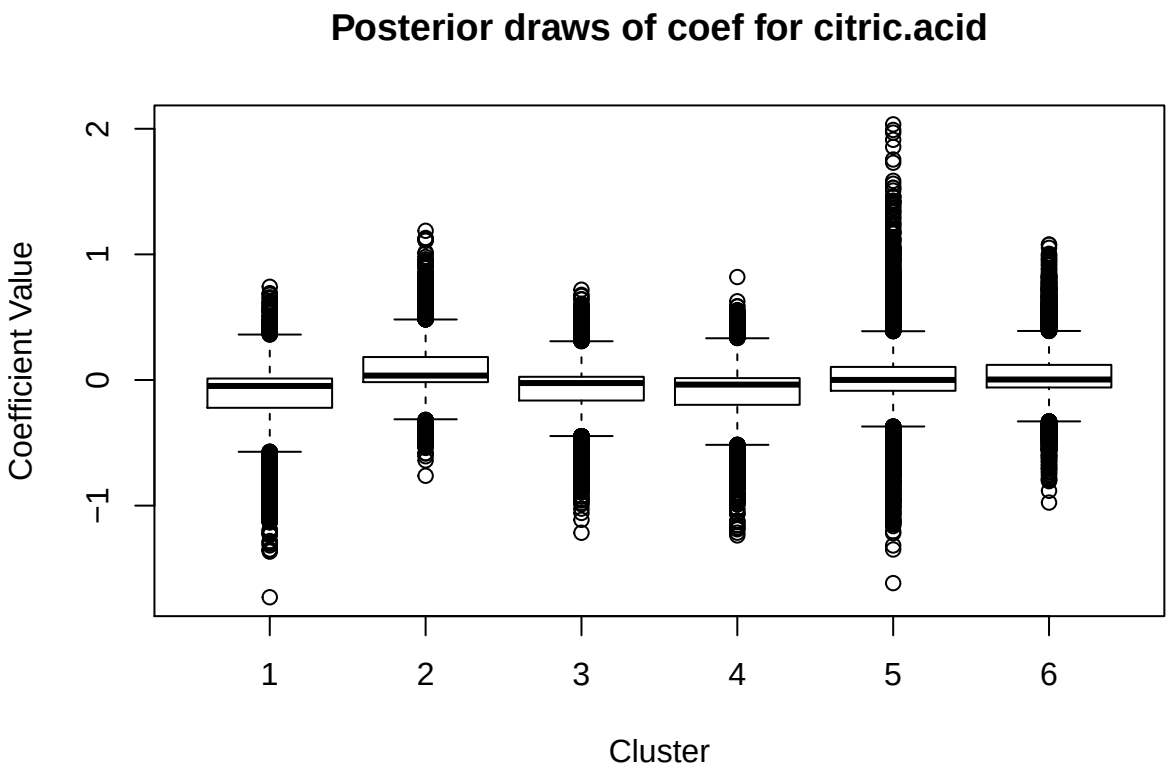


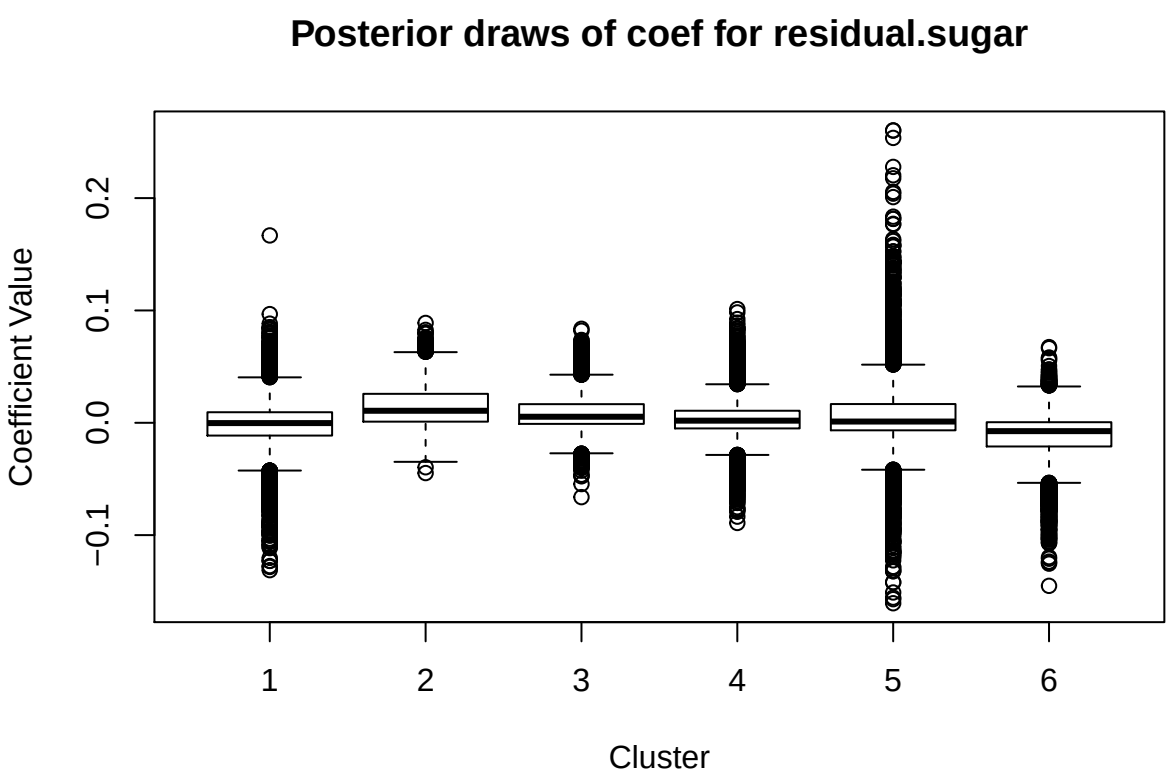

```
# varnames = NULL
# for(i in 1:12) for(j in 1:6){
# varnames = c(varnames, paste("beta[" ,i, ",", j, "]", sep=''))
# }
# for(i in 0:11){
# ttplot = traceplot(wine.fit, pars=varnames[6*i + 1:6],
#   inc_warmup=T)
# ttplot = traceplot(wine.fit, pars=varnames)
# ttplot
# }
wine.fit.extract = extract(wine.fit)
coef.names = c("Intercept",names(x)[1:11])
for(k in 1:12){
  boxplot.matrix(wine.fit.extract$beta[,k,],
      main=paste("Posterior draws of coef for",coef.names[k]),
      xlab="Cluster",ylab="Coefficient Value")
}
```

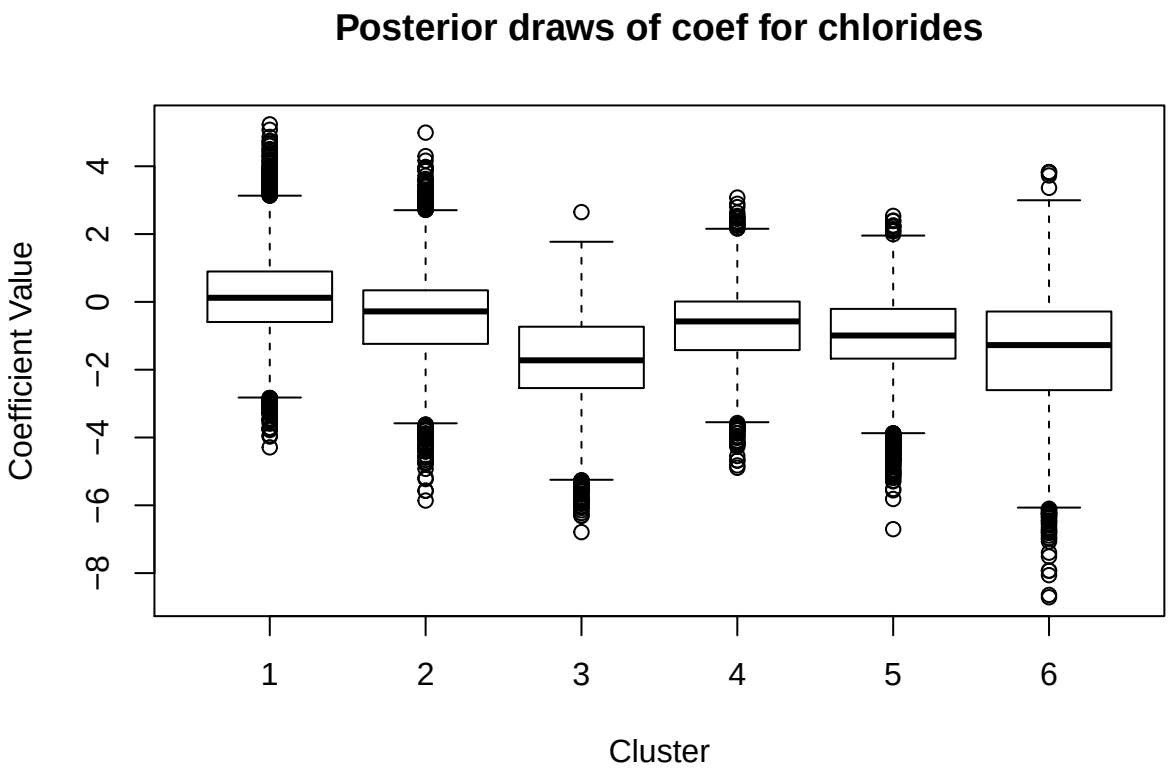

Posterior draws of coef for Intercept
Coefficient Value
−5
0
5
10
15
1
2
3
4
5
6
Cluster

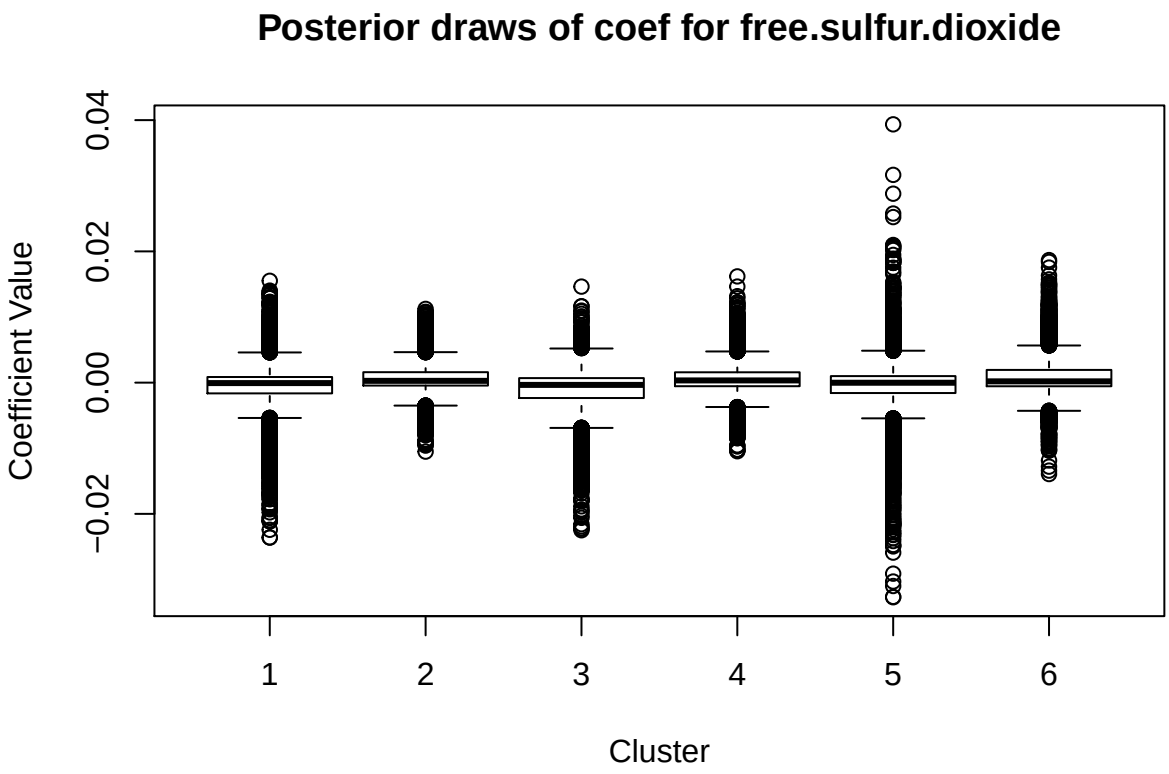

Posterior draws of coef for fixed.acidity
Coefficient Value
0.2
0.0
−0.2
−0.4
1
2
3
4
5
6
Cluster

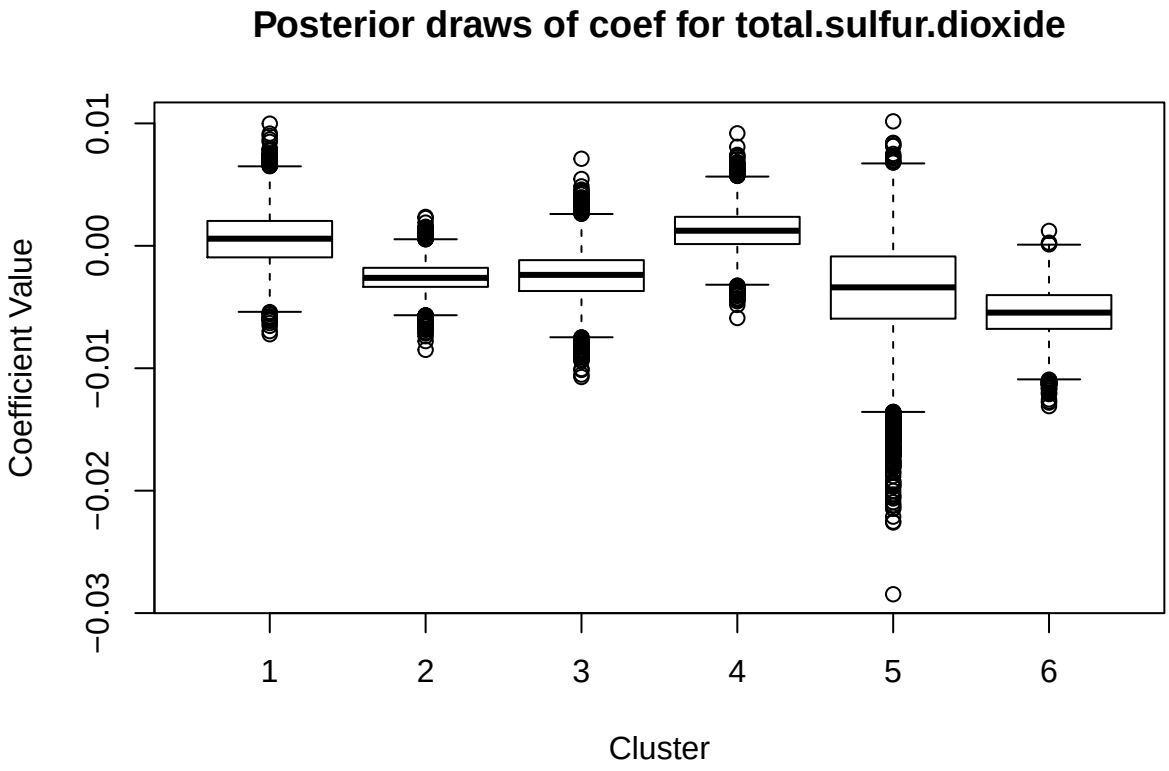

Posterior draws of coef for volatile.acidity
Coefficient Value
3
2
1
0
−1
−2
−3
1
2
3
4
5
6
Cluster

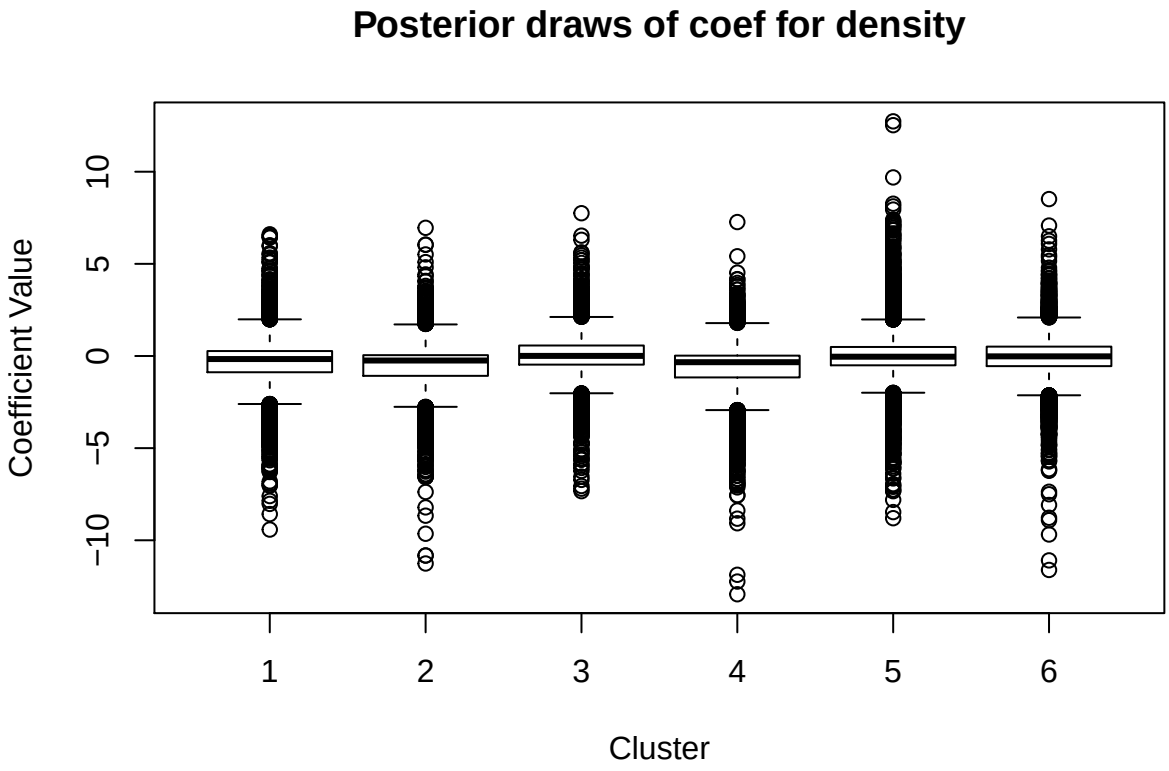

Posterior draws of coef for citric.acid
Coefficient Value
2
1
0
−1
1
2
3
4
5
6
Cluster

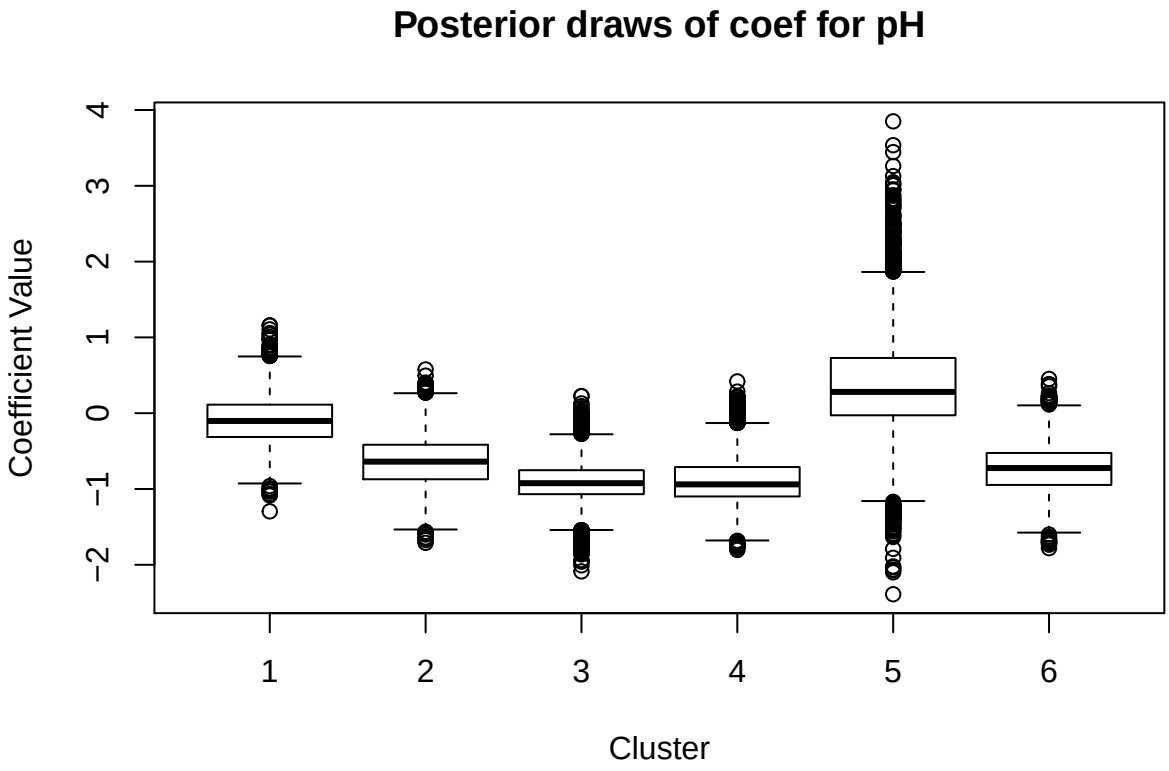

Posterior draws of coef for residual.sugar
Coefficient Value
−0.1
0.0
0.1
0.2
1
2
3
4
5
6
Cluster

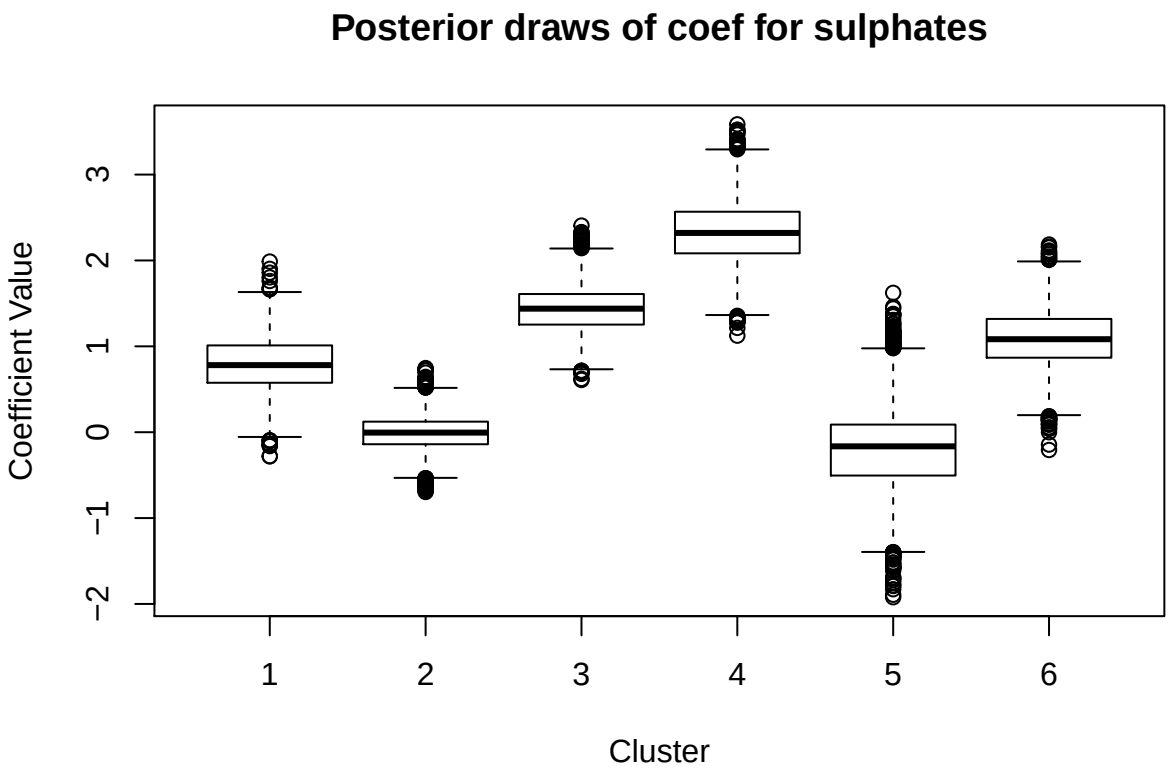

Posterior draws of coef for chlorides
Coefficient Value
4
2
0
−2
−4
−6
−8
1
2
3
4
5
6
Cluster

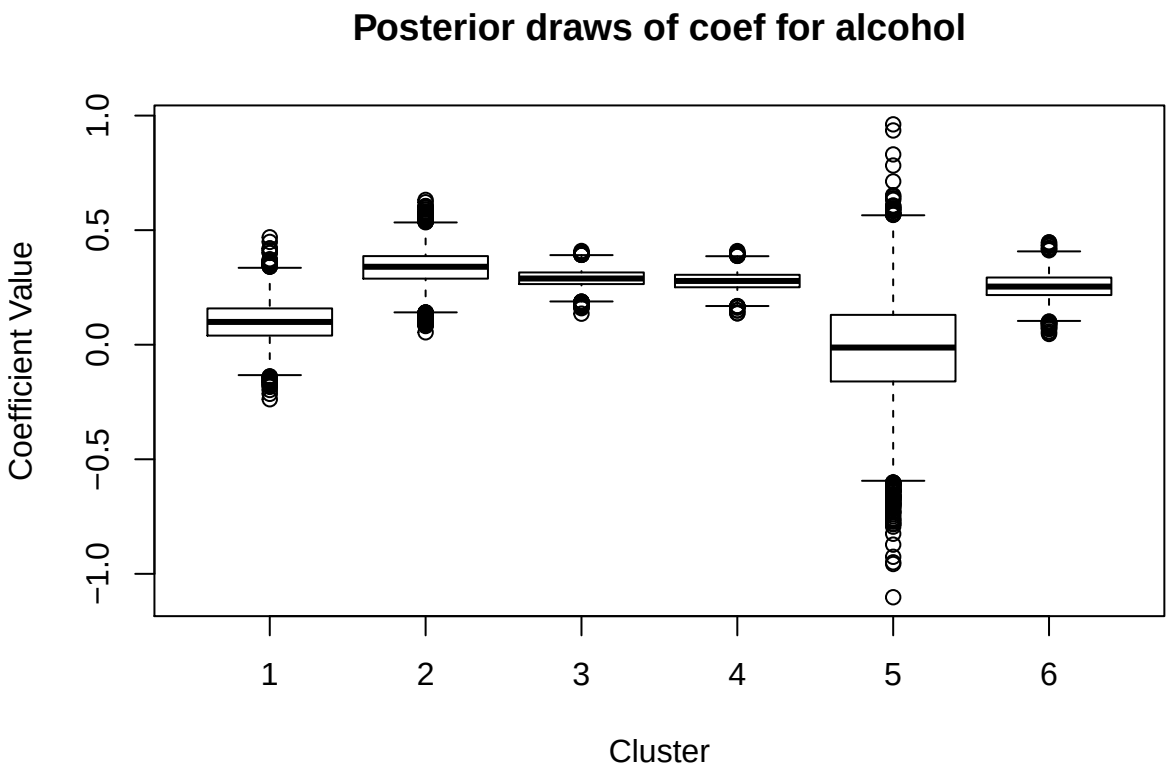

Posterior draws of coef for free.sulfur.dioxide
Coefficient Value
0.04
0.02
0.00
−0.02
1
2
3
4
5
6
Cluster


Posterior draws of coef for total.sulfur.dioxide
Coefficient Value
0.01
0.00
−0.01
−0.02
−0.03
1
2
3
4
5
6
Cluster


Posterior draws of coef for density
Coefficient Value
10
5
0
−5
−10
1
2
3
4
5
6
Cluster


Posterior draws of coef for pH
Coefficient Value
4
3
2
1
0
−1
−2
1
2
3
4
5
6
Cluster


Posterior draws of coef for sulphates
Coefficient Value
−2
−1
0
1
2
3
1
2
3
4
5
6
Cluster



The posterior sampler in Stan was run for 5000 iterations, with the first 2000 iterations being the burn-in period (fewer iterations is probably fine too), and with 3 parallel chains. Trace plots of all the coefficients indicate convergence. The boxplots of the posterior-simulated coefficients mostly do not vary appreciably across clusters, though for some variables there are notably differences. Variables including `total.sulfur.dioxide`, `pH`, `alcohol`, and especially `sulphates` show effects on mean quality that vary noticeably by cluster. For example, wines in cluster 4 have a strong positive relationship between quality and sulphates concentration, whereas wines in cluster 5 have a slightly negative relationship between quality and sulphates concentration. It appears that the hierarchical linear model was successful in separating out the effects of different physio-chemical effects on quality by different groupings of the data.