Exam

Nonparametric Statistics, AY 2021/22

July 11, 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.[1]
- 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

Dr. Andreas Qapos Ph.D. is a Galician mathematician/fisherman, who proves theorems by day, and picks up goose barnacles [2], *percebes* in Spanish, an extremely pricey seafood delicacy typical of Galicia, by night. Goose barnacles live on rocks washed by the strong waves of the Atlantic Ocean. Dr. Qapos has identified two spots on the cliffs close to his hometown, Pontevedra, identified (since they are very secret) as spot A and spot B. Dr. Qapos suspects that the goose barnacles picked out of spot B tend to be slightly shorter than the ones in spot A, but plumpier. For this reason he has collected some barnacles from spot A and some (less, access to the cliff is terrible!) from spot B, and he measured their weight [g] and length [mm]. You can find the data in `percebes_1.rds`.

Now, assuming the tuple in each group $(weight, length)$ being $i.i.d$:

1. To help Dr. Qapos in assessing his hypothesis, start by computing the projection sample medians of the two groups

```
percebes=readRDS(here("2022-07-11/data/percebes_1.rds"))

p.median.a=depthMedian(percebes$spot.A,depth_params = list(method='Projection'))
p.median.b=depthMedian(percebes$spot.B,depth_params = list(method='Projection'))
p.median.a
```

```
##   length   weight
## 63.05206 34.90240
```

```
p.median.b
```

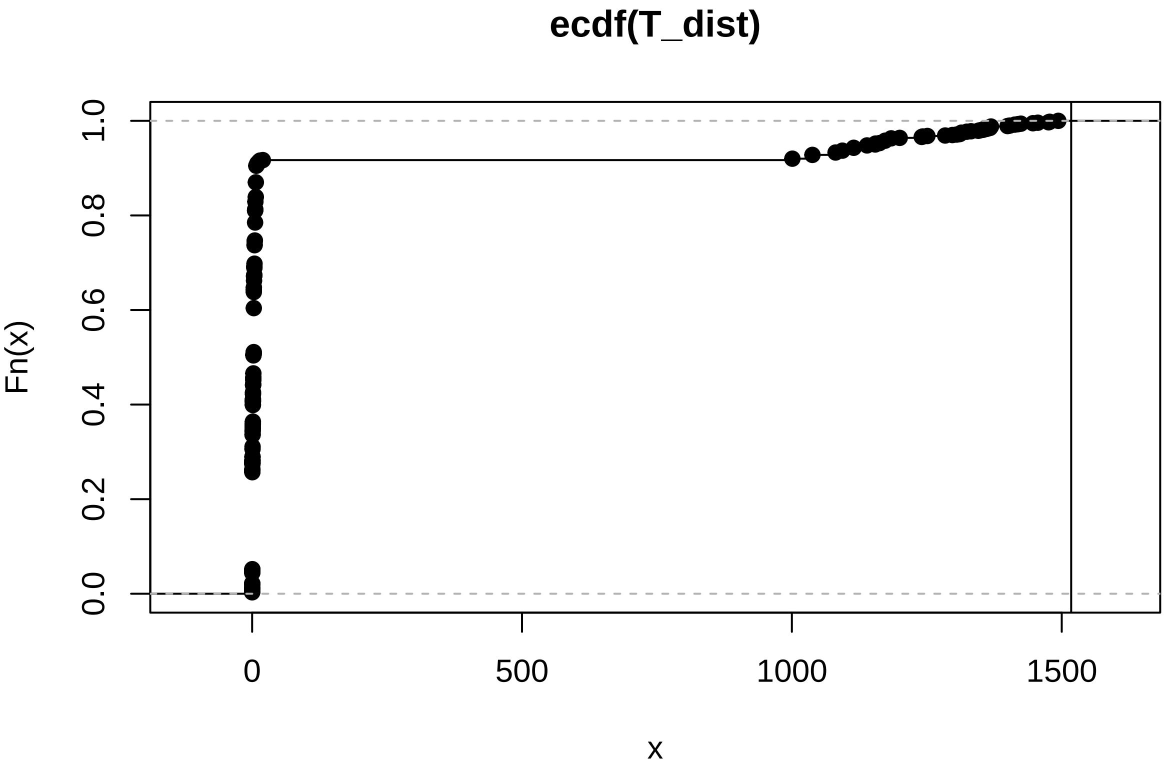
```
##   length   weight
## 35.69795 62.63765
```

[1] Be advised that, except for the number of points, these are the default conditions of the `ConformalInference`

[2] https://en.wikipedia.org/wiki/Pollicipes_pollicipes

2. To test the equality of the two theoretical projection medians, perform a two-sample permutation test using as a test statistics the squared euclidean distance between the two sample projection medians. Please describe briefly the properties of permutation tests and present the empirical cumulative distribution function of the permutational test statistic as well as p-value for the test.

```r
percebes$spot.A$spot='A'
percebes$spot.B$spot='B'

percebes_df=rbindlist(percebes)



B=1000
T_dist=numeric(B)
set.seed(2022)

t.stat=sum((p.median.b-p.median.a)^2)

n1=nrow(percebes$spot.A)
n2=nrow(percebes$spot.B)


for(index in 1:B){
  perm=sample(1:(n1+n2))
  percebes_df.p=percebes_df[perm,1:2]
  mean1.p=depthMedian(percebes_df.p[1:n1,],depth_params = list(method='Projection'))
  mean2.p=depthMedian(percebes_df.p[(n1+1):(n1+n2),],depth_params = list(method='Projection'))
  T_dist[index]=sum((mean2.p-mean1.p)^2)
}


plot(ecdf(T_dist))
abline(v=t.stat)
```



```
sum(T_dist>=t.stat)/B
```

```
## [1] 0
```

3. Is Dr. Qapos justified in risking his life to go for spot B, given that the barnacles are sold by their weight and not by their length? Justify your answer.

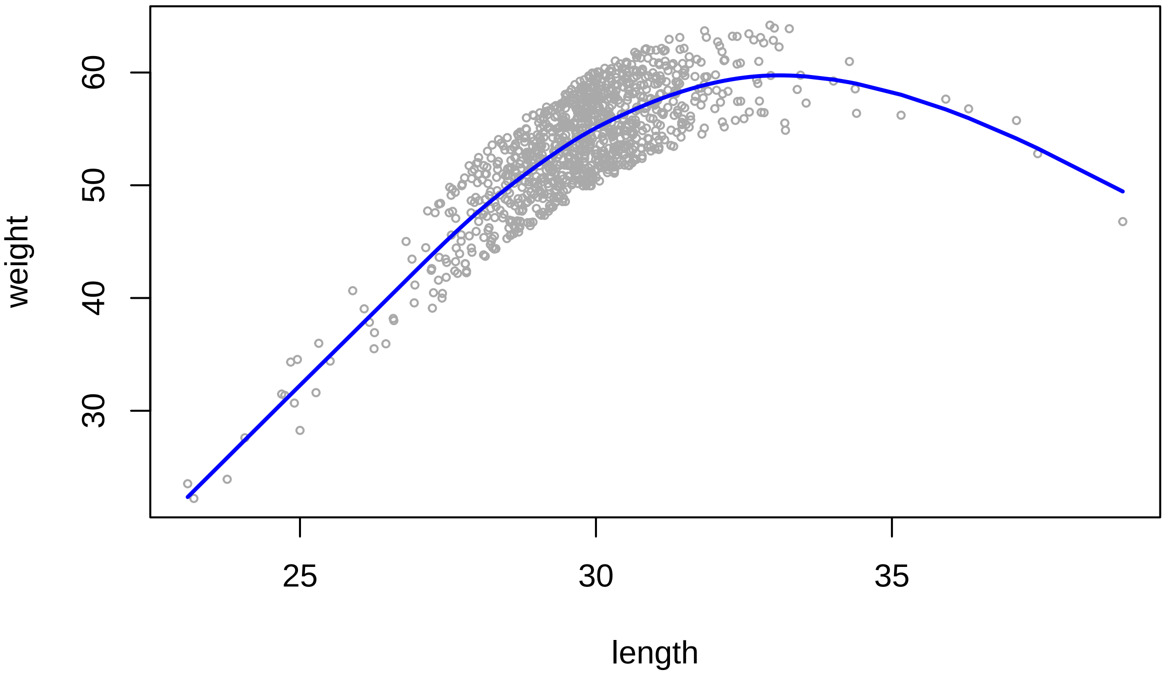
```
# Absolutely, let Dr. Qapos fall off those cliffs!
```

Exercise 2

Dr. Andreas Qapos, Ph.D, after having deducted from what you have told him after Exercise 1 that the relationship between length and weight of the barnacles is not a linear one, wants to understand what is the best time to harvest his precious goose barnacles. To do so, he has collected 999 barnacles from his best spot, and wants to build a prediction model for the weight of the barnacle (which can be assessed only after picking it up...) as a function of the length of the barnacle (which can be assessed using a special barnacle caliber). You can find the data in `percebes_2.rds`, Now

1. Help Dr. Qapos in building his model: specifically, he would like to use a smoothing spline model of order 4, with lambda selected via cross-validation. Please report the lambda value with 3 decimal digits, as well as a plot of the regression line, alongside the pointwise prediction of the weight of a barnacle whose length is 33 mm, which from a visual inspection of the data seems the optimal value for harvesting.

```
percebes2=readRDS(here("2022-07-11/data/percebes_2.rds"))
attach(percebes2)

fit=smooth.spline(length,weight,cv=T)
plot(length,weight,cex =.5, col =" darkgrey ")
lines(fit,col="blue",lwd=2)
```



```
opt=fit$lambd
names(opt)='Optimal Lambda'
round(opt,3)
```

```
## Optimal Lambda
##          0.003
```

```
pred=predict(fit,33)$y
names(pred)='Pointwise Prediction'
round(pred,2)
```

```
## Pointwise Prediction
##                59.74
```

2. Assess the uncertainty of this prediction by calculating, via residual bootstrapping, its bias, its variance and its MSE.

```
#residuals by bootstrap

n=nrow(percebes2)
B=1000
fitted=predict(fit,length)$y
residuals=weight-fitted

boot_d=numeric(B)
set.seed(2022)
for(sample in 1:B){

  weight.boot=fitted+sample(residuals,n,replace=T)
  new_model=smooth.spline(length,weight.boot,lambda = opt)
  boot_d[sample]=predict(new_model, 33)$y
}
```

```
variance2=var(boot_d)
bias2=mean(boot_d)-predict(fit,33)$y

MSE2= variance2 + bias2^2

data.frame(bias=bias2, variance=variance2, MSE=MSE2)
```

```
##         bias  variance       MSE
## 1 -0.1085574 0.1454632 0.1572479
```

3. Like every good statistician, Dr. Qapos is not happy at all with a pointwise prediction. After having defined the distributional hypotheses behind this approach, and stated its coverage properties, build a prediction interval for the weight of a barnacle 33mm long using a full conformal prediction approach, using as a non-conformity score the absolute value of the regression residuals.

```
train_ss=function(x,y,out=NULL){
  smooth.spline(x,y,lambda = opt)
}
predict_ss=function(obj, new_x){
  predict(obj,new_x)$y
}
set.seed(100)
pred=conformal.pred(length,weight,33,train_ss,predict_ss,alpha=0.05)
data.frame(lwr=pred$lo,pred=pred$pred,upr=pred$up)
```

```
##        lwr     pred     upr
## 1 55.92782 59.74053 64.0333
```

Exercise 3

Dr. Andrea Qapos, Ph.D would like to be able to assess the length of his barnacles from the cozyness of his seaside mansion. To do so he requires very good quality glass. Luckily, a former collegue of his is Dr. Mateo De la Fuente, Ph.D, a Mexican former academic statistician, who now turned to the glass-making industry. To this extent, he collects measurements of the presence of chemical constituents in 76 pieces of glass lenses he has produced: his aim is to evaluate how these compounds affect the refractive index (RI) of the glass pieces. In details, the considered chemical constituents (unit measurement weight percent in corresponding oxide) are: sodium oxide (Na20), magnesium oxide (MgO), aluminum oxide (Al2O3), silcon oxide (SiO2) potassium oxide (K2O) and calcium oxide (CaO). The resulting samples are contained in the `glass_3.rds` file. Mateo De la Fuente wants to reduce his scraps to zero, he thus prefers to employ robust methods to analyse his data as some defective parts may have been produced. He therefore asks you to:

1. Compute the Minimum Covariance Determinant estimator for the chemical constituents (i.e., all the variables in the dataset but the refractive index RI). Consider 1000 subsets for initializing the algorithm and set `alpha` equal to 0.5. Report the reweighed MCD estimates of location and scatter. Define a vector `ind_out_MCD_rw` of row indexes identifying the glass samples that are outliers according to the final MCD estimates[3] and report it.

```
df_3 <- readRDS(here("2022-07-11/data/glass_3.rds"))
X_mcd <- df_3[,-1]
N <- nrow(X_mcd)
p <- ncol(X_mcd)

alpha_mcd <- .5

set.seed(2022)
```

[3] Hint: the observations NOT used in the calculation of the final estimate for the location and scatter

```
fit_MCD <-
  covMcd(
    x = X_mcd,
    alpha = .5,
    nsamp = 1000
  )

fit_MCD$center
```

```
##       Na2O        MgO      Al2O3       SiO2        K2O        CaO
## 13.1628261  3.6347826  1.4669565 72.6986957  0.6117391  8.1832609
```

```
fit_MCD$cov
```

```
##              Na2O          MgO       Al2O3        SiO2          K2O         CaO
## Na2O   0.12476845  0.029190693 -0.02167778 -0.11150555 -0.010387464 -0.01217078
## MgO    0.02919069  0.042164388 -0.03172186 -0.05459884 -0.008384621  0.02407330
## Al2O3 -0.02167778 -0.031721860  0.08296551  0.02084749  0.010849647 -0.05627259
## SiO2  -0.11150555 -0.054598843  0.02084749  0.20527543  0.014573755 -0.05738992
## K2O   -0.01038746 -0.008384621  0.01084965  0.01457375  0.006116732 -0.01180291
## CaO   -0.01217078  0.024073296 -0.05627259 -0.05738992 -0.011802908  0.10215956
```

```
ind_obs_MCD_rw <-
  which(
    mahalanobis(
      x = X_mcd,
      center = fit_MCD$raw.center,
      cov = fit_MCD$raw.cov
    ) <= qchisq(p = .975, df = p)
  )
ind_out_MCD_rw <- setdiff(1:N,ind_obs_MCD_rw)
ind_out_MCD_rw
```

```
##  [1]  1  9 15 17 22 23 24 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 58
## [26] 59 60 61 62 75
```

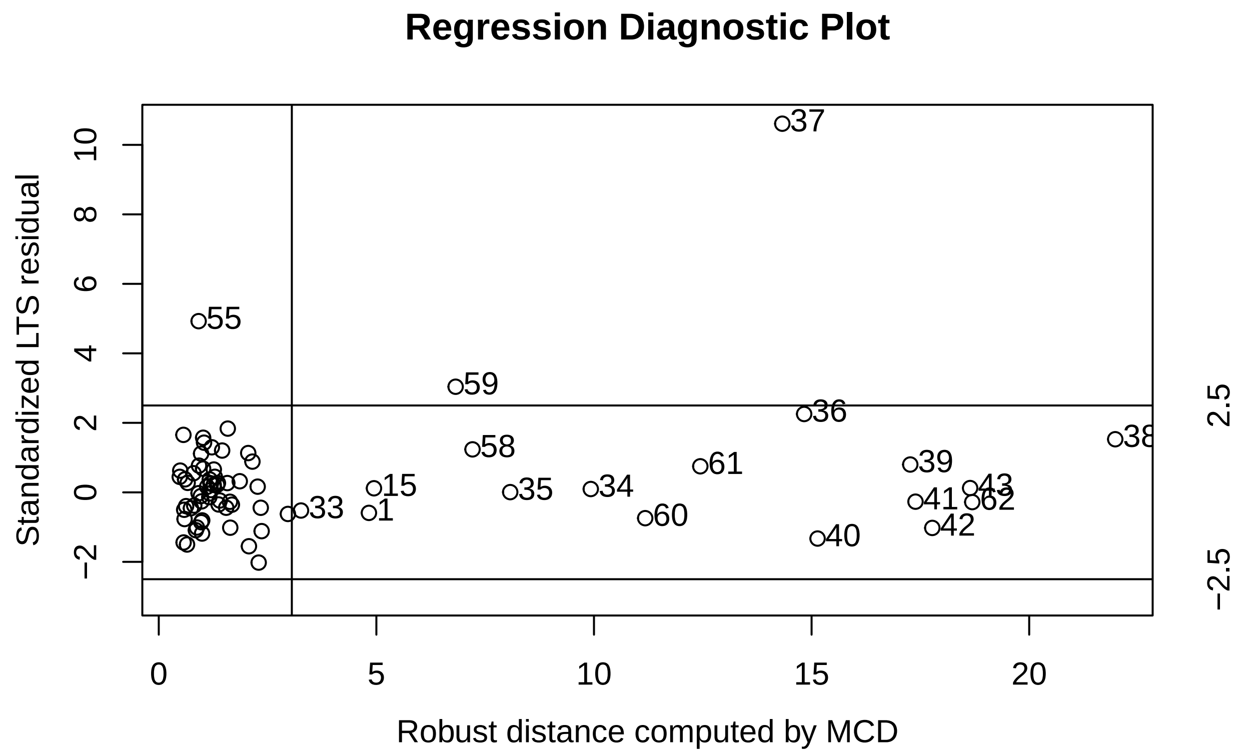
```
# Alternatively
ind_out_MCD_rw_2 <- (1:N)[fit_MCD$mcd.wt==0]
```

2. Build a robust linear model to regress the refractive index (RI) on the sodium oxide (Na20), calcium oxide (CaO) and magnesium oxide (MgO) using a Least Trimmed Squares (LTS) approach, setting the hyperparameter $\alpha = 0.75$. Report the table of robustly estimated coefficients. Plot the resulting outlier map and report the row indexes of bad leverage points and vertical outliers present in the dataset according to the diagnostic plot.

```
fit_lts <-
  ltsReg(RI ~ Na2O+CaO+MgO, alpha = 0.75, data = df_3)

summary(fit_lts)
```

```
##
## Call:
## ltsReg.formula(formula = RI ~ Na2O + CaO + MgO, data = df_3,
##     alpha = 0.75)
##
## Residuals (from reweighted LS):
##        Min         1Q     Median         3Q        Max
## -0.0015962 -0.0003691  0.0000000  0.0003521  0.0014484
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## Intercept 1.4720643  0.0028815  510.87  < 2e-16 ***
## Na2O      0.0010729  0.0001531    7.01 1.36e-09 ***
## CaO       0.0029359  0.0001079   27.20  < 2e-16 ***
## MgO       0.0018669  0.0001679   11.12  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0006802 on 68 degrees of freedom
## Multiple R-Squared: 0.9636,  Adjusted R-squared: 0.962
## F-statistic: 600.9 on 3 and 68 DF,  p-value: < 2.2e-16
```

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



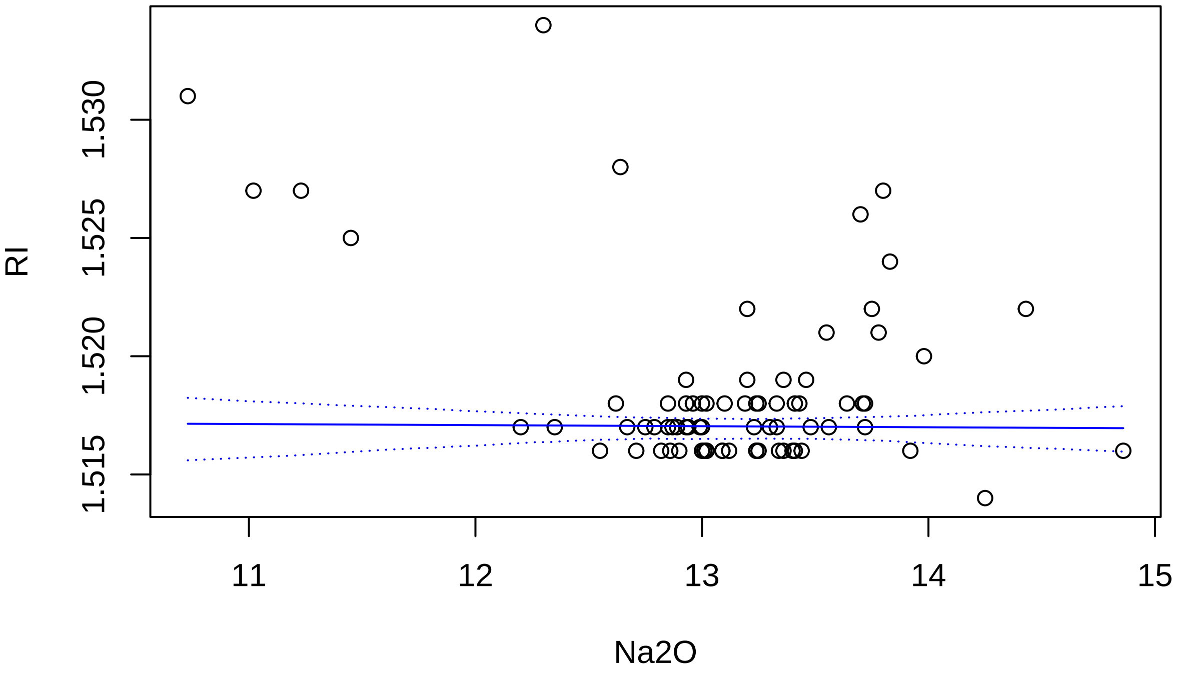
```
# 37 and 59 bad leverage points
# 55 vertical outlier
```

3. Simplify the robust linear model of the previous exercise regressing the refractive index (RI) on the sodium oxide (Na20) only. Using a bootstrap approach, provide the reverse percentile confidence intervals for the corresponding mean value of the RI. Plot the result.[4]

```
fit_lts <-
  ltsReg(RI ~ Na2O, alpha = 0.75, data = df_3)
fitted.obs <- fit_lts$fitted.values
res.obs <- fit_lts$residuals

Na2O_grid = seq(range(df_3$Na2O)[1],
```

[4] Not easy, as the `lts` class does not have a predict method and everything needs to be hard-coded. Try to solve this only if you have already finished all the other exercises!

```
                    range(df_3$Na2O)[2],
                    length.out = 100)

preds = c(cbind(1, Na2O_grid) %*% fit_lts$coefficients)

with(df_3,
     plot(Na2O ,
          RI))

lines(Na2O_grid, preds, col="blue")
set.seed(2022)
B <- 1000
preds_boot_container <- matrix(ncol = length(Na2O_grid), nrow = B)
pb = progress::progress_bar$new(total = B,
                                format = " Processing [:bar] :percent eta: :eta")
for(b in 1:B) {

  RI_boot <- fitted.obs + sample(res.obs, replace = T)
  fit_lts_boot <-
    ltsReg(RI_boot ~ df_3$Na2O, alpha = 0.75)
  preds_boot = c(cbind(1, Na2O_grid) %*% fit_lts_boot$coefficients)
  preds_boot_container[b,] = preds_boot
pb$tick()
}

alpha <- 0.05
right.quantile.preds <- apply(preds_boot_container,2,quantile,probs=1 - alpha/2)
left.quantile.preds  <- apply(preds_boot_container,2,quantile,probs=alpha/2)

CI.preds <-
  list(
    low = preds - (right.quantile.preds - preds),
    up = preds - (left.quantile.preds - preds)
  )

matlines(
  Na2O_grid ,
  cbind(CI.preds$up, CI.preds$low) ,
  lwd = 1,
  col = " blue",
  lty = 3
)
```


RI
1.515
1.520
1.525
1.530
11
12
13
14
15
Na2O