Laboratory 4

Ex. 1

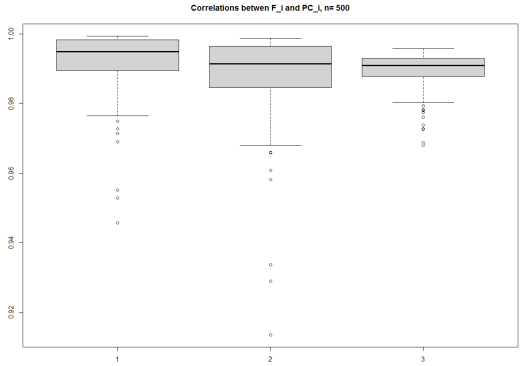
The task is to generate design matrix $X_{n\times p}$ according to formula (code 1.1):

$$X = F_{n\times r}(W_{p\times r})^T + E_{n\times p}$$

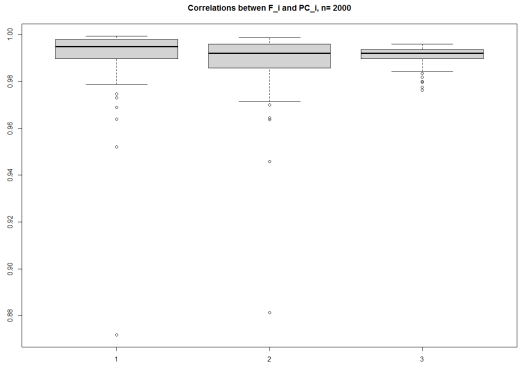
Where $F_{,i} \sim N(0, I_{n\times n})$, $W_{,j} \sim (r+1-j)N(0, I_{p\times p})$, $E_{ij} \sim N(0, \sigma^2)$ and then performing PCA using eigenvalue decomposition of X^TX (or singular value decomposition). We are interested to see, how closely related are columns of F matrix and estimated principal components and how well can we estimate σ using eigenvalues. Estimator of $\hat{\sigma}^2$ has the following form:

$$\hat{\sigma}^2 = \frac{1}{n-1}\frac{1}{p-r}\sum_{i=r+1}^{p}\lambda_i$$

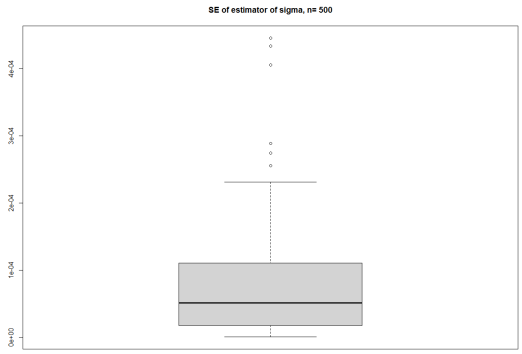
Where λ_i is the i-th eigenvalue for X^TX in decreasing order. Figure 1.1 presents results of 100 replications for setups (code 1.2): $n = 500, 2000$, $p = 100$, $r = 3$, $\sigma = 1$.



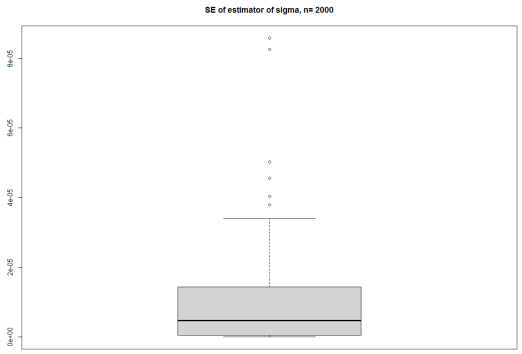
(a) Correlations between F_i and PC_i for $n = 500$.



(b) Correlations between F_i and PC_i for $n = 2000$.



(c) $(\hat{\sigma}^2 - \sigma^2)^2$ for $n = 500$.



(d) $(\hat{\sigma}^2 - \sigma^2)^2$ for $n = 2000$.

Figure 1.1: Boxplots of correlations and squared errors.

Both methods give the same results in this case, so the one choosen was eigenvalue decomposition. Boxplots with correlations, present only the strenght of correlation (no signs are taken into account). We can see that indeed, matrix F is strongly correlated with principal components no matter the n. Also, estimator of σ based on the rest (and smaller) eigenvalues is close to the true value, with higher n the difference is smaller.

Ex. 2

The task is to generate design matrix X the same as previous task, but this time we are interested in number of dimensions to choose. It is done by PESEL procedure. Figure 2.1 presents results of 100 replications for setups (code 2.1): $n = 50$, $p = 100, 500, 1000, 5000$, $r = 5$, $\sigma = 10$.

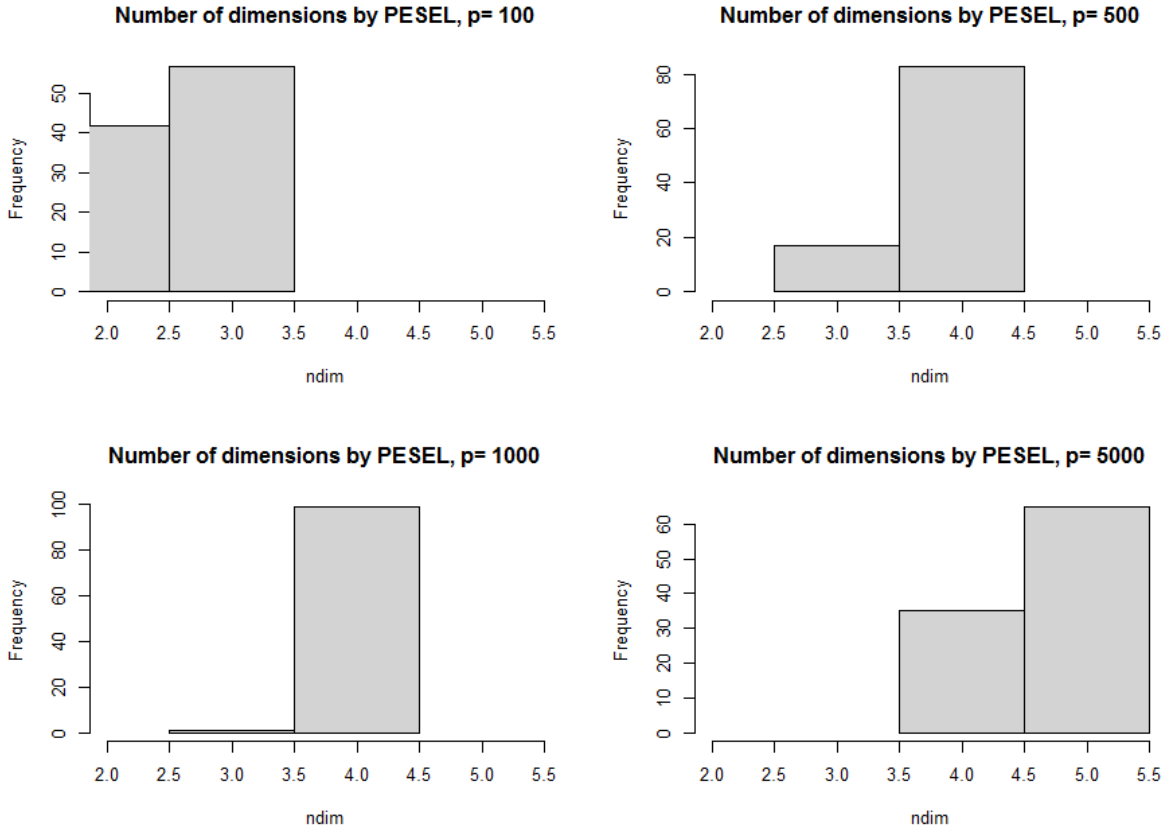


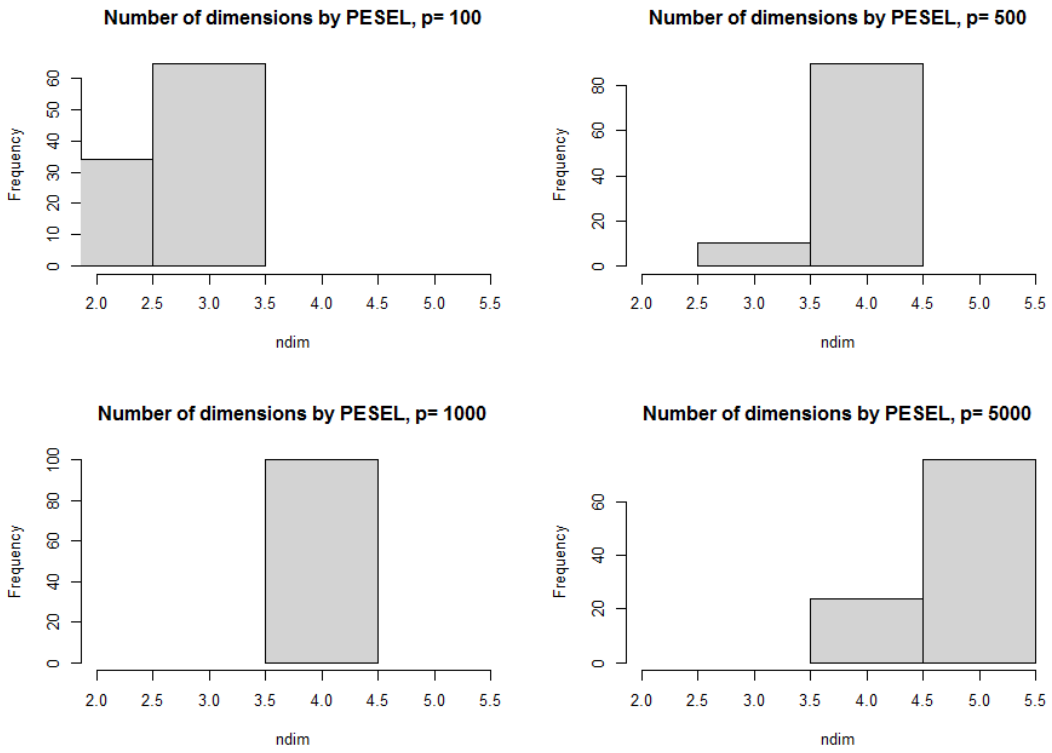
Figure 2.1: Histograms of number of dimensions choosen by PESEL.

There is little to no varying of number of dimensions. The number increases with p. PESEL tends to choose less dimensions than there are true ones. Only with high $p = 5000$ it chooses the right amount, but there are still a lot of errors (4's). In low dimensional setup of $p = 100$, it is almost equaly likely to choose between 2 and 3 dimensions.

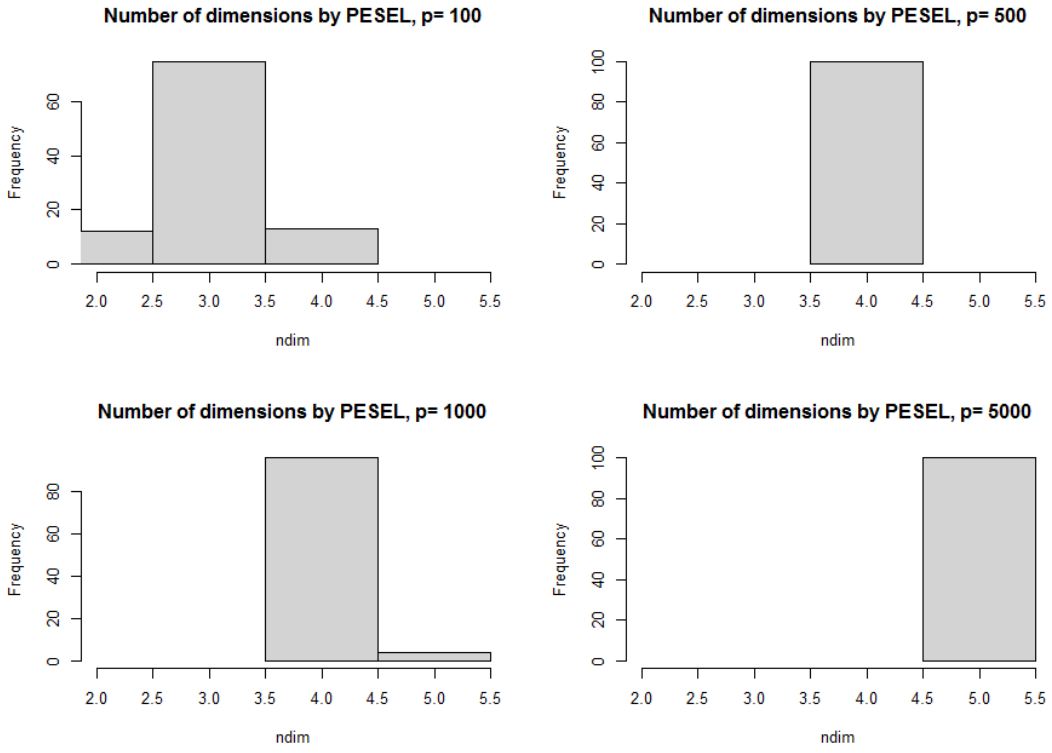
Ex. 3

The task is to repeat previous task, but with different distributions of E_{ij}: $Exp(0.1)$ and $Cauchy(0, 1)$. Figure 3.1 presents results of 100 replications (code 3.1).

There is almost no difference between those setups and the one from the previous task. In case of exponential distribution there is even improvement in $p = 5000$ – there is lower probability of choosing 4. It could also mean, that there is a need for more replications to asure this conclusion. In case of Cauchy distribution, there is tendancy for higher dimensions.



(a) $Exp(0.1)$



(b) $Cauchy(0, 1)$

Figure 3.1: Histograms of number of dimensions choosen by PESEL.

Ex. 4

The task is to generate 100 independent replicated of the design matrix $X_{50\times 800}$ according to the formula:

$$X = \left[X^1, \ldots, X^4\right]$$

Where $X^i_{50\times 200}$ and is obtained as in first task. Then generate vector of response variable according to the formula:

$$Y = F\beta + Z$$

Where $F = \left[F^1, \ldots, F^4\right]$, $\beta = [0, 0.5, 0.5, 0, 0.5, 0.5, 0, 0, 0, 0, 0, 0]$. Code at 4.1

a) using PESEL and *varclust*, drawing the histograms of the estimated number of dimensions and number of clusters (fig. 4.1):

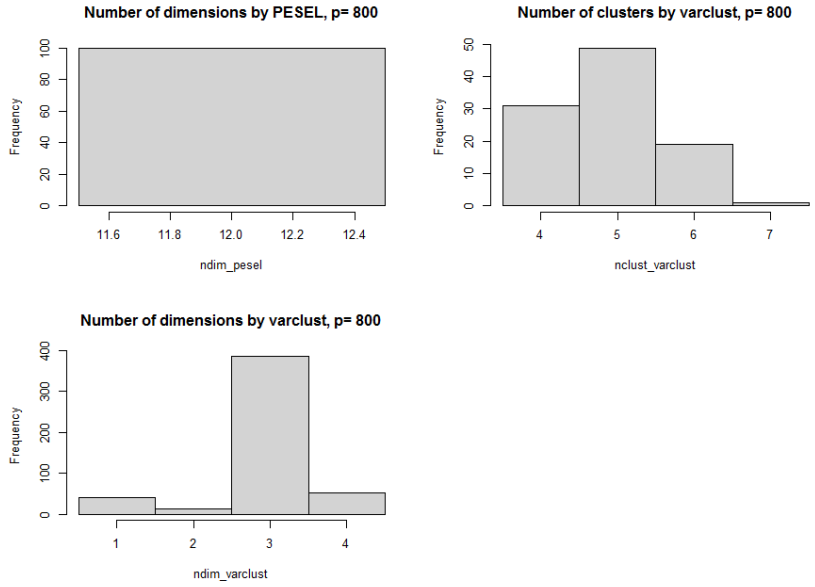


Figure 4.1: Histograms of number of dimensions (and clusters) choosen by PESEL and *varclust*.

b) using *varclust*, calculating values of measures of similarity and drawing histograms (fig. 4.2) and boxplots (fig. 4.3):

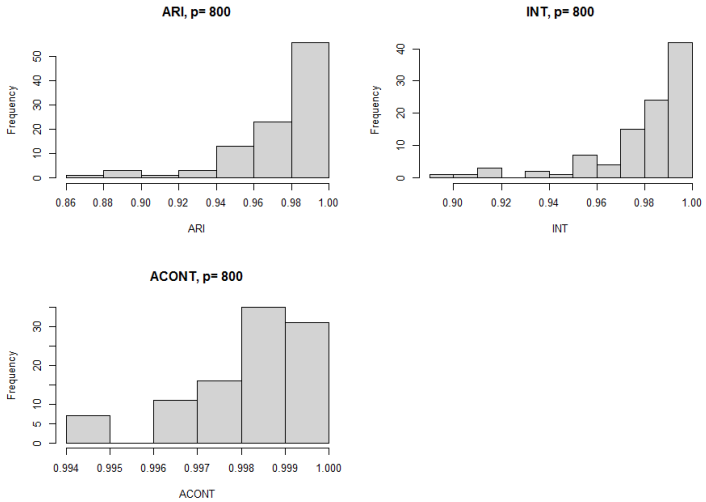


Figure 4.2: Histograms of values of ARI, Integration and Acontamination.

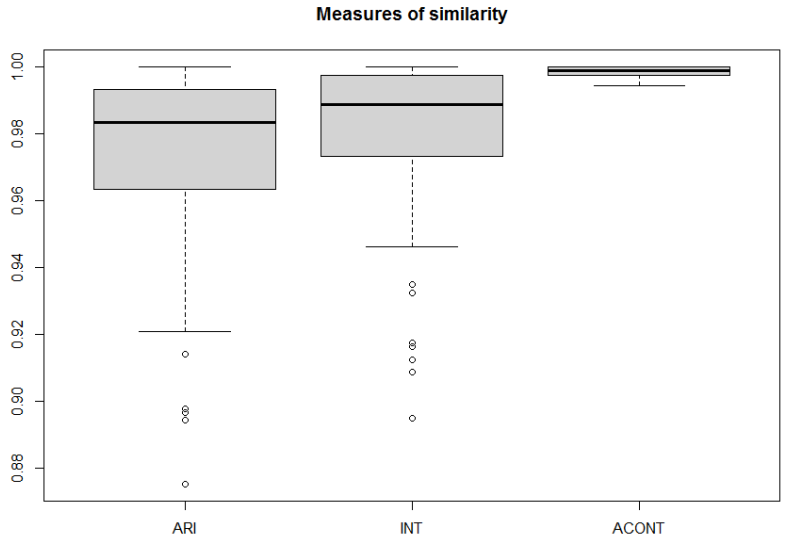


Figure 4.3: Boxplots of values of ARI, Integration and Acontamination.

c) fitting the predictive model for Y based on X using LASSO with CV, Adaptive LASSO, OLS using PCs and drawing boxplots (fig. 4.4) of SSE:

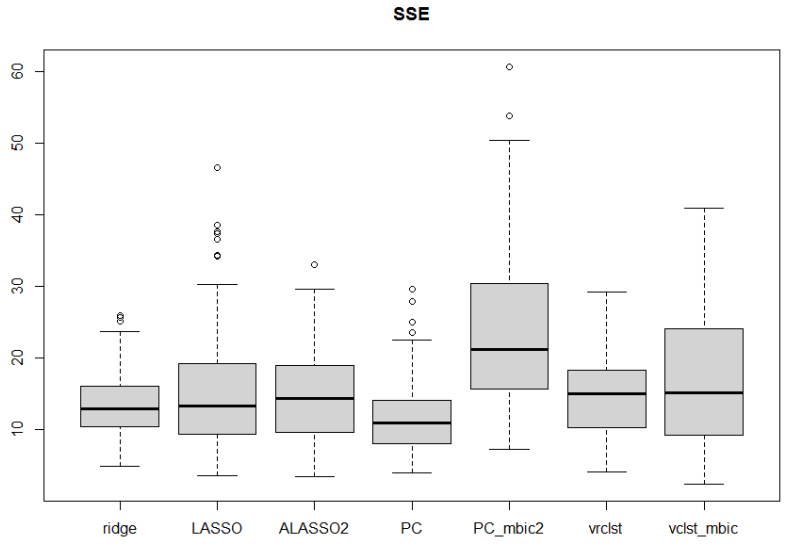


Figure 4.4: Boxplots of $\|\mu - \hat{\mu}\|^2$ for different methods.

PESEL always chooses 12 dimensions, which is the right amount. *Varclust* is using 4-5 clusters with 3 dimensions, which is almost right ($4 \cdot 3 = 12$).

Measures of similarity (quality of clustering) ARI, Integration and Acontaminaion are close to 1, which means almost perfect clustering. Acontamination is highly concentrated on 1.

Medians of sums of squared errors hold closly to 10. MBIC2 seems not to improve methods that implement principal components. It's probably because, those methods often guess the right amout of dimensions and MBIC2 is removing some of those right dimensions. ALASSO2 is using estimator of σ based on the eigenvalues (like in 1st task) and only eliminantes outliries compared to regular LASSO.

Overall the ordinary least squares based on principal components is the best method in this setup and also the one with least computation involved.

Appendices

Ex. 1

Listing 1.1: Generating data and additional functions.

```
center <- function(X) apply(X, 2, function(x)x-mean(x))
scale <- function(X) apply(X, 2, function(x)(x-mean(x)) /sd(x) )
#Ex. 1
generate_setup <- function(n=500, p=100, r=3) {
  F_r = matrix(rnorm(n*r),n,r)
  W = matrix(rnorm(p*r),p,r)
  W = sweep(W,2,(r+1 - 1:r),'*')
  return(list(F_r=F_r, W=W))
}

generate_design <- function(F_r, W, rdist=rnorm) {
  n = nrow(F_r)
  p = nrow(W)
  E  = matrix(rdist(n*p), n, p)
  return(F_r %*% t(W) + E)
}
```

Listing 1.2: Boxplots of correlations and SE of $\hat{\sigma}$.

```
ex1 <- function(n=500, p=100, r=3, sigma=1, rep=100) {
  COR = matrix(0,rep,r); sigma_hat = NULL
  setup  = generate_setup(n,p,r)
  F_r = setup$F_r
  W = setup$W
  for (i in 1:rep) {
    X = center(generate_design(F_r, W))
    C = t(X) %*% X
    ev = eigen(C)
    PC = X %*% ev$vectors[,1:r]
    sigma_hat[i] = sum(ev$values[(r+1):p]) / (p-r) / (n-1)

    for (j in 1:r) {
      COR[i,j] = cor(PC[,j], F_r[,j])
    }

  }
  boxplot(abs(COR), main=paste0("Correlations betwen F_i and PC_i, n= ", n)
      )
  boxplot((sigma_hat-sigma)^2, main=paste0("SE of estimator of sigma, n= ",
      n))
  return(mean((sigma_hat-sigma)^2))
}
```

Ex. 2

Listing 2.1: Boxplots of number of dimensions choosen by PESEL.

```
library(pesel)
ex2 <- function(N=50, P=c(1,5,10,50)*100, r=5, rep=100, rdist=function(x)
    rnorm(x,sd=10)) {
  results = NULL
  for (n in N) {
```

```
    par(mfrow=c(2,2))
    for (p in P) {
      ndim = NULL
      for (i in 1:rep) {
        setup = generate_setup(n, max(P), r)
        X = generate_design(setup$F_r, setup$W[1:p,], rdist)
        ndim[i] = pesel(X, npc.max = 20)$nPCs
        #cat("p= ",p, "rep=",i, "\n")
      }
      results = cbind(results, ndim)
      cat("p=",p,"\n"); print(table(ndim))
      hist(ndim, main=paste0("Number of dimensions by PESEL, p=",p),
          breaks = (min(ndim)-0.5):(max(ndim)+0.5), xlim = c(2,5.5))
    }
    colnames(results) = paste(P)
    #boxplot(results, xlab="p", main="Number of dimensions by PESEL")
  }
  par(mfrow=c(1,1))
  return(results)
```

Ex. 3

Listing 3.1: Boxplots of number of dimensions choosen by PESEL for different distributions.

```
ex2(rdist=function(x)rexp(x,rate=0.1))
ex2(rdist=function(x)rcauchy(x))
```

Ex. 4

Listing 4.1: Comparison of all methods.

```
library(varclust)
library(mclust)
library(glmnet)
library(bigstep)
norm2 <- function(X,Y) sum((X-Y)^2)
adaptive_lasso2_select <- function(X, Y, beta_hat, sigma_lassoCV, q=.2) {
  n = nrow(X); p = ncol(X); lambda_alasso = qnorm(1-q/2/p) #CONSTANTS

  RSS = sum((Y- X%*%beta_hat)^2)
  nonzero_id = which(beta_hat!=0)
  X_nonzero = X[,nonzero_id]
  beta_nonzero =  beta_hat[nonzero_id]
  l = length(nonzero_id)
  #sigma_lassoCV = sqrt(RSS/(n-l))

  W = sigma_lassoCV/abs(beta_nonzero)
  X_nonzero = sweep(X_nonzero,2,W,'/')

  ad_lasso = glmnet(X_nonzero, Y, intercept=FALSE, standardize=FALSE,lambda
      =sigma_lassoCV*lambda_alasso/n)
  beta_nonzero = rep(0,p)
  beta_nonzero[nonzero_id] = coef(ad_lasso)[-1,1] / W
  return(beta_nonzero)
}

BETAS = c(0, 0.5, 0.5, 0, 0.5, 0.5, 0, 0, 0, 0, 0, 0)
ex4 <- function(n=50, p=800, K=4, r=3, betas=BETAS, rep=100) {
  ndim_pesel=NULL; nclust_varclust=NULL; ndim_varclust=NULL;
```

```
ARI=NULL; INT=NULL; ACONT=NULL; SSE = matrix(0, rep, 7)

true_lab=NULL;
for (j in 1:K) true_lab = c(true_lab, rep(j, p/K))

for (i in 1:rep) {
  t0 = proc.time()

  X_K=NULL; F_K=NULL;
  for (k in 1:K) {
    setup = generate_setup(n,p/K,r)
    F_K = cbind(F_K, setup$F_r)
    X_K = cbind(X_K, generate_design(setup$F_r, setup$W))
  }
  Y = F_K%*%betas + rnorm(n)
  mu = F_K%*%betas

  ndim_pesel[i] = pesel(X_K, npc.max = 20)$nPCs
  X_cent = center(X_K)
  C = t(X_cent) %*% X_cent
  ev = eigen(C)
  sigma_hat = sum(ev$values[(ndim_pesel[i]+1):p]) / (p-ndim_pesel[i]) / (
      n-1)
  PC = X_K %*% ev$vectors[,1:ndim_pesel[i]]
  PC_mbic2 = as.numeric(fast_forward(prepare_data(Y,PC),crit=mbic2)$model
      )

  varclust_model = mlcc.bic(X_K)
  varclust_factors = do.call(cbind, varclust_model$factors)
  varclust_mbic2 = fast_forward(prepare_data(Y, varclust_factors),crit=
      mbic2)$model

  nclust_varclust[i] = varclust_model$nClusters
  ndim_varclust = c(ndim_varclust, unlist(varclust_model$
      subspacesDimensions))
  ARI[i] = adjustedRandIndex(varclust_model$segmentation, true_lab)
  INT[i] = integration(varclust_model$segmentation, true_lab)[1]
  ACONT[i] = integration(varclust_model$segmentation, true_lab)[2]

  ridge = predict(cv.glmnet(X_K,Y,alpha=0,intercept=FALSE, standardize=
      FALSE), X_K, s='lambda.min')
  LASSO_model = cv.glmnet(X_K,Y,intercept=FALSE, standardize=FALSE)
  LASSO_coef = coef(LASSO_model, s='lambda.min')[-1,1]
  LASSO = predict(LASSO_model, X_K, s='lambda.min')
  if(sum(LASSO_coef!=0)>0)
    ALASSO2 = X_K %*% adaptive_lasso2_select(X_K,Y,LASSO_coef,sigma_hat
        ,0.1)
  else
    ALASSO2 = 0

  OLS_PC = predict(lm(Y~PC-1))
  OLS_varclust = predict(lm(Y~varclust_factors-1))

  if(length(PC_mbic2)>0)
    OLS_PC_mbic2 = predict(lm(Y~PC[,PC_mbic2]-1))
  else
    OLS_PC_mbic2 = 0
```

```
    if(length(varclust_mbic2)>0)
      OLS_varclust_mbic = predict(lm(Y~varclust_factors[,varclust_mbic2]-1)
          )
    else
      OLS_varclust_mbic = 0

    prediction_list = list(ridge, LASSO, ALASSO2, OLS_PC, OLS_PC_mbic2, OLS
        _varclust, OLS_varclust_mbic)
    for (j in 1:length(prediction_list)) {
      pred = prediction_list[[j]]
      SSE[i,j] = norm2(pred, mu)
    }
    cat("rep= ",i," time= ", (proc.time()-t0)[3], "\n")
  }
  save(list=c("SSE"), file="SL_lab3_ex4_SSE.RData")

  save(list=c("ndim_pesel"), file="SL_lab3_ex4_ndim_pesel.RData")
  save(list=c("nclust_varclust"), file="SL_lab3_ex4_nclust_varclust.RData")
  save(list=c("ndim_varclust"), file="SL_lab3_ex4_ndim_varclust.RData")

  save(list=c("ARI"), file="SL_lab3_ex4_ARI.RData")
  save(list=c("INT"), file="SL_lab3_ex4_INT.RData")
  save(list=c("ACONT"), file="SL_lab3_ex4_ACONT.RData")

  par(mfrow=c(2,2))
  hist(ndim_pesel, main=paste0("Number of dimensions by PESEL, p=",p),
      breaks = (min(ndim_pesel)-0.5):(max(ndim_pesel)+0.5))
  hist(nclust_varclust, main=paste0("Number of clusters by varclust, p=",p
      ), breaks = (min(nclust_varclust)-0.5):(max(nclust_varclust)+0.5))
  hist(ndim_varclust, main=paste0("Number of dimensions by varclust, p=",p
      ), breaks = (min(ndim_varclust)-0.5):(max(ndim_varclust)+0.5))
  par(mfrow=c(1,1))

  par(mfrow=c(2,2))
  hist(ARI, main=paste0("ARI, p=",p))
  hist(INT, main=paste0("INT, p=",p))
  hist(ACONT, main=paste0("ACONT, p=",p))
  par(mfrow=c(1,1))

  colnames(SSE) = c("ridge", "LASSO", "ALASSO2", "PC", "PC_mbic2", "vrclst"
      , "vclst_mbic")
  boxplot(SSE, main="SSE")
  #boxplot(cbind(ndim_pesel, nclust_varclust, ndim_varclust), main="Number
      of dimensions and clusters")
  boxplot(cbind(ARI, INT, ACONT), main="Measures of similarity")
}

ex4(rep=100)
```