Exam

Nonparametric Statistics, AY 2021/22

June 20, 2022

Algorithmic Instructions

- All the numerical values required need to be put on an A4 sheet and uploaded, alongside the required plots.
- 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.
- 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.
- Data for the exam can be found at this link

Exercise 1

Mr András Kaponzyi, an Hungarian data scientist, is about to become a father of his first daughter. During the gynecological examination of the second quarter, the fetus weighted 353 grams. Mr András Kaponzyi is worried that the newborn will be very chubby, he is therefore interested in estimating the expected birth weight of his daughter. To do so, he has collected 240 measurements of fetus'weights (in grams) at 20 weeks pregnancy (`baby_weight_20_weeks`), and the respective weight in grams at birth (`baby_weight_birth`). The resulting samples are contained in the `df_1.Rds` file. Assume that:

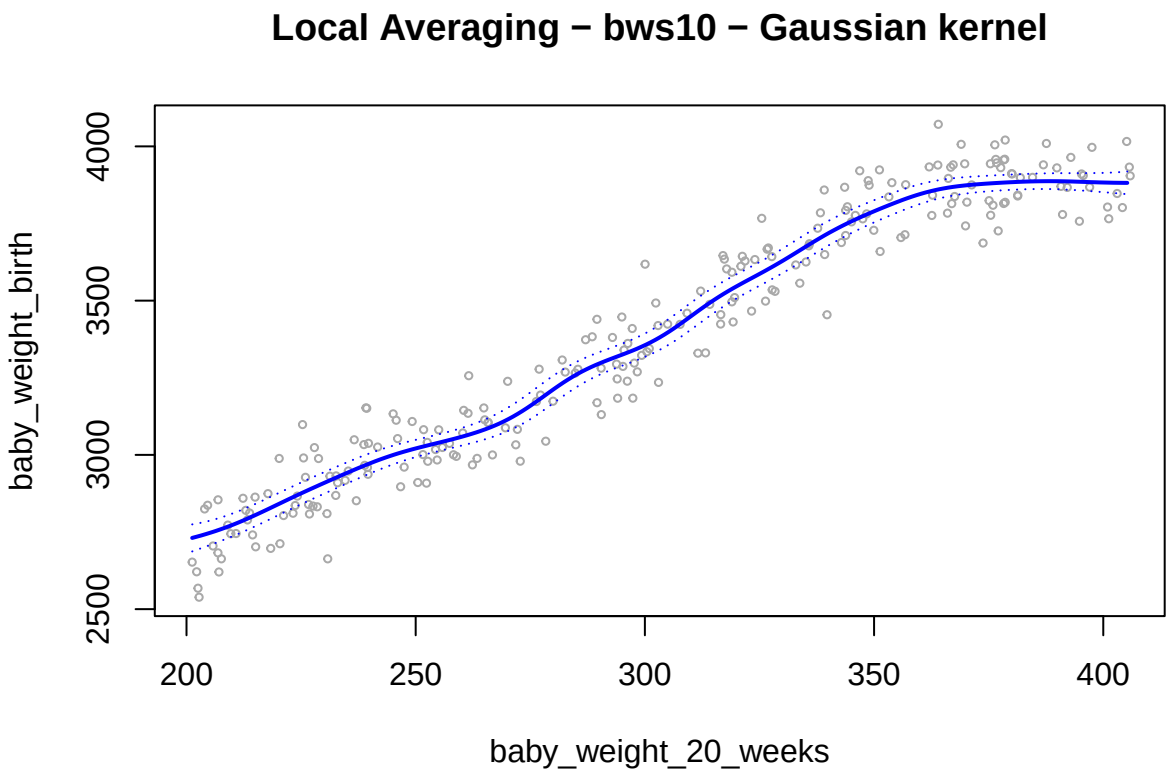
$$baby\_weight\_birth = f(baby\_weight\_20\_weeks) + \varepsilon$$

.

1. Fit a local regression model with a Gaussian kernel and bandwidth equal to 10 grams to predict the birth weight as a function of the weight at 20 weeks pregnancy. Provide a plot of the regression curve and compute the point-wise prediction (round it at the second decimal digit) for the birth weight of Kaponzyi's daughter. Using a bootstrap approach, provide a reverse percentile confidence interval ($\alpha = 0.05$) for the expected weight of Kaponzyi's daughter.

```
# Local regression Gaussian kernel
N <- nrow(df_1)
m_loc = npreg(baby_weight_birth ~ baby_weight_20_weeks,
              ckertype = 'gaussian',
              bws = 10,
              residuals=TRUE,
              data = df_1)


newdata = data.frame(baby_weight_20_weeks = with(df_1, seq(
  range(baby_weight_20_weeks)[1],
  range(baby_weight_20_weeks)[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_1,
  plot(
    baby_weight_20_weeks ,
    baby_weight_birth ,
    xlim = range(newdata$baby_weight_20_weeks) ,
    cex = .5,
    col = " darkgrey ",
    main = 'Local Averaging - bws10 - Gaussian kernel'
  )
)

lines(newdata$baby_weight_20_weeks,preds_loc$fit ,lwd =2, col =" blue")
matlines(newdata$baby_weight_20_weeks,se.bands_loc ,lwd =1, col =" blue",lty =3)
```




```
# Point-wise prediction

pred_kaponzyi <- predict(m_loc,newdata=list(baby_weight_20_weeks=353))
round(pred_kaponzyi,2)
```

```
## [1] 3807.94
```

```
# Reverse percentile confidence interval
B = 1000
res.obs <- m_loc$resid
fitted.obs <- m_loc$mean
```

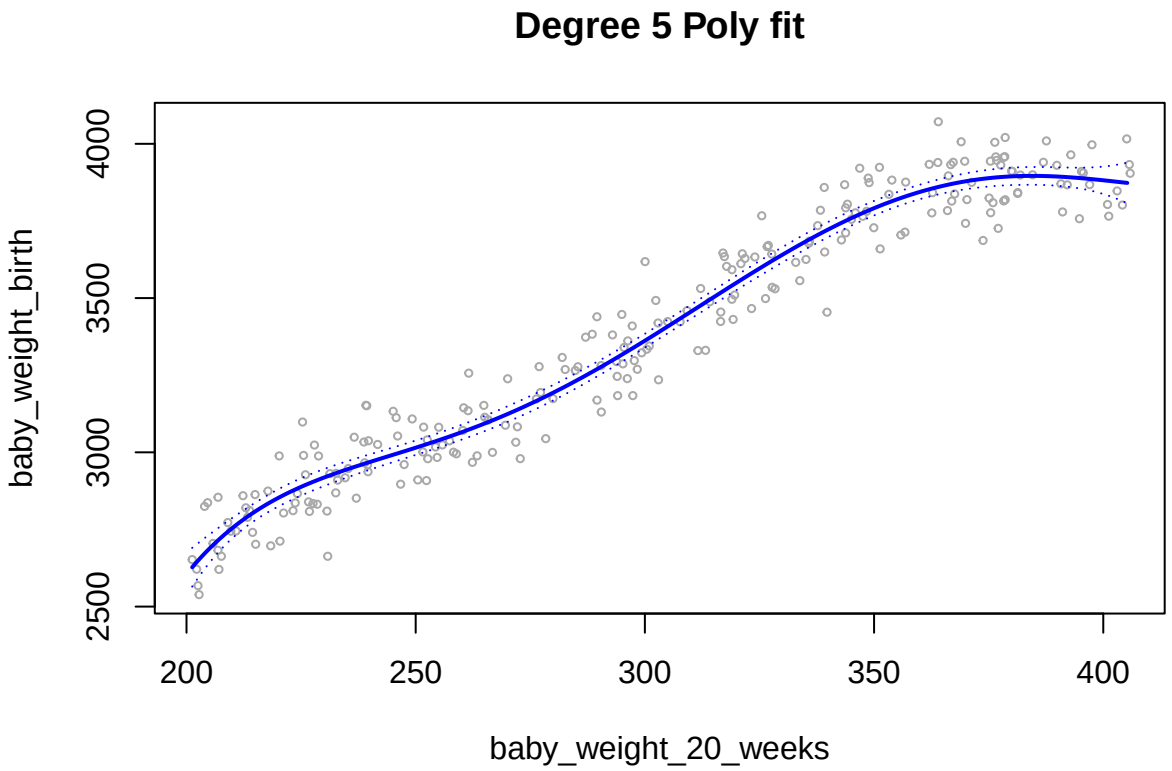
```
boot_pred_kaponzyi = numeric(B)
set.seed(2022)
for (b in 1:B) {
  baby_weight_birth_boot <- fitted.obs + sample(res.obs, replace = T)
  df_boot <-
    data.frame(baby_weight_birth_boot,
               baby_weight_20_weeks = df_1$baby_weight_20_weeks)
  fit_kernel_boot <-
    npreg(
      baby_weight_birth_boot ~ baby_weight_20_weeks,
      ckertype = 'gaussian',
      bws = 10,
      data = df_boot
    )
  boot_pred_kaponzyi[b] = predict(fit_kernel_boot, newdata = list(baby_weight_20_weeks =
                                                                    353))
}
alpha <- 0.05
right.quantile.pred_kaponzyi <- quantile(boot_pred_kaponzyi, 1 - alpha / 2)
leftquantile.pred_kaponzyi <- quantile(boot_pred_kaponzyi, alpha / 2)
CI.RP.pred_kaponzyi <-
  c(
    pred_kaponzyi - (right.quantile.pred_kaponzyi - pred_kaponzyi),
    pred_kaponzyi - (leftquantile.pred_kaponzyi - pred_kaponzyi)
  )
names(CI.RP.pred_kaponzyi) = c('lwr', 'upr')
CI.RP.pred_kaponzyi
```

```
##      lwr      upr
## 3791.794 3846.545
```

1. Fit a polynomial regression model of degree 5 to predict the birth weight as a function of the weight at 20 weeks pregnancy. Provide a plot of the regression line and a table with the estimated model coefficients. By using a permutational approach, use the Anova F test statistic to check whether a polynomial regression model of degree 6 should be preferred. Specify the null and the alternative hypothesis you are testing and report the resulting p-value. Comment on the result.

```
fit_poly5 <-
  lm(baby_weight_birth ~ poly(baby_weight_20_weeks, degree = 5, raw = F),
     data = df_1)
preds_poly5 = predict(fit_poly5,
                      list(baby_weight_20_weeks = newdata$baby_weight_20_weeks),
                      se = T)
se.bands_poly5 = cbind(preds_poly5$fit + 2 * preds_poly5$se.fit ,
                       preds_poly5$fit - 2 * preds_poly5$se.fit)
with(
  df_1,
  plot(
    baby_weight_20_weeks ,
    baby_weight_birth ,
    xlim = range(newdata$baby_weight_20_weeks) ,
    cex = .5,
    col = " darkgrey ",
    main = 'Degree 5 Poly fit'
```

```
  )
)
lines(newdata$baby_weight_20_weeks,
      preds_poly5$fit ,
      lwd = 2,
      col = " blue")
matlines (
  newdata$baby_weight_20_weeks ,
  se.bands_poly5 ,
  lwd = 1,
  col = " blue",
  lty = 3
)
```



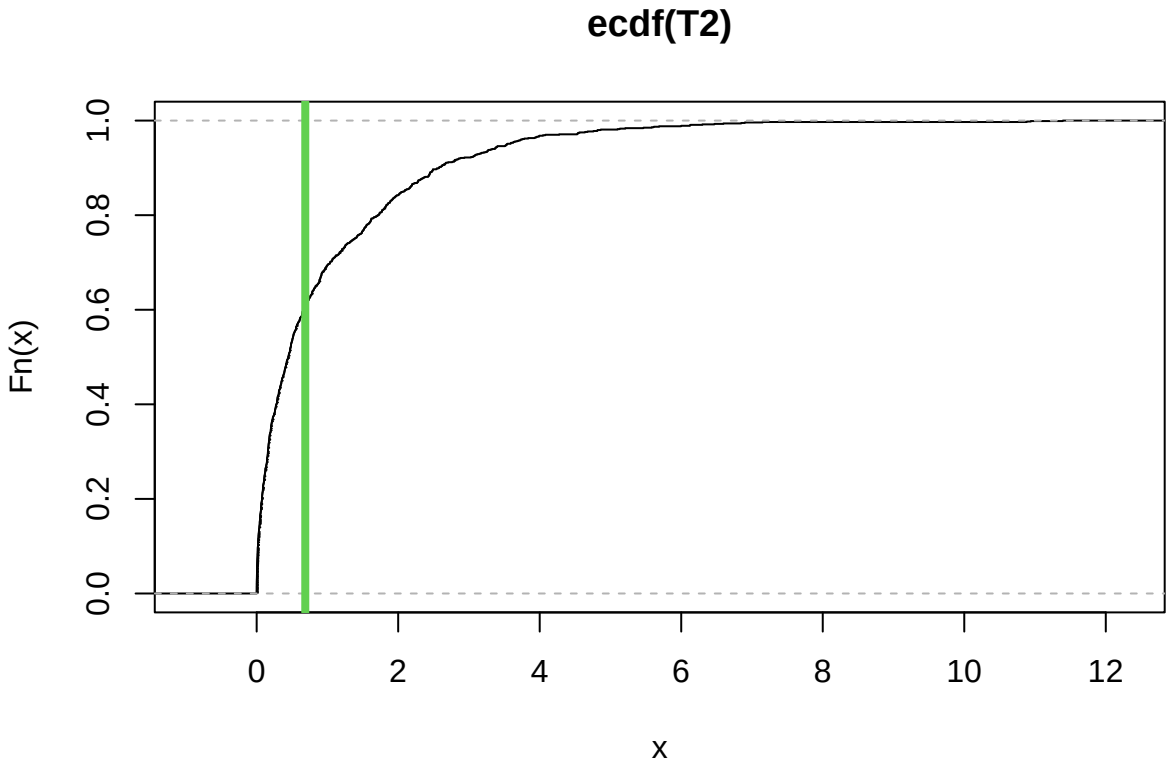

```
knitr::kable(broom::tidy(summary(fit_poly5)))
```

term	estimate	std.error	statistic	p.value
(Intercept)	3378.7336	5.849855	577.575576	0.0000000
poly(baby_weight_20_weeks, degree = 5, raw = F)1	6252.6423	90.625570	68.994239	0.0000000
poly(baby_weight_20_weeks, degree = 5, raw = F)2	-418.9035	90.625570	-4.622354	0.0000063
poly(baby_weight_20_weeks, degree = 5, raw = F)3	-615.1968	90.625570	-6.788336	0.0000000
poly(baby_weight_20_weeks, degree = 5, raw = F)4	-424.9162	90.625570	-4.688700	0.0000047
poly(baby_weight_20_weeks, degree = 5, raw = F)5	264.2494	90.625570	2.915837	0.0038920

```
# Permutational anova

fit_poly5_reduced <- fit_poly5
```

```r
fit_poly6_full <-
  lm(baby_weight_birth ~ poly(baby_weight_20_weeks, degree = 6, raw = F),
     data = df_1)

fitted.obs <- fit_poly5_reduced$fitted.values
res.obs <- fit_poly5_reduced$residuals

T_0 <- anova(fit_poly5_reduced,fit_poly6_full)[2,5]

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

set.seed(2022)

for (perm in 1:B) {
  res_reduced_perm <- res.obs[sample(1:N)]
  y_perm <- fitted.obs + res_reduced_perm
  # New dataset with permuted response
  df_1_perm <- df_1
  df_1_perm$baby_weight_birth <- y_perm
  fit_poly6_full_perm <-
    lm(baby_weight_birth ~ poly(baby_weight_20_weeks, degree = 6, raw = F),
       data = df_1_perm)
  fit_poly5_reduced_perm <-
    lm(baby_weight_birth ~ poly(baby_weight_20_weeks, degree = 5, raw = F),
       data = df_1_perm)
  T2[perm] <- anova(fit_poly5_reduced_perm, fit_poly6_full_perm)[2, 5]
}

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


```
plot(ecdf(T2))
abline(v = T_0, col = 3, lwd = 4)
```




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

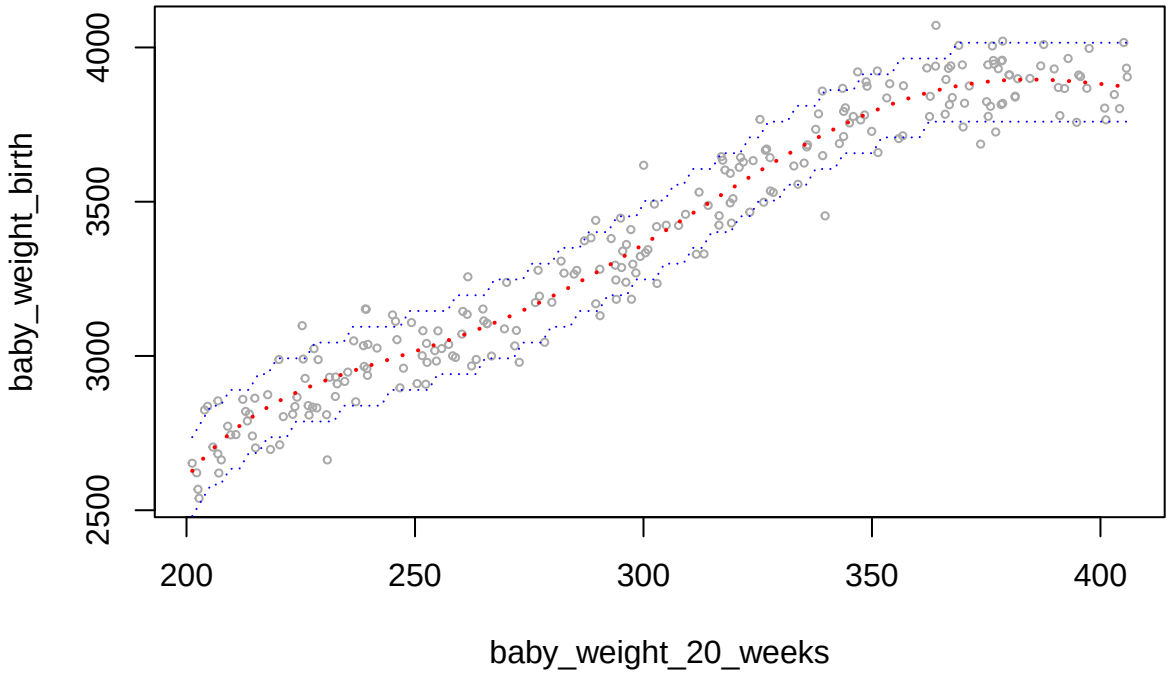
```
## [1] 0.395
```

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

3. 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.

```
baby_weight_20_weeks_grid = seq(
  range(df_1$baby_weight_20_weeks)[1],
  range(df_1$baby_weight_20_weeks)[2],
  length.out = 100
)

with(
  df_1,
  plot(
    baby_weight_20_weeks ,
    baby_weight_birth ,
    xlim = range(baby_weight_20_weeks_grid) ,
    cex = .5,
    col = " darkgrey "
  )
)

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

design_matrix = matrix(poly(df_1$baby_weight_20_weeks, degree = 5), ncol =
                         5)

pred_grid = matrix(poly(
  baby_weight_20_weeks_grid,
  degree = 5,
  coefs = attr(poly(df_1$baby_weight_20_weeks, degree = 5), "coefs")
), ncol = 5)

c_preds = conformal.pred(
  design_matrix,
  df_1$baby_weight_birth,
  pred_grid,
  alpha = 0.1,
  verbose = F,
  train.fun = lm_train,
  predict.fun = lm_predict,
  num.grid.pts = 200
)

lines(
  x = baby_weight_20_weeks_grid,
  y = c_preds$pred,
  lwd = 2,
  col = "red",
  lty = 3
)
matlines(
```

```
  baby_weight_20_weeks_grid ,
  cbind(c_preds$up, c_preds$lo) ,
  lwd = 1,
  col = " blue",
  lty = 3
)
```



Exercise 2

Since becoming a parent is not stressful enough, Mr András Kaponzyi is also considering buying a house in Budapest. In order to do so, he has collected information on 505 properties on sale in the capital, recording the Price (in mln Hungarian forint), the Dimension in m^2 and the Area in which the property is located (either Buda or Pest). The resulting samples are contained in the `df_2.Rds` file. He now requires you to:

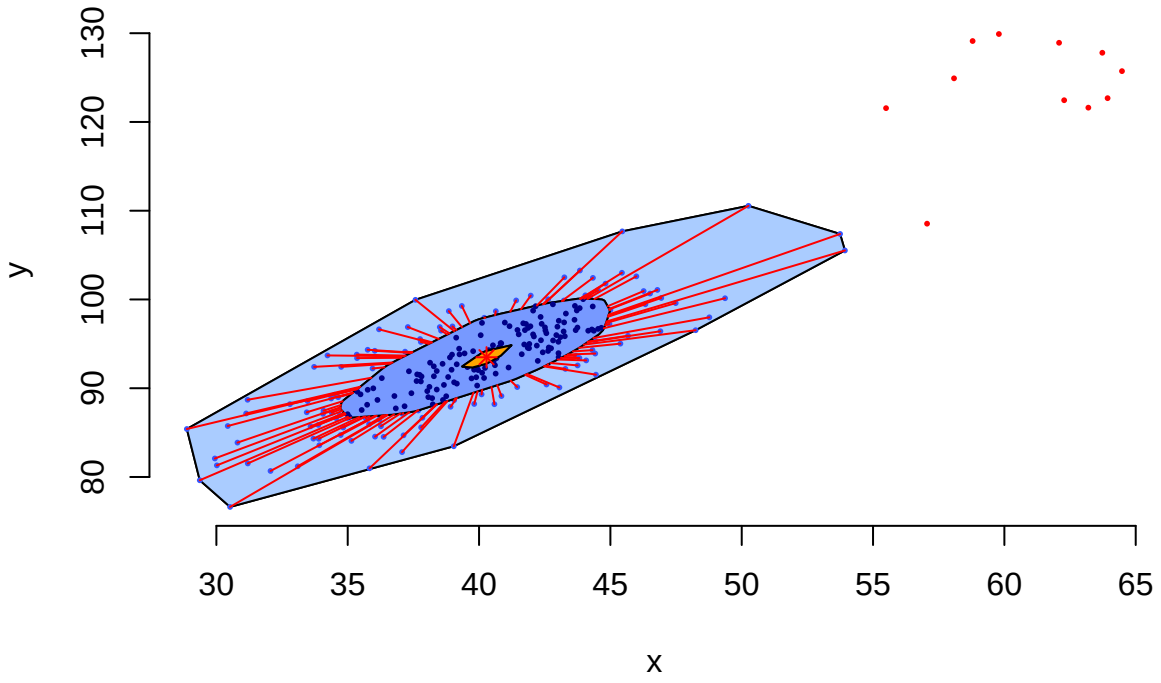
1. Provide the Tukey median of the properties located in Pest.

```
N <- nrow(df_2)
p <- ncol(df_2)
X_pest <- df_2[df_2$Area=="Pest",1:2]
(tukey_median_pest <- depthMedian(as.matrix(X_pest),depth_params = list(method="Tukey")))
```

```
##     Price Dimension
##  36.59497  95.79681
```

2. The real estate agent helping Mr András Kaponzyi with the house hunting told him that the Pest area is outliers-free, while some peculiar properties may be found in the Buda area. Therefore, for the properties in the Buda area, plot a bagplot of the two collected variables and determine a vector of row indexes identifying the samples that are outliers according to such procedure. What characteristics do the outlying properties showcase?

```
X_buda <- df_2[df_2$Area=="Buda",1:2]
bagplot_buda <- bagplot(X_buda)
```




```
ind_out <-
  which(apply(X_buda[, 1:2], 1, function(x)
    all(x %in% bagplot_buda$pxy.outlier)))
unname(ind_out)
```

```
##  [1]  32  64  66  81  82  93  94 182 221 228 229
```

```
# Big and expensive properties
```

3. After having eliminated the properties identified as outliers in the Buda area[1], use a permutation test to check whether the variance covariance matrix of Price and Dimension differ in the two areas (Buda and Pest). To do so, consider the following procedure: firstly center the Buda (no outliers) and Pest observations by subtracting their respective sample means[2]. Then, to perform the test, use as test statistic the squared Frobenius distance[3] between the sample covariance matrices of the centered data. Specify assumptions, the null and the alternative hypothesis you are testing, provide the histogram of the permuted distribution of the test statistic, its cumulative distribution function and the p-value, commenting the results.

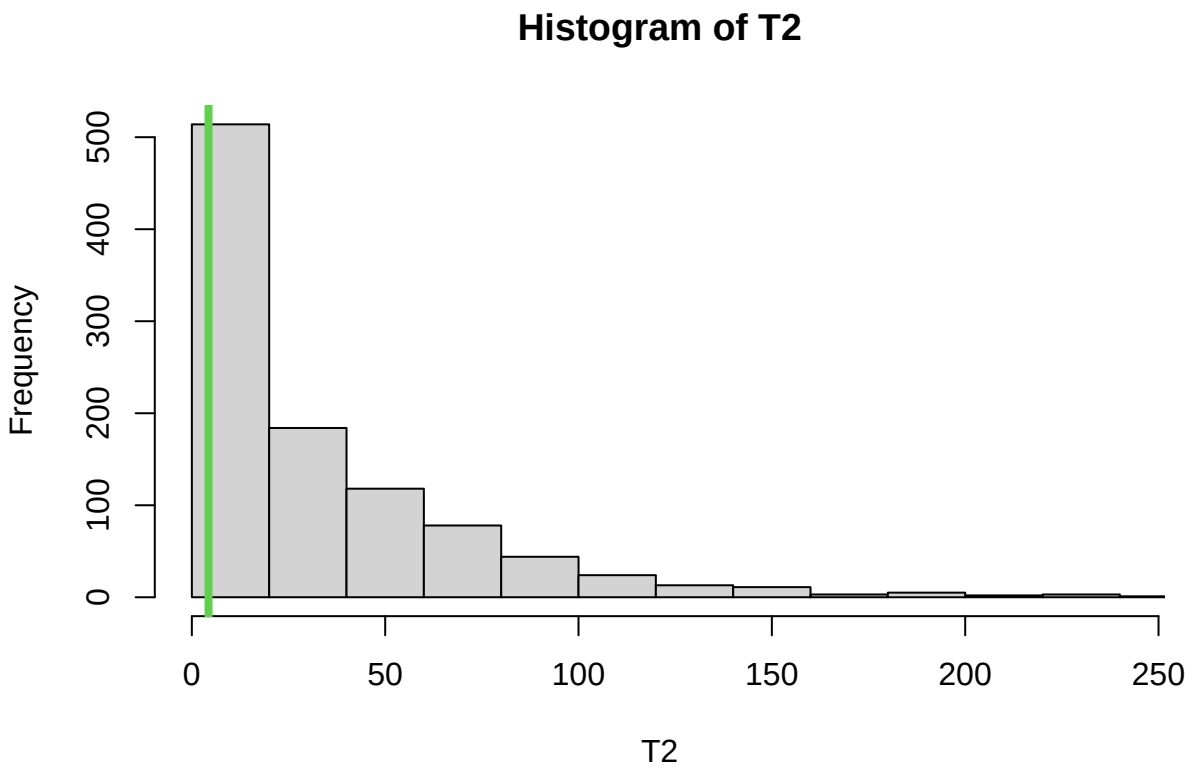
```
X_buda_no_out <- X_buda[-ind_out,]
X_buda_no_out_centered <-
  scale(X_buda_no_out, center = TRUE, scale = FALSE)

X_pest_centered <-
  scale(X_pest, center = TRUE, scale = FALSE)
S_buda <- cov(X_buda_no_out_centered)
S_pest <- cov(X_pest_centered)

# permutation test

T_20 <- norm(S_buda-S_pest,"F")^2
```

[1] if you did not manage to solve point 2., here you find the vector of row indexes identifying the outlying Buda properties (for the Buda subset): 32 64 66 81 82 93 94 182 221 228 229

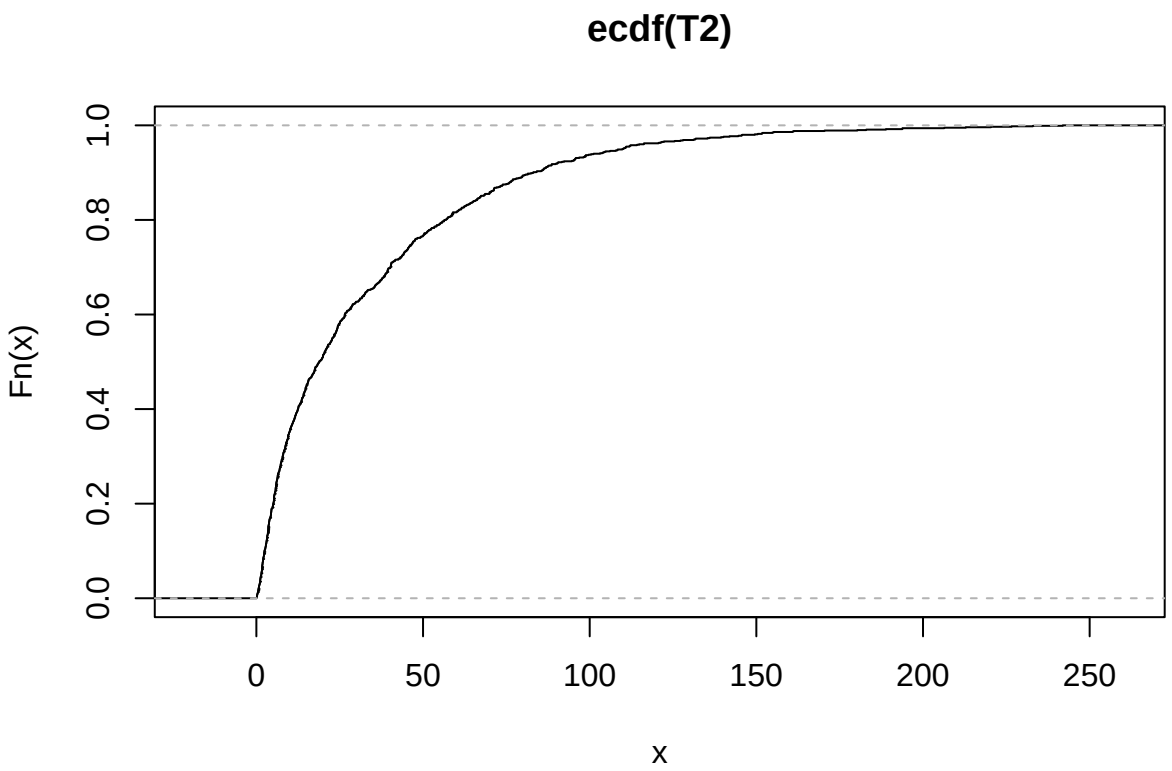
[2] Hint: use the `scale(x, center = TRUE, scale = FALSE)` function

[3] Hint: check the `norm` function

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

set.seed(2022)
X_merged <- rbind(X_pest_centered,X_buda_no_out_centered)
N_merged <- nrow(X_merged)
pb=progress::progress_bar$new(total=B, format = " Processing [:bar] :percent eta: :eta")
N_pest <- nrow(X_pest)
for(b in 1:B) {
  perm = sample(1:N_merged)
  X_perm = X_merged[perm, ]
  S_buda.p = cov(X_perm[1:N_pest, ])
  S_pest.p = cov(X_perm[(N_pest + 1):N_merged, ])
  T2[b] = norm(S_buda.p-S_pest.p,"F")^2
  pb$tick()
}

# Plotting the permutational distribution under H0
hist(T2,xlim=range(c(T2,T_20)))
abline(v=T_20,col=3,lwd=4)
```




```r
plot(ecdf(T2))
```




```
# p-value
p_val <- sum(T2>=T_20)/B
p_val
```

```
## [1] 0.826
```

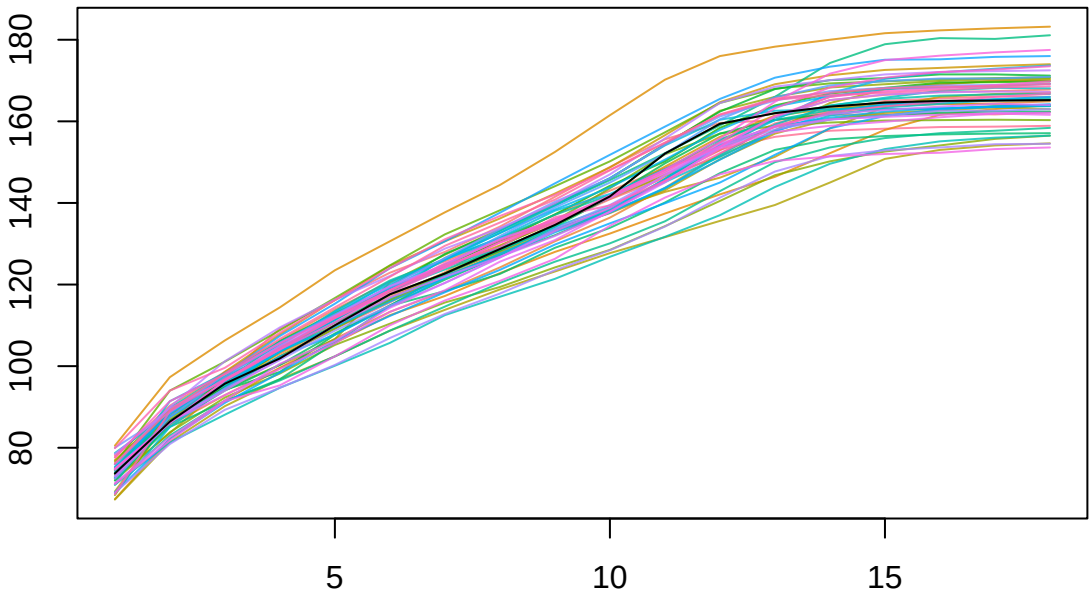
Exercise 3

Mr András Kaponzyi is now interested in deeper understanding how the height of his soon-to-be-born daughter might look like in the years to come. To do so, he asks you to analyze (a simplified version of) the famous Berkeley Growth Study dataset. The `df_3.Rds` file contains a list of two components.

- `height_values`: a 54 by 18 numeric matrix giving the heights in centimeters of 54 girls
- `age_range`: a numeric vector of length 18 giving the ages at which the heights were measured

1. By suitably defining a functional data object, plot the resulting height for the $N = 54$ girls contained in the sample. Then, compute the median age using the modified band depth and superimpose it to the previous plot.

```
grid <- df_3$age_range
f_data <- fData(grid,df_3$height_values)
plot(f_data)
N <- f_data$N
band_depth <- MBD(Data = f_data)
median_curve <- median_fData(fData = f_data, type = "MBD")
plot(f_data)
lines(grid,median_curve$values)
```



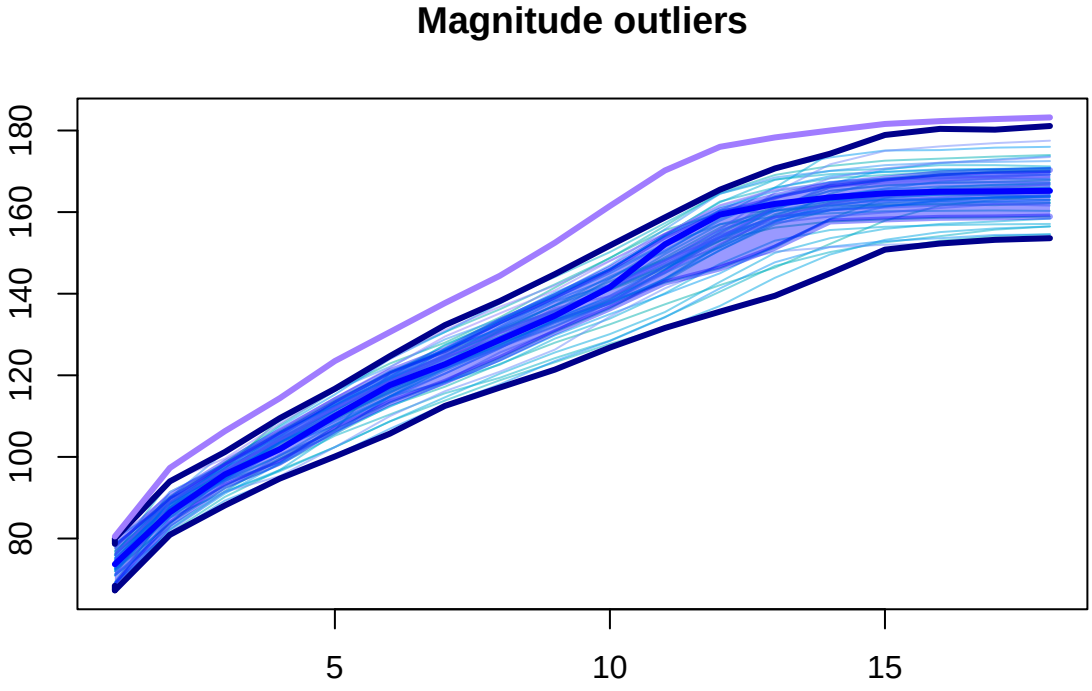
2. Produce a functional boxplot and use it to identify the outlying curves. Are there amplitude outliers? Without resorting to the `outliergram` function, manually build a scatterplot of Modified Epigraph Index (MEI) vs Modified Band Depth (MBD), and flag the curves (i.e., color them in red in the scatterplot) whose $d_i \geq Q_3(d) + 2 \cdot IQR(d)$, with

$$d_i = a_0 + a_1 MEI(f) + a_2 N^2 MEI^2(f) - MBD(f), \quad i = 1, \ldots, N$$

and

$$a_0 = a_2 = -2/(N(N-1)), \quad a_1 = 2(N+1)/(N-1)$$

```
fb_plot <- fbplot(f_data, main="Magnitude outliers")
```



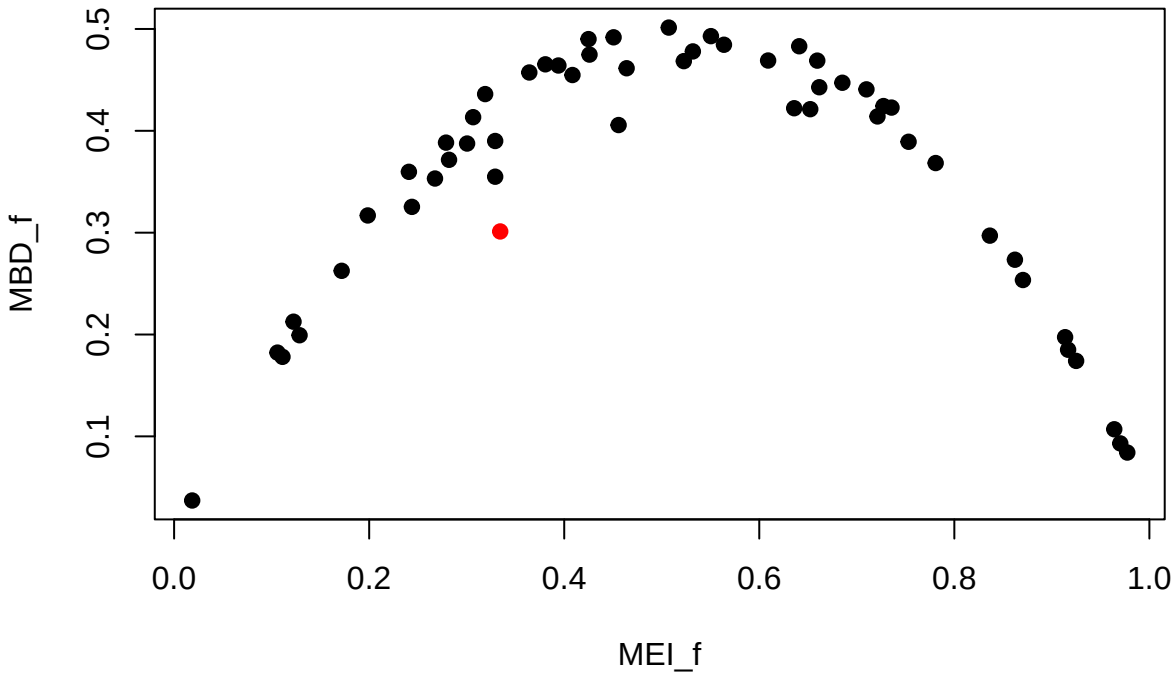

```
FDA_out <- c(fb_plot$ID_outliers)
FDA_out
```

```
## girl08
##      8
```

```
# outgr_plot <- outliergram(f_data,Fvalue = 2)
MEI_f <- MEI(f_data)
MBD_f <- MBD(f_data)
a_0 <- a_2 <- -2/(f_data$N*(f_data$N-1))
a_1 <- 2*(f_data$N+1)/(f_data$N-1)
d_manual <- a_0+a_1*MEI_f+a_2*f_data$N^2*MEI_f^2-MBD_f
ID_outliers_manual <- which(d_manual>quantile(d_manual,probs = .75)+2*IQR(x = d_manual))
ID_outliers_manual
```

```
## girl03
##      3
```

```
plot(MEI_f, MBD_f, col=ifelse(1:N %in% ID_outliers_manual,"red","black"), pch=19)
```



3. Build a split conformal prediction band for a height curve using the previously computed functional median as point-wise predictor, the sup norm of absolute value of the functional residuals from the functional median as a NCM and the identity function as the modulation function. Consider a 70-30 percent split for calibration and validation sets, respectively.

```
alpha=.1
set.seed(2022)
i1=sample(1:N,N*2/3)
t_set=data.frame(df_3$height_values[i1,])
c_set=data.frame(df_3$height_values[-i1,])

mu <- median_fData(fData = fData(grid,df_3$height_values), type = "MBD")
mu <- mu$values
res=abs(sweep(x = c_set,MARGIN = 2,STATS = t(mu),"-"))

ncm=apply(res,2,max)
ncm_sort=c(sort(ncm),Inf)
```

```
d=ncm_sort[ceiling((N/3 + 1)*(1-alpha))]

matplot(cbind(t(mu),t(mu)+d,t(mu)-d),type='l')
```