Exam

Nonparametric Statistics, AY 2022/23

June 19, 2023

Instructions

- For all computations based on permutation/bootstrapping, use $B = 1000$ replicates, and $seed = 2022$ every time a permutation/bootstrap procedure is run.
- For Full Conformal prediction intervals, use a regular grid, where, for each dimension, you have $N = 20$ equispaced points with lower bound $\min(data) - 0.25 \cdot range(data)$ and upper bound $\max(data) + 0.25 \cdot range(data)$. Moreover, do not exclude the test point when calculating the conformity measure. Be advised that, except for the number of points, these are the default conditions of the `ConformalInference` R package.
- Both for confidence and prediction intervals, as well as tests, if not specified otherwise, set $\alpha = 0.05$.
- When reporting univariate confidence/prediction intervals, always provide upper and lower bounds.
- For solving the exam, you must use one of the templates previously provided and available here. Particularly, **for each question** you are required to report:
  - *Synthetic description of assumptions, methods, and algorithms*: which methodological procedure you intend to use to answer the question, succinctly describing the main theoretical characteristics of the chosen approach, and why it is suitable for the analytical task at hand,
  - *Results and brief discussion*. the actual result of the procedure applied to the data at hand, including any requested comment, output and plot.
- Data for the exam can be found at this link.

Exercise 1

Simon Le Vanten is a French winemaker whose aim is to produce the greatest wine in all of France. Knowing that he must rely on the most sophisticated analytical techniques to achieve his goal he collects $N = 178$ wine samples, measuring three chemical properties of his wines, namely levels of alcohol (pure content in percentage), proline (milligrams per deciliter), and calcium (milligrams per deciliter). The resulting samples are contained in the `df_1.Rds` file. To help Simon achieve his goal he asks you to:

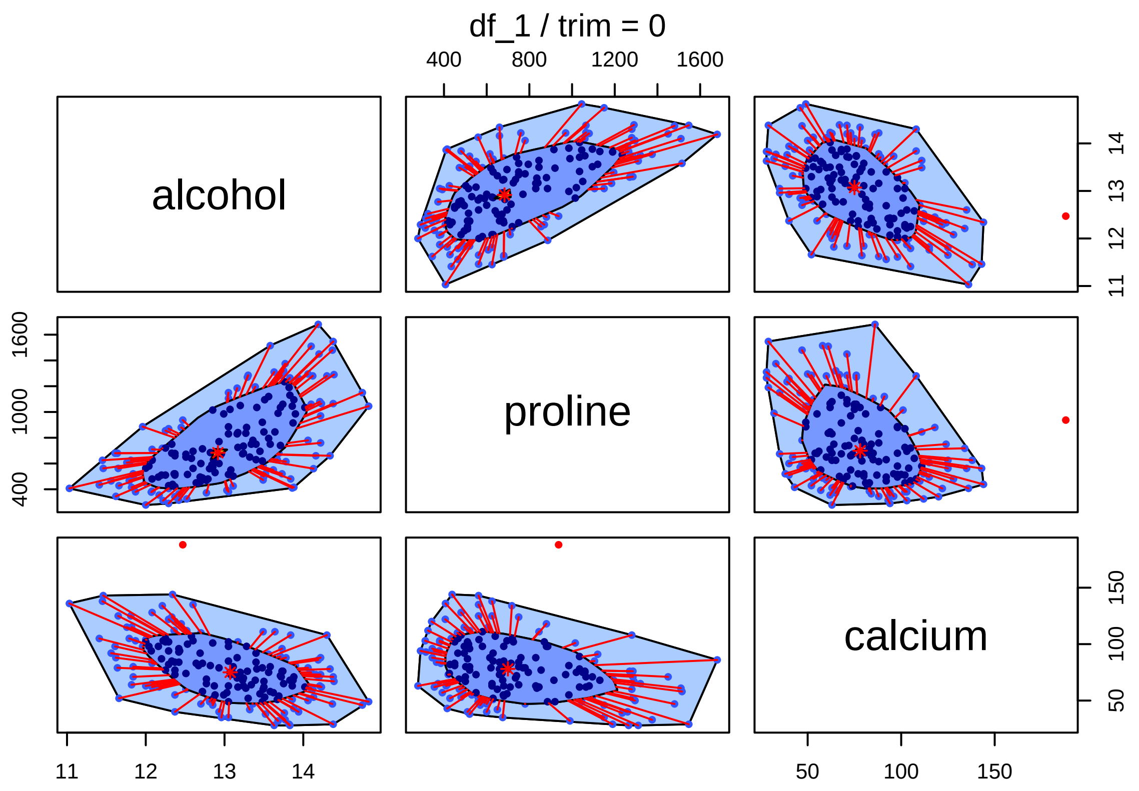
1. Provide the sample Mahalanobis median of the wine samples

```
df_1 = readRDS(here("2023-06-19/data/df_1.Rds"))
N <- nrow(df_1)
p <- ncol(df_1)
(maha_median <- depthMedian(df_1,depth_params = list(method='Mahalanobis')))
```

```
## alcohol proline calcium
##   12.93  770.00   79.00
```

2. Plot a bagplot matrix of the three collected variables, and determine a vector of row indexes identifying the wine samples that are outliers at least in one panel of the bagplot matrix.

```
bagplot_matrix <- aplpack::bagplot.pairs(df_1)
```



```
# Just one outlier
bagplot_13 <- compute.bagplot(df_1[, 1], df_1[, 3])
ind_out <-
  which(apply(df_1[, c(1, 3)], 1, function(x)
    all(x %in% bagplot_13$pxy.outlier)))
ind_out
```

```
## 96
## 96
```

3. Employing a naive bootstrap approach, provide a reverse percentile confidence interval with $\alpha = 0.01$ for each component of the Mahalanobis Median of the wine samples.

```
N <- nrow(df_1)

B <- 1000
maha_median_boot = matrix(nrow = B,ncol = p)

set.seed(2022)

for(b in 1:B) {
  boot_id <- sample(x = 1:N,size = N,replace = TRUE)
  df_boot <- df_1[boot_id,]
  maha_median_boot[b,] <- depthMedian(df_boot,depth_params = list(method='Mahalanobis'))
}

alpha <- 0.01
right.quantile.maha <- apply(maha_median_boot,2,quantile, probs=1 - alpha/2)
```

```r
left.quantile.maha <- apply(maha_median_boot,2,quantile, probs=alpha/2)

CI.maha <-
  matrix(c(
    maha_median - (right.quantile.maha - maha_median),
    maha_median,
    maha_median - (left.quantile.maha- maha_median)),byrow = TRUE,nrow = 3,ncol = p)
rownames(CI.maha)=c('lwr','pointwise','upr')
colnames(CI.maha)=names(maha_median)
CI.maha
```

```
##           alcohol proline calcium
## lwr         12.62     700      70
## pointwise   12.93     770      79
## upr         13.14     855      96
```

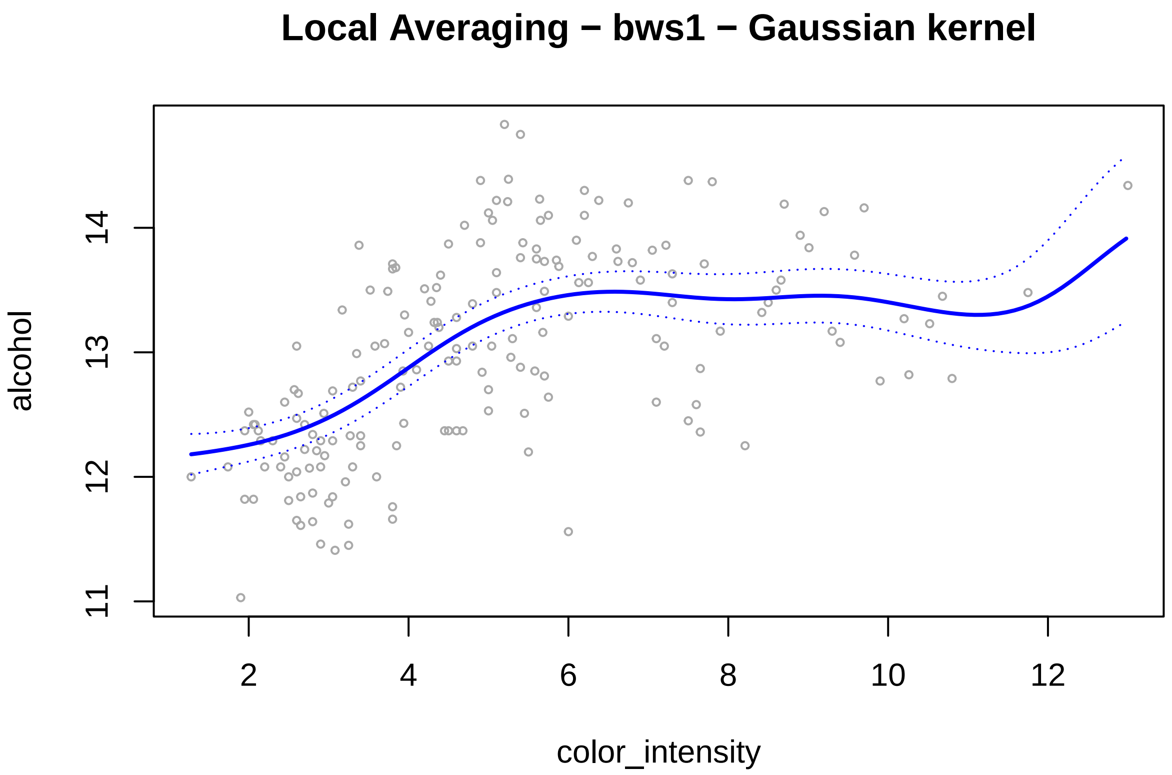
Exercise 2

Simon Le Vanten is now interested in building a nonparametric model to predict the Alcohol by Volume of his wines as a function of color intensity (measured in absorbance units) and concentration of potassium (milligrams per liter), contained in the `df_2.Rds` file. He therefore asks you to:

1. Fit a local linear regression model with a Gaussian kernel and bandwidth equal to 1 absorbance unit to predict the Alcohol by Volume as a function of color intensity. Provide a plot of the regression curve.

```r
df_2 <- readRDS(here("2023-06-19/data/df_2.Rds"))
# Local regression Gaussian kernel
N <- nrow(df_2)
m_loc = npreg(alcohol ~ color_intensity,
              ckertype = 'gaussian',
              bws = 1,
              residuals=TRUE,
              data = df_2)



newdata = data.frame(color_intensity = with(df_2, seq(
  range(color_intensity)[1],
  range(color_intensity)[2],
  by = .1
)))
preds_loc=predict(m_loc,newdata=newdata,se.fit = T)
se.bands_loc=cbind(preds_loc$fit +2* preds_loc$se.fit ,preds_loc$fit -2* preds_loc$se.fit)
with(
  df_2,
  plot(
    color_intensity ,
    alcohol ,
    xlim = range(newdata$color_intensity) ,
    cex = .5,
    col = " darkgrey ",
    main = 'Local Averaging - bws1 - Gaussian kernel'
  )
)

lines(newdata$color_intensity,preds_loc$fit ,lwd =2, col =" blue")
```

```r
matlines(newdata$color_intensity,se.bands_loc ,lwd =1, col =" blue",lty =3)
```



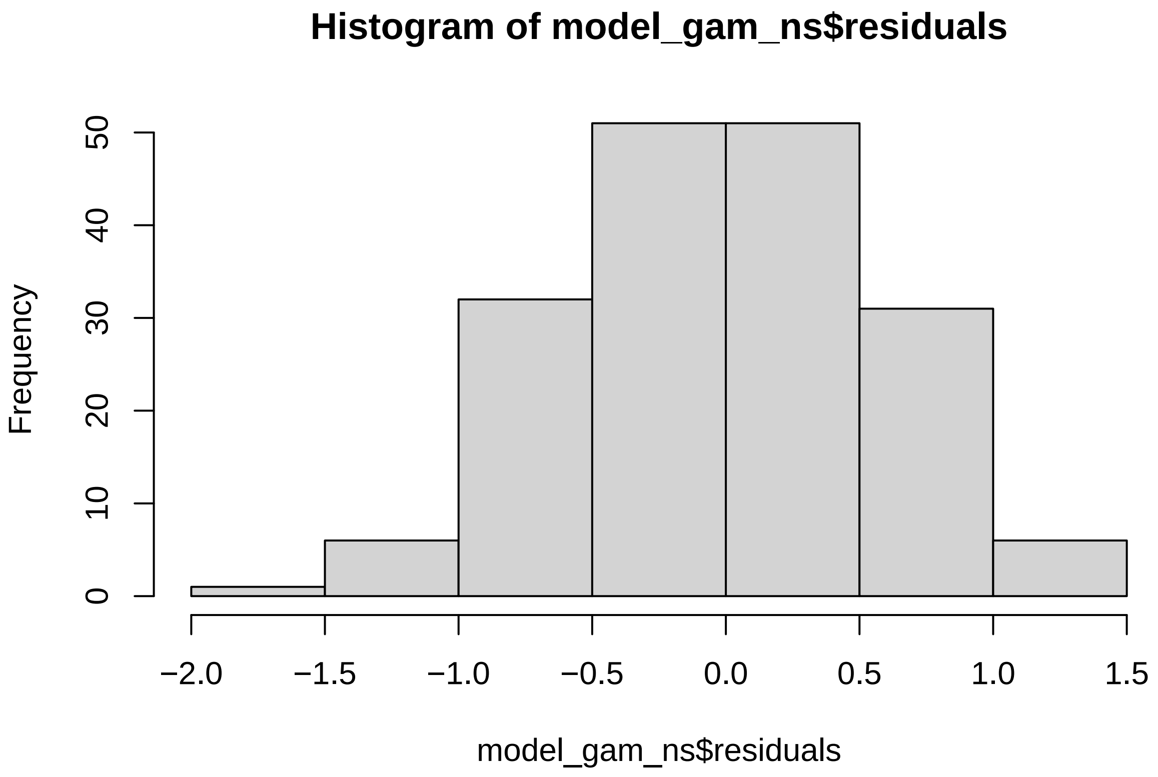
2. Build an additive model for regressing Alcohol by Volume on color intensity and concentration of potassium using natural cubic splines with one knot at the median and boundary knots at the 5th and 95th percentiles as univariate smoothers for the two predictors. Report the summary table including a comment on statistical significance. Provide an histogram of the residuals of the model.

```r
knots_col_intensity <- median(df_2$color_intensity)
knots_potassium <- median(df_2$potassium)

boundary_knots_col_intensity <- quantile(df_2$color_intensity,probs = c(.05,.95))
boundary_knots_potassium <- quantile(df_2$potassium,probs = c(.05,.95))

model_gam_ns <-
  lm(
    alcohol ~ ns(color_intensity, knots = knots_col_intensity,
                 Boundary.knots = boundary_knots_col_intensity) +
      ns(potassium, knots = knots_potassium, Boundary.knots = boundary_knots_potassium),
    data = df_2
  )

# summary(model_gam_ns)
knitr::kable(broom::tidy(summary(model_gam_ns)),digits = 4)
```

term	estimate	std.error	statistic	p.value
(Intercept)	11.9187	0.1340	88.9441	0.0000
ns(color_intensity, knots = knots_col_intensity, Boundary.knots = boundary_knots_col_intensity)1	2.8167	0.2408	11.6974	0.0000
ns(color_intensity, knots = knots_col_intensity, Boundary.knots = boundary_knots_col_intensity)2	0.7953	0.1332	5.9712	0.0000

term	estimate	std.error	statistic	p.value
ns(potassium, knots = knots_potassium, Boundary.knots = boundary_knots_potassium)1	0.1109	0.2427	0.4568	0.6484
ns(potassium, knots = knots_potassium, Boundary.knots = boundary_knots_potassium)2	-0.2455	0.1453	-1.6893	0.0930

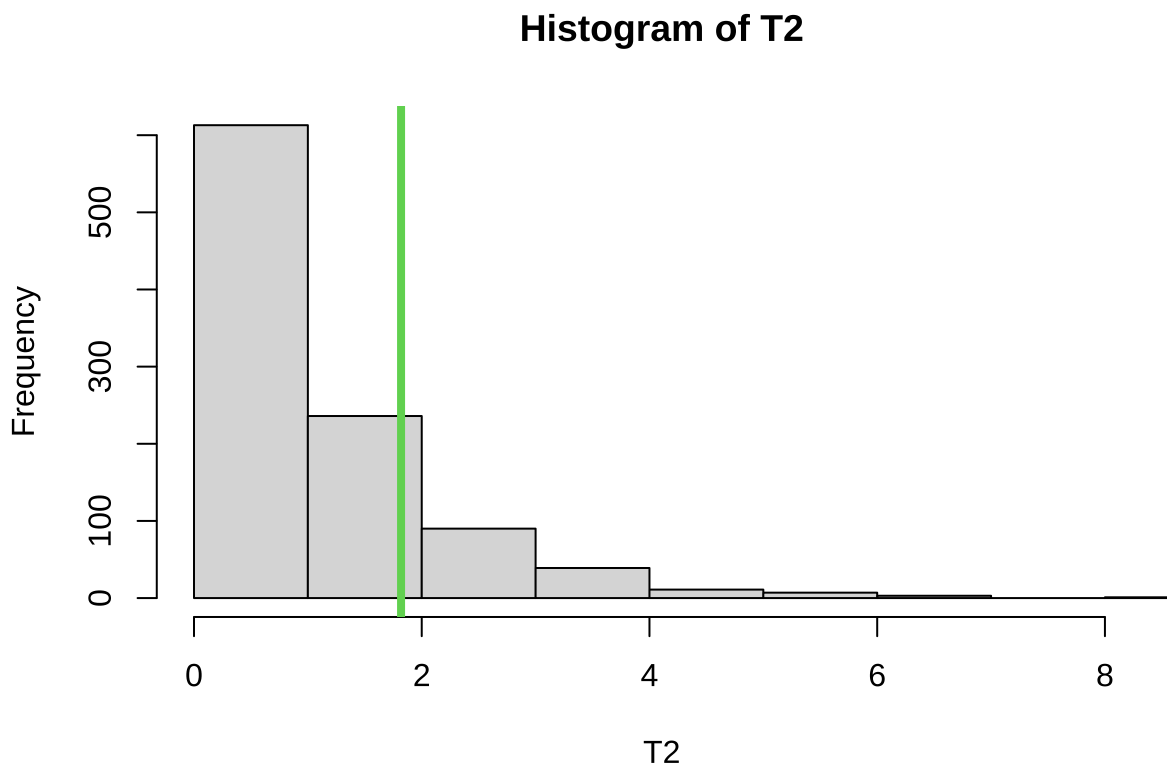
```
hist(model_gam_ns$residuals)
```



3. Build a reduced version of the previous model considering only the contribution of color intensity for explaining Alcohol by Volume. Employ a permutational test to validate which model should be preferred and report the resulting p-value. Consider the F value as test statistic and use the Friedman & Lane permutation scheme.

```
model_gam_ns_reduced <-
  lm(
    alcohol ~ ns(color_intensity, knots = knots_col_intensity,
                 Boundary.knots = boundary_knots_col_intensity),
    data = df_2
  )

fitted.obs <- model_gam_ns_reduced$fitted.values
res.obs <- model_gam_ns_reduced$residuals

T_0 <- anova(model_gam_ns_reduced,model_gam_ns)[2,5]

# Estimating the permutational distribution under H0
B <- 1000
T2 <- numeric(B)
```

```r
set.seed(2022)

for (perm in 1:B) {
  res_reduced_perm <- res.obs[sample(1:N)]
  y_perm <- fitted.obs + res_reduced_perm

  df_2_perm <- df_2
  df_2_perm$alcohol <- y_perm
  model_gam_ns_perm <-
    lm(
      alcohol ~ ns(color_intensity, knots = knots_col_intensity,
                  Boundary.knots = boundary_knots_col_intensity) +
        ns(potassium, knots = knots_potassium,
          Boundary.knots = boundary_knots_potassium),
      data = df_2_perm
    )
  model_gam_ns_reduced_perm <-
    lm(
      alcohol ~ ns(color_intensity, knots = knots_col_intensity,
                  Boundary.knots = boundary_knots_col_intensity),
      data = df_2_perm
    )

  T2[perm] <- anova(model_gam_ns_reduced_perm,model_gam_ns_perm)[2,5]
}

hist(T2, xlim = range(c(T2, T_0)))
abline(v = T_0, col = 3, lwd = 4)
```



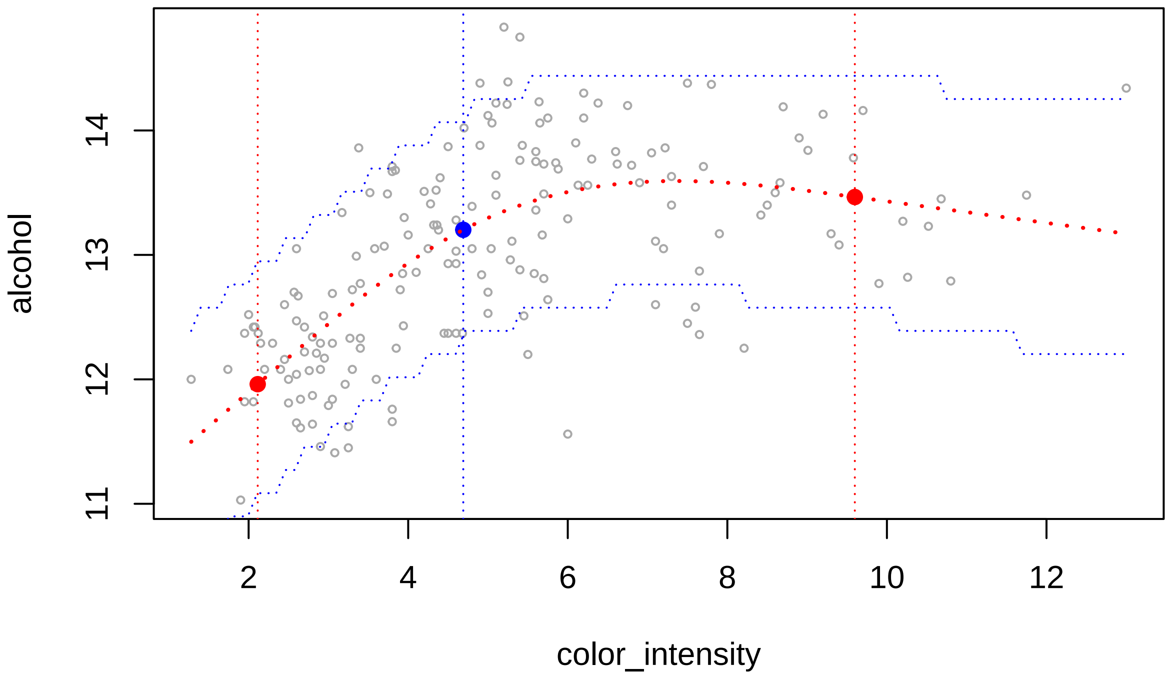
```r
p_val <- sum(T2 >= T_0) / B
p_val
```

```
## [1] 0.169
```

```r
# Cannot reject H0: reduced model is better
```

4. Compute the prediction bands for the regression model selected according to the test performed in the previous exercise, using a full conformal approach and setting $\alpha = 0.1$ as the miscoverage level

```r
color_intensity_grid = seq(range(df_2$color_intensity)[1],
                    range(df_2$color_intensity)[2],
                    length.out = 100)

preds = predict(model_gam_ns_reduced,
                list(color_intensity = color_intensity_grid),
                se = T)

with(
  df_2,
  plot(
    color_intensity ,
    alcohol ,
    xlim = range(color_intensity_grid) ,
    cex = .5,
    col = " darkgrey "
  )
)

knots_pred = predict(model_gam_ns_reduced,
                     list(color_intensity = knots_col_intensity))
points(knots_col_intensity,
       knots_pred,
       col = 'blue',
       pch = 19)
boundary_pred <-
  predict(model_gam_ns_reduced,
          list(color_intensity = boundary_knots_col_intensity))
points(boundary_knots_col_intensity,
       boundary_pred,
       col = 'red',
       pch = 19)
abline(v = knots_col_intensity, lty = 3, col = "blue")
abline(v = boundary_knots_col_intensity, lty = 3, col = "red")

lm_train = lm.funs(intercept = T)$train.fun
lm_predict = lm.funs(intercept = T)$predict.fun

design_matrix = ns(df_2$color_intensity, knots = knots_col_intensity,
                   Boundary.knots = boundary_knots_col_intensity)
pred_grid = matrix(
  ns(
    color_intensity_grid,
    knots = knots_col_intensity,
    Boundary.knots = boundary_knots_col_intensity
  ),
  nrow = length(color_intensity_grid)
)

c_preds = conformal.pred(
  x = design_matrix,
  y = df_2$alcohol,
  pred_grid,
  alpha = 0.1,
  verbose = F,
  train.fun = lm_train,
  predict.fun = lm_predict,
  num.grid.pts = 200
)

lines(
  color_intensity_grid,
  c_preds$pred ,
  lwd = 2,
  col = "red",
  lty = 3
)
matlines(
  color_intensity_grid ,
  cbind(c_preds$up, c_preds$lo) ,
  lwd = 1,
  col = " blue",
  lty = 3
)
```



Exercise 3

Simon Le Vanten knows that a good balance of sweetness and acidity is essential for the right pH to be achieved and ultimately make good wines. Since to improve yourself you need to learn from the best, he collects $N = 107$ samples of two Italian wine types (Barolo and Barbera) for which he measures glycerol, sugar free extract, tartaric acid and malic acid levels (all expressed in grams per liter) along with the resulting pH. The dataset containing this information is stored in the `df_3.rds` file.

1. Focusing on the Barolo wine samples, compute the Minimum Covariance Determinant estimator for the `glycerol`, `sugar_free_extract`, `tartaric_acid` and `malic_acid` variables. Consider 1000 subsets for initializing the algorithm and set the sample size of H, the subset over which the determinant is minimized, equal to 40. Report the raw MCD estimates of location and scatter. Define a vector `ind_out_MCD` of row indexes identifying the samples (within the Barolo subpopulation) that are outliers according to the MCD call and report it.

```r
df_3 <- readRDS(here("2023-06-19/data/df_3.rds"))
X_barolo <- df_3[df_3$type=="Barolo",3:6]

N <- nrow(X_barolo)
set.seed(2022)

fit_MCD <-
  covMcd(
    x = X_barolo,
    alpha = (N - 19) / N,
    nsamp = 1000
  )

fit_MCD$raw.center
```

```
##           glycerol sugar_free_extract      tartaric_acid         malic_acid
##           9.352195          26.281463           1.612927           1.766585
```

```r
fit_MCD$raw.cov
```

```
##                       glycerol sugar_free_extract tartaric_acid   malic_acid
## glycerol            1.28618262         0.36667822  -0.085221790 -0.011855114
## sugar_free_extract  0.36667822         2.66513365   0.173053657  0.049014872
## tartaric_acid      -0.08522179         0.17305366   0.178480600  0.001040072
## malic_acid         -0.01185511         0.04901487   0.001040072  0.051255521
```

```r
ind_out_MCD <- setdiff(1:N,fit_MCD$best)
ind_out_MCD
```

```
##  [1]  3  5 10 12 13 14 20 22 28 33 37 40 42 44 46 47 50 57
```

2. Simon Le Vanten knows that Italians may be a little bit sneaky when it comes to wines, and he believes some Barolo wines may have been purposely labeled as Barbera. Employing the (reweighted) MCD estimates obtained in the previous exercise, check if some Barbera samples have been wrongly labeled by computing robust squared Mahalanobis distances using $\chi^2_{4,0.975}$ as cut-off value, with $\chi^2_{p,\alpha}$ denoting the α-quantile of a χ^2 distribution with p degrees of freedom

```r
X_barbera <- df_3[df_3$type == "Barbera", 3:6]
ind_wrongly_labeled_obs <-
  which(
    mahalanobis(
      x = X_barbera,
      center = fit_MCD$center,
      cov = fit_MCD$cov
    ) <= qchisq(p = .975, df = 4)
  )
length(ind_wrongly_labeled_obs)
```

```
## [1] 3
```

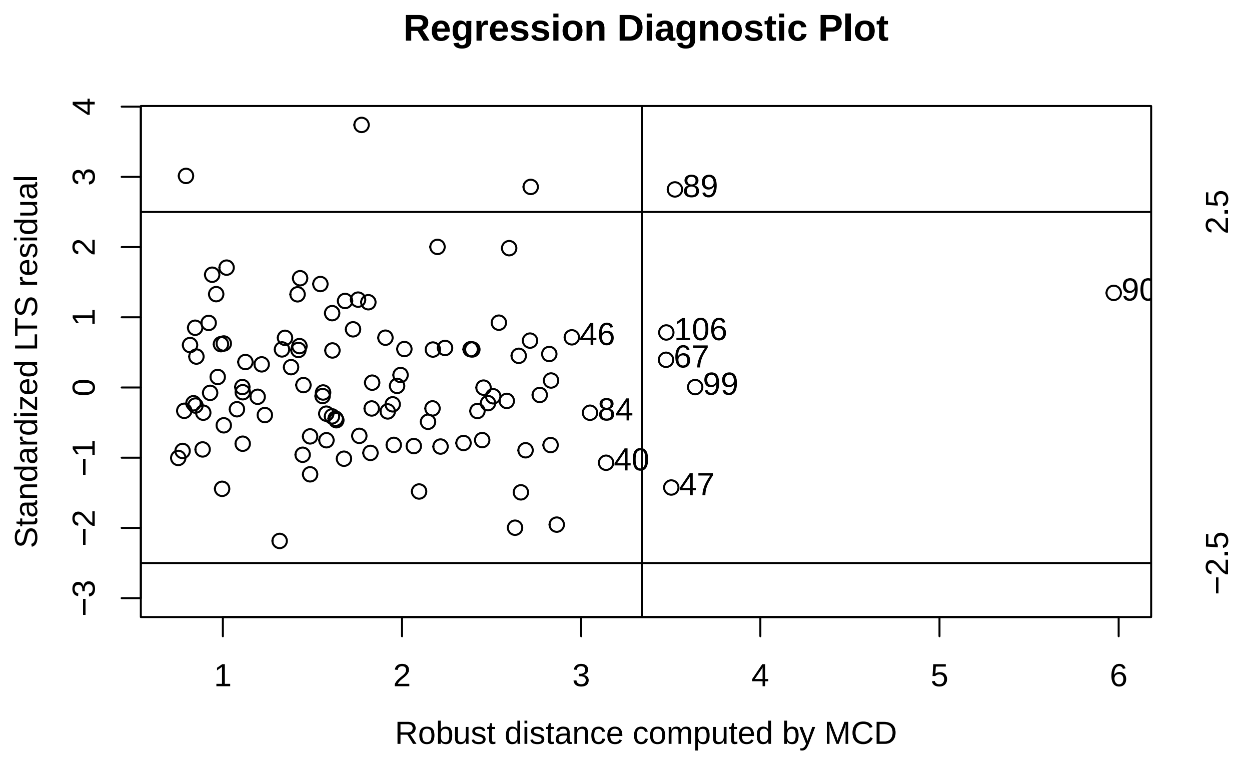
```
# 3 obs may have been wrongly labeled!
```

3. Since the type variable could be unreliable, Simon Le Vanten asks you to build a robust linear model for the entire dataset to regress `p_h` on the remaining variables (excluding `type`) using a Least Trimmed Squares (LTS) approach, setting the hyperparameter $\alpha = 0.75$. Report the summary table including a comment on statistical significance. Provide the outlier map for the robust linear model: are bad leverage points present in the dataset according to the diagnostic plot?

```
fit_lts <-
  ltsReg(p_h ~ .-type, alpha = 0.75, data = df_3)
summary(fit_lts)
```

```
##
## Call:
## ltsReg.formula(formula = p_h ~ . - type, data = df_3, alpha = 0.75)
##
## Residuals (from reweighted LS):
##        Min         1Q     Median         3Q        Max
## -0.2593207 -0.0609438 -0.0002128  0.0659201  0.2376421
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## Intercept             3.887760   0.152659  25.467  < 2e-16 ***
## glycerol             -0.004311   0.007472  -0.577   0.5653
## sugar_free_extract -0.009655   0.005768  -1.674   0.0974 .
## tartaric_acid        -0.126530   0.017410  -7.268 8.94e-11 ***
## malic_acid           -0.017981   0.010765  -1.670   0.0981 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1062 on 98 degrees of freedom
## Multiple R-Squared: 0.4511,  Adjusted R-squared: 0.4287
## F-statistic: 20.13 on 4 and 98 DF,  p-value: 3.977e-12
```

```
plot(fit_lts, which="rdiag")
```



```
# One single leverage point: unit 89
```