Appendix I: A Benchmark Data Set for Logistic Reduced Rank Regression and its analysis

Mark de Rooij

Preamble

Load necessary libraries

```
library(mvtnorm)
library(ggplot2)
```

Population

We generate a sample from a population model defined by the logistic reduced rank model. The sample has 1000 observations, 4 predictors and 3 response variables. The predictors are simply drawn from a multivariate normal distribution with mean zero and covariance matrix equal to an identity matrix (i.e., the variables are independent and have variance equal to one). For reproducibility, we set a seed.

```
N = 1000
P = 4
R = 3
set.seed(1234)
X = rmvnorm(N, rep(0,4), diag(4))
m = runif(R, min = -1, max = 1)
V = matrix(c(1,0,
             .5, .25,
```

```
              -.25, .5), 3, 2, byrow = TRUE)
V = V/sqrt(rowSums(V^2))
B = matrix(c(1, 1,
             -1, 1,
             1.25, -.75,
             -.75, -1.25), 4, 2, byrow = T)
U = X %*% B
```

The responses are drawn from the binomial distribution. Therefore, we first need to compute the canonical form, take the logistic transform to obtain probabilities and

```
theta = outer(rep(1, N), m) + X %*% B %*% t(V)
pi = plogis(theta)
Y = matrix(NA, N, R)
for(r in 1:R){ Y[, r] = rbinom(N, 1, pi[, r]) }

colnames(X) = c("x1", "x2", "x3", "x4")
colnames(Y) = c("y1", "y2", "y3")
```

LRRR analysis

Before analyzing the data, we throw away everything except **X** and **Y.** We also load the functions for analysis.

```
rm(list=ls()[! ls() %in% c("Y","X")])
source("~/surfdrive/LogitMDA/lrrr-diagnostics/lrrr.R")
source("~/surfdrive/LogitMDA/lrrr-diagnostics/diagnosis.R")
source("~/surfdrive/LogitMDA/lmap-package/new/R/procx.R")
source("~/surfdrive/LogitMDA/lrrr-diagnostics/plot.lrrr.R")
N = nrow(X)
P = ncol(X)
R = ncol(Y)
```

To analyse the data using a logistic reduced rank model. we call the function lpca() from the lmap-package. Reduced rank regresssion can be considered a constraint or restricted PCA, where the object points lie in the column space of **X**.

```
lrrr.out = lpca(Y = Y, X = X, S = 2)
```

The fitted model has deviance 2698.78. Its estimated weight matrix is

```
round(lrrr.out$B, digits = 2)
```

```
      [,1]  [,2]
[1,] -0.57 -0.44
[2,]  0.48 -0.50
[3,] -0.57  0.45
[4,]  0.44  0.59
```

and the estimated loading matrix is

```
round(lrrr.out$V, digits = 2)
```

```
      [,1]  [,2]
[1,] -2.04  0.11
[2,] -1.86 -0.76
[3,]  0.62 -1.89
```

Finally, we can make a biplot.

```
plot.lpcah(lrrr.out, ocol = "red")
```

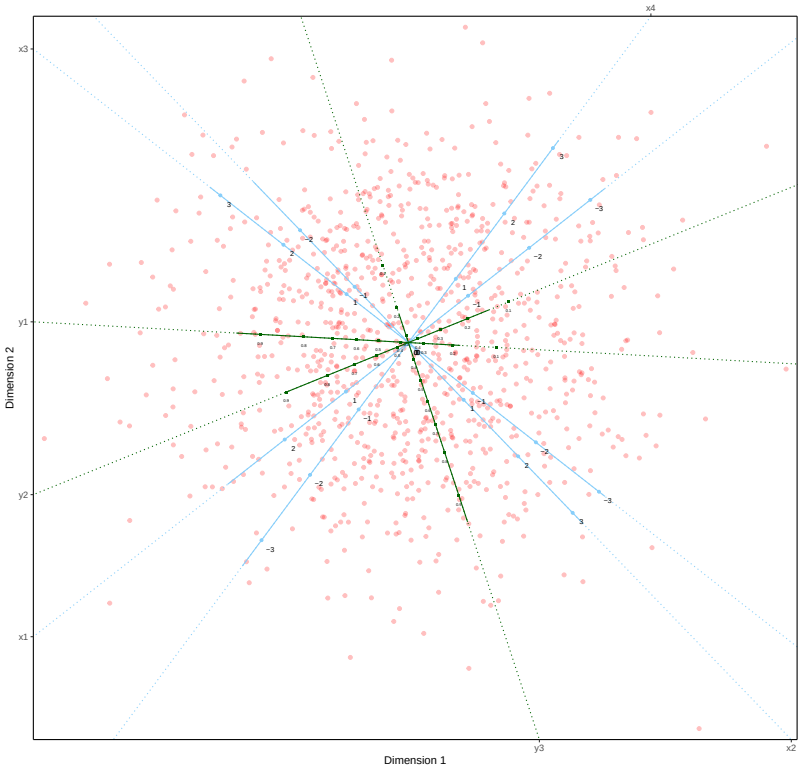


Figure 1: Triplot of Benchmark data

In the biplot, we already visualized the deviance residuals (see next Section) by the area of the points. The larger the point, the higher the contribution of that observation to the total deviance.

Diagnosis

Fit measures

First, we look at fit measures. We already saw that the deviance of the fitted model is 2698.78. The total deviance can be partitioned in contributions of the individual observations, that is, $\mathcal{D} = \sum_{i,r} d_{ir}^2$. We will give the definition of these "cell-wise' contributions in the next Section.

The elements can be used to define contributions of misfit for the individuals, that is $\mathcal{D} = \sum_i d_i$, where $d_i = \sum_r d_{ir}^2$ and contributions of response variables, that is $\mathcal{D} = \sum_r d_r$, where $d_r = \sum_i d_{ir}^2$. To get to fit measures, from the deviance of the model, we can compute the so-called McFadden R^2. For response variable r, this measure is defined as

$$R^2 = 1 - \frac{d_r}{d_0},$$

where d_0 is the deviance of an intercept only logistic regression model for response variable r. A similar overall R^2 measure is given by

$$R_m^2 = 1 - \frac{\mathcal{D}}{\mathcal{D}_0}$$

where $\mathcal{D}_0$ is the sum of the null-deviances over all response variables. We developed an R-function called fit.lrrr() for assessing the fit of the model.

```
fit = fit.lrrr(lrrr.out)
```

For the benchmark data set, the overall R_m^2 is 0.34. The R^2 for the three response variables is

```
round(fit$R2.variables, digits = 2)
```

```
  y1   y2   y3
0.34 0.33 0.34
```

We, can also verify the Quality of Representation. This is a measure of information loss due to the rank constraint. It compares the fit per response variable (d_r) with the fit of a logistic regression of response variable r with the same predictor variables. The latter is equal to the fit of the response variable in a full-dimensional model (i.e., when the dimensionality is equal to $\min(P, R)$). The quality of representation for our analysis is

```r
round(fit$QoR, digits = 2)
```

```
y1 y2 y3
 1  1  1
```

which is very good, that is no loss due to the reduced rank. This was expected, as the data were generated with the LRRR model in two dimensions.

Residuals

Raw residuals are the most simple ones, and defined by

$$e_{ir} = y_{ir} - \hat{\pi}_{ir}$$

These residuals are bounded between -1 and 1, they are positive when $y_{ir} = 1$ and negative for $y_{ir} = 0$.

Pearson residuals scale the raw residuals, and are defined as

$$\frac{e_{ir}}{\sqrt{\hat{\pi}_{ir}(1 - \hat{\pi}_{ir})}}.$$

Deviance residuals are the individual contributions of the observations to the total deviance. Usually, they are defined as

$$d_{ir} = s(e_{ir})\sqrt{-2\log(1 + \exp(-q_{ir}\theta_{ir})},$$

where $s(x)$ gives the sign of x. Using these residuals we have $\mathcal{D} = \sum_{i,r} d_{ir}^2$.

We are often interested in the fit of persons, so therefore we sum the (squared) residuals over r. In Figure 1 we already showed an analysis, where the points for observations are larger for those with higher contributions to the deviance. The area of a point is equal to $\sum_r d_{ir}^2$.

We developed an R-function called residuals.lrrr() for computation of these residuals of the model.

To get a better feeling of these residuals, we can look a bit more into their distribution. Let us summarize the raw residuals

```
residuals = residuals.lrrr(lrrr.out)
summary(residuals$raw)
```

```
      y1                  y2                  y3
Min.   :-0.9553847   Min.   :-0.9094470   Min.   :-0.951836
1st Qu.:-0.2101240   1st Qu.:-0.2451220   1st Qu.:-0.198063
Median :-0.0326568   Median :-0.0192124   Median :-0.042342
Mean   :-0.0000755   Mean   :-0.0000227   Mean   :-0.000081
3rd Qu.: 0.2615428   3rd Qu.: 0.2370372   3rd Qu.: 0.217613
Max.   : 0.9698485   Max.   : 0.9859904   Max.   : 0.977353
```

the Pearson residuals

```
summary(residuals$pearson)
```

```
      y1                 y2                  y3
Min.   :-4.62751   Min.   :-3.169110   Min.   :-4.445503
1st Qu.:-0.51577   1st Qu.:-0.569840   1st Qu.:-0.496970
Median :-0.18374   Median :-0.139958   Median :-0.210273
Mean   :-0.02075   Mean   : 0.007746   Mean   : 0.004887
3rd Qu.: 0.59513   3rd Qu.: 0.557387   3rd Qu.: 0.527390
Max.   : 5.67150   Max.   : 8.389252   Max.   : 6.569282
```

and the deviance residuals

```
summary(residuals$deviance)
```

```
      y1                 y2                  y3
Min.   :-2.49386   Min.   :-2.19172   Min.   :-2.46299
1st Qu.:-0.68685   1st Qu.:-0.74993   1st Qu.:-0.66441
Median :-0.25769   Median :-0.19697   Median :-0.29416
Mean   :-0.03849   Mean   :-0.01189   Mean   :-0.04278
3rd Qu.: 0.77871   3rd Qu.: 0.73559   3rd Qu.: 0.70058
Max.   : 2.64633   Max.   : 2.92165   Max.   : 2.75235
```

Hat values

In linear and logistic regression, so-called hat values are important diagnostics. The hat-matrix in linear regression is defined as

$$\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$$

and its called the hat matrix, as

$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{y},$$

that is, the matrix puts the hat on the response variable. The values in the diagonal cells of the hat matrix, h_{ii} are often used as leverage measures. Furthermore, these values play an iimportant role in the definition of influence statistics. In logistic regression the situation is a bit more complex as logistic regression uses an iterative procedure and therefore there is no matrix that literaly puts the hat on the response. Usually, the hat matrix in logistic regression is defined as

$$\mathbf{H} = \mathbf{V}^{1/2}\mathbf{X}(\mathbf{X}'\mathbf{V}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{1/2},$$

where $\mathbf{V}$ is a diagonal matrix with values $v_{ii} = \hat{\pi}_i(1 - \hat{\pi}_i)$.

For logistic reduced rank regression, the situation is even more complex because we have multiple response variables. However, we can use the ideas of logistic regression and work per response variable. Therefore, define $v_{ii}^{(r)} = \hat{\pi}_{ir}(1 - \hat{\pi}_{ir})$ as the diagonal elements of the matrix $\mathbf{V}_r$. Then a hat matrix for the r-th response variable can be defined as

$$\mathbf{H}_r = \mathbf{V}_r^{1/2}\mathbf{X}(\mathbf{X}'\mathbf{V}_r\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}_r^{1/2}.$$

We obtain such a matrix for every response variable $r = 1, \ldots, R$. As we usually only take into account the diagonal values of the hat matrix, $h_{ii}^{(r)}$, we now obtain R such measures. These R values can be summed to get a measure of leverage for an individual observation.

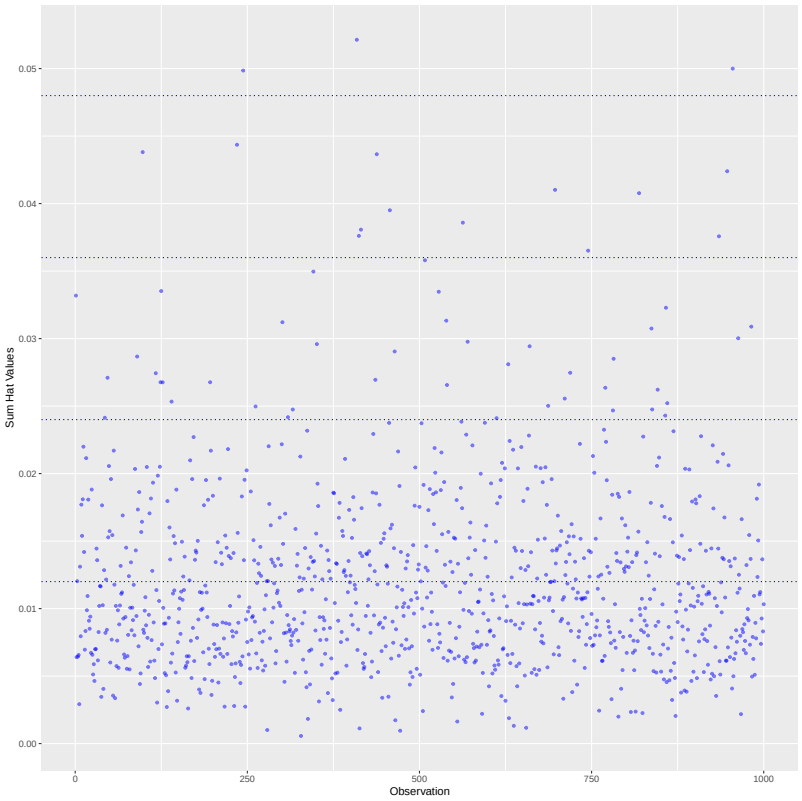
We developed an R-function called hat.lrrr() for estimating these hat values.

```
hat = hat.lrrr(lrrr.out)
head(hat)
```

```
  idx           y1          y2           y3
1   1 0.0115184205 0.007661882 0.0140045008
2   2 0.0017170851 0.002060027 0.0026254566
3   3 0.0042740264 0.004028098 0.0037291677
4   4 0.0026483343 0.002375412 0.0014184356
5   5 0.0026246662 0.001404752 0.0025370817
6   6 0.0009741415 0.001050361 0.0008963307
```

For each response variable, the sum of the hat values equals the number of predictors (4 for the benchmark data set). As we have 3 response variables the total sum is $3 \times 4 = 12$, or more generally $P \times R$. The average therefore is PR/N. For logistic models, researchers adviced to inspect hat values that exceed two or three times this average. Horizontal lines showing these thresholds and a four-times-average threshold are added to the participant-hat scatterplot (below). We see that even for perfect data quite some participants exceed these values.

```
hat$sum = rowSums(hat[, -1])
ggplot(hat, aes(x = idx, y = sum)) +
  geom_point(size = 1, col = "blue", alpha = 0.5) +
  geom_hline(yintercept = c(P*R/N, 2*P*R/N, 3*P*R/N, 4*P*R/N),
             col = "darkblue", lty = "dotted") +
  labs(x = "Observation", y = "Sum Hat Values")
```



Influential Points

From the residuals, we can see which observations are well fitted and which not. Another aspect is whether a particular observation is *influential*. Influence is often the result of high leverage and discrepancy. We investigate influence by leaving out one observation, refitting the model and look at specific changes. We do so for each and every observations. Whereas earlier papers gave approximations to such leave-one-out statistics, in the era of high performance computing we think such approximations are not necessary anymore.

Our focus will be on changes in deviance, changes in the regression weights, and changes in the loadings. For the change in deviance, we fit the LRRR model on the data set with observation i left out, derive its parameter estimates and subsequently

compute the deviance for the whole data set with these estimated parameters.

For the change in regression weights we compute

$$\delta_B = \|\hat{\mathbf{B}} - \hat{\mathbf{B}}_{-i}\|^2$$

and a similar measure can be defined for the loadings, which will de denoted by δ_V. These deletion statistics can be computed using the function influence.lrrr that we developed.

```
deletion = influence.lrrr(lrrr.out)
```

These influence statistics can be plotted against case number. For the benchmark data set, the change in deviance plot is given in Figure 2, the change in parameter estimates is shown in Figure 3.

```
ggplot(deletion, aes(x = idx, y = deviance)) +
  geom_point(size = 1, col = "blue", alpha = 0.5) +
  labs(x = "Observation",
       y = "Change in Deviance")
```

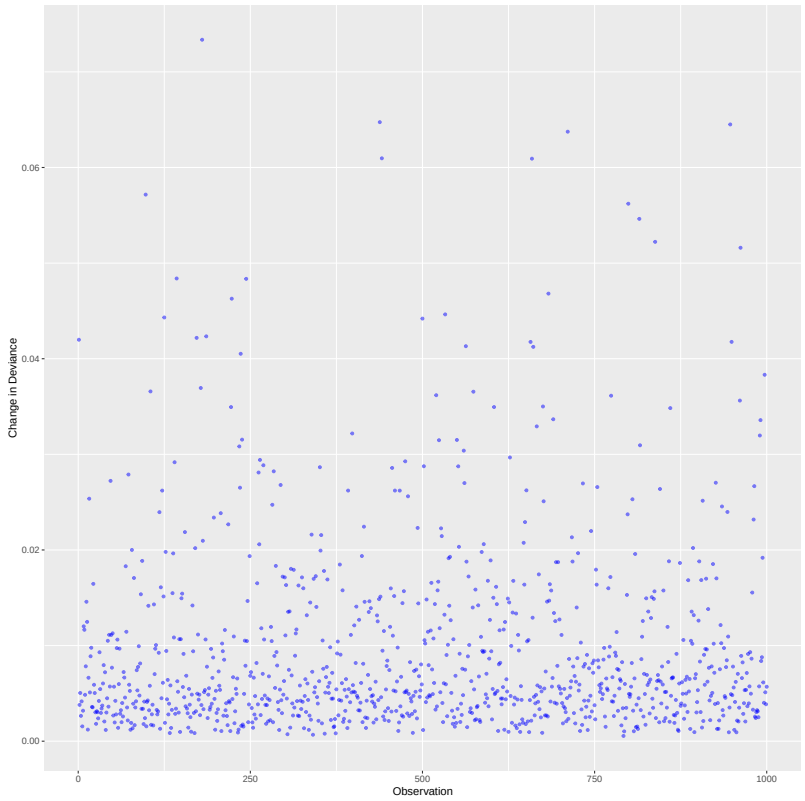


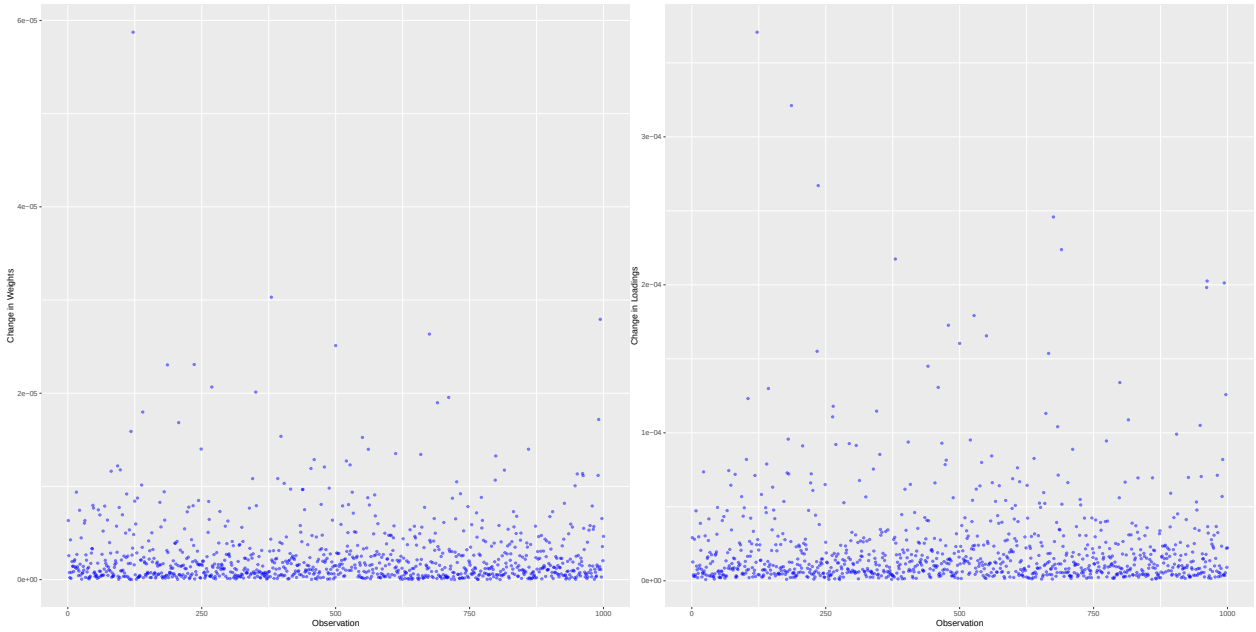
Figure 2: Change in deviance for left out observations

In the change in deviance plot, we can see that most changes are really small, that is smaller than 0.07. This number, however, also depends on the sample size (true?), so cannot be easily used for other samples.

```
ggplot(deletion, aes(x = idx, y = B)) +
  geom_point(size = 1, col = "blue", alpha = 0.5) +
  labs(x = "Observation",
       y = "Change in Weights")

ggplot(deletion, aes(x = idx, y = V)) +
  geom_point(size = 1, col = "blue", alpha = 0.5) +
  labs(x = "Observation",
       y = "Change in Loadings")
```

In the change in parameter plots, we can verify that also these



(a) Change in Weights

(b) Change in Loadings

Figure 3: Change in Parameter Estimates

numbers are really small when the data strictly follows the model.

Non-linearity

In the reduced rank model, we make a linearity assumption. For linear and logistic regression, so called component plus residual plots are often used to verify discrepancies from linearity. For our logistic regression model, we can make a similar plot. We obtain one plot per pair of predictor and response variable. The component in this case is defined as $x_{ip}a_{pr}$ with $a_{pr} = \mathbf{b}_p'\mathbf{v}_r$. The working residual follows from our algorithm. That is, in each step of the algorithm a working response is defined as this component plus four times the raw residual $4(y_{ir} - \hat{\pi}_{ir})$. This component with residual is plotted against the predictor variable x_{ip}.

```
A = lrrr.out$B %*% t(lrrr.out$V)

# p = 1
df_partialresidual11 = as.data.frame(cbind(lrrr.out$X[, 1, drop = F],
                                           lrrr.out$X[, 1, drop = F] %*% A[1, 1, drop = F] +
                                             residuals$working[ , 1, drop = FALSE]))
colnames(df_partialresidual11) = c("Predictor", "Partial_Residual")
```

We add two lines to this scatterplot. The first is a linear fit for which we include the confidence bound. The second is a smooth loess fit. If the loess fit is far from the linear fit it suggests nonlinearity. This can easily be done in ggplot. For the first response variable in the benchmark data these plots are shown in Figure 4.

```
ggplot(df_partialresidual11, aes(x = Predictor, y = Partial_Residual)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = lm, formula = y ~ x, se = TRUE, col = "blue") +
  geom_smooth(method = loess,  formula = y ~ x, se = FALSE, col = "red") +
  labs(title = paste("Predictor:", lrrr.out$xnames[1], " ; ",
                     "Response:", lrrr.out$ynames[1]),
       x = "Predictor variable",
       y = "Component + Residual")

ggplot(df_partialresidual21, aes(x = Predictor, y = Partial_Residual)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = lm, formula = y ~ x, se = TRUE, col = "blue") +
  geom_smooth(method = loess,  formula = y ~ x, se = FALSE, col = "red") +
  labs(title = paste("Predictor:", lrrr.out$xnames[2], " ; ",
                     "Response:", lrrr.out$ynames[1]),
       x = "Predictor variable",
       y = "Component + Residual")

ggplot(df_partialresidual31, aes(x = Predictor, y = Partial_Residual)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = lm, formula = y ~ x, se = TRUE, col = "blue") +
  geom_smooth(method = loess,  formula = y ~ x, se = FALSE, col = "red") +
  labs(title = paste("Predictor:", lrrr.out$xnames[3], " ; ",
                     "Response:", lrrr.out$ynames[1]),
```

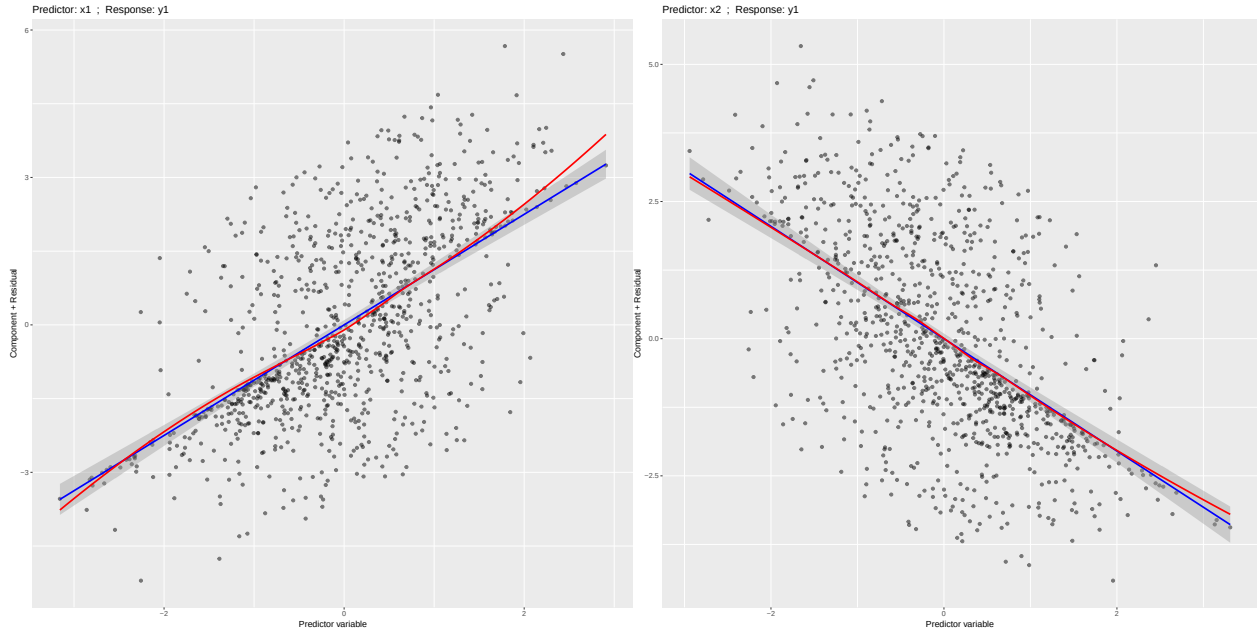
```
        x = "Predictor variable",
        y = "Component + Residual")

ggplot(df_partialresidual41, aes(x = Predictor, y = Partial_Residual)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = lm, formula = y ~ x, se = TRUE, col = "blue") +
  geom_smooth(method = loess,  formula = y ~ x, se = FALSE, col = "red") +
  labs(title = paste("Predictor:", lrrr.out$xnames[4], " ; ",
                     "Response:", lrrr.out$ynames[1]),
        x = "Predictor variable",
        y = "Component + Residual")
```

For the second response variable, we again have four component plus residual plots. These are given in the following plots

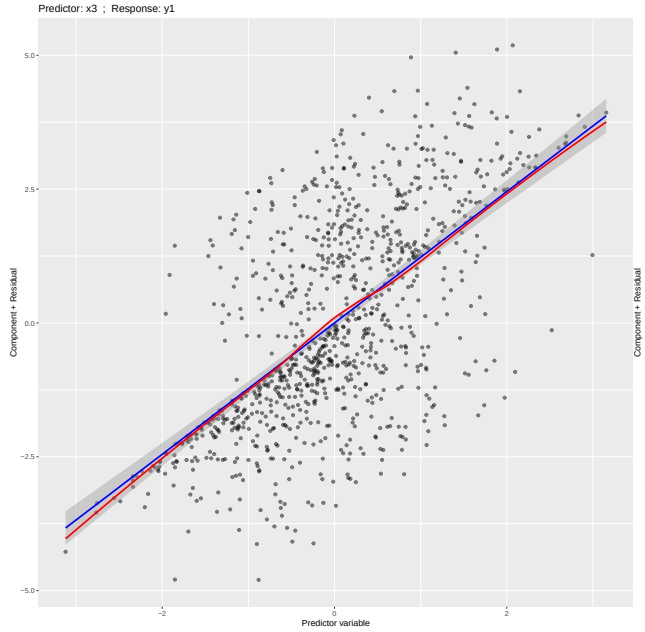
Finally, for the third response variable the four component plus residual plots are

We see in all plots that the loess curve aligns closely with the linear fit. Sometimes, at the boundary of the grpahs the loess curve is outside the confidence bound. As these data are generated by the model, this departure is something to consider when interpreting the corresponding plots from empirical data.

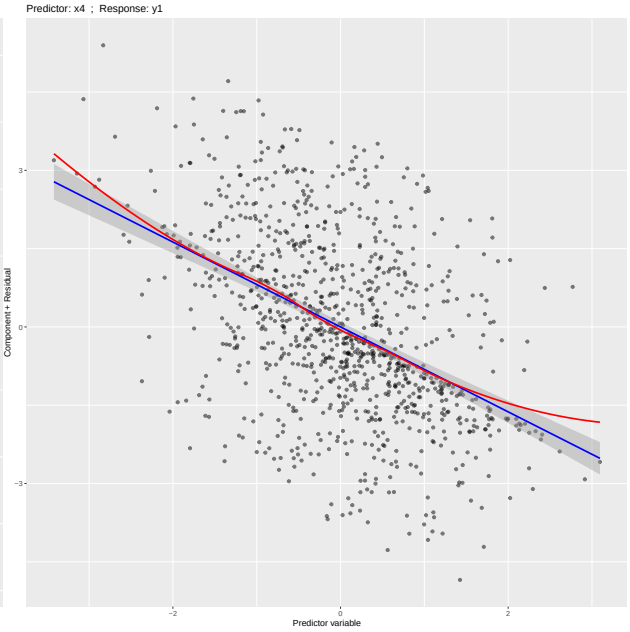


(a) First predictor

(b) Second predictor

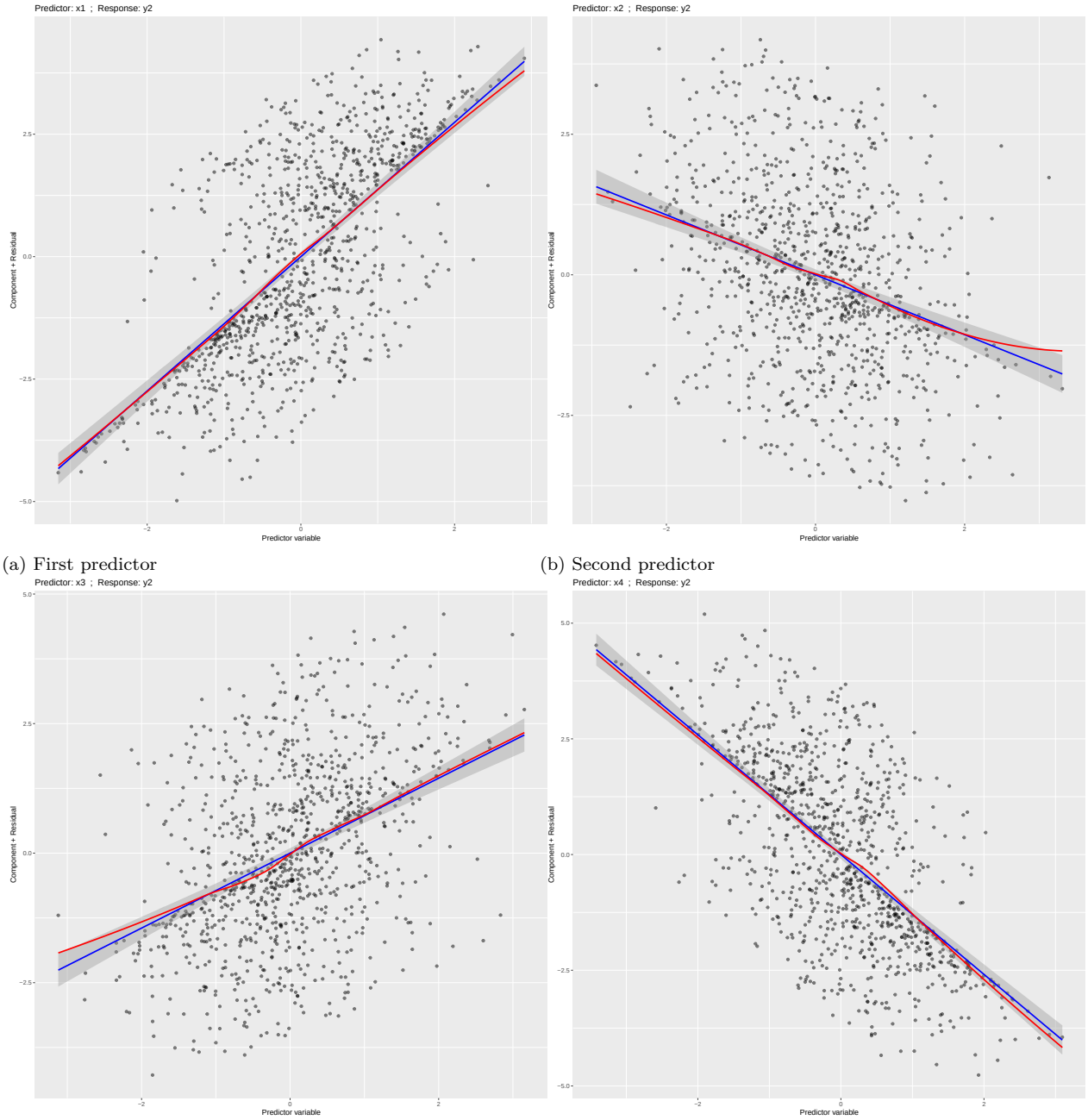


(c) Third predictor



(d) Fourth predictor

Figure 4: Component plus residual plots for 1st response variable



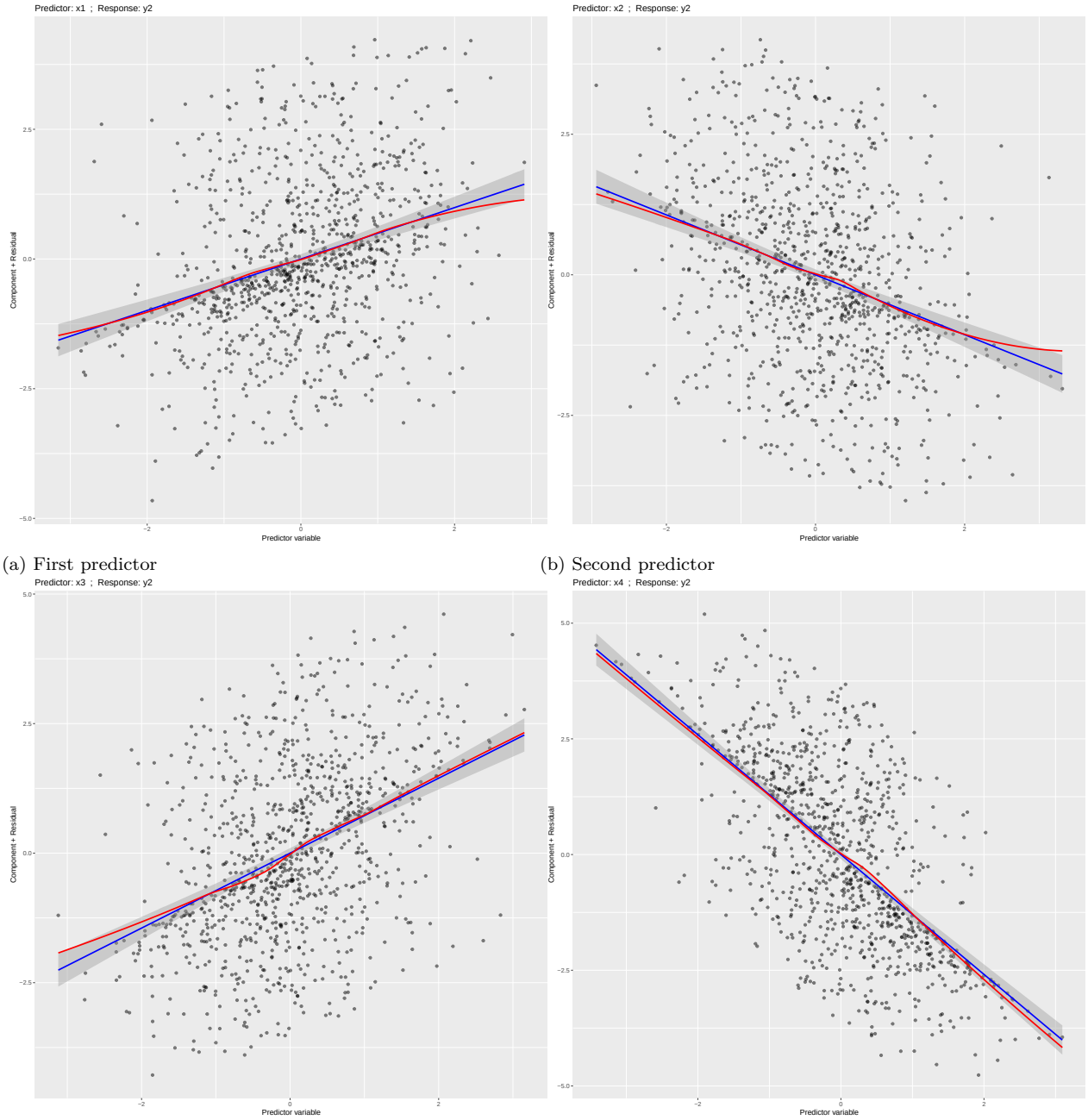
(a) First predictor

(b) Second predictor

(c) Third predictor

(d) Fourth predictor

Figure 5: Component plus residual plots for 2nd response variable



(a) First predictor

(b) Second predictor

(c) Third predictor

(d) Fourth predictor

Figure 6: Component plus residual plots for 3rd response variable