Multivariate Regression und Klassifikation mit Anwendungen aus der Chemometrie

Peter Filzmoser

Institut für Statistik und Wahrscheinlichkeitstheorie
Technische Universität Wien

Herbstseminar des WBS, Wien

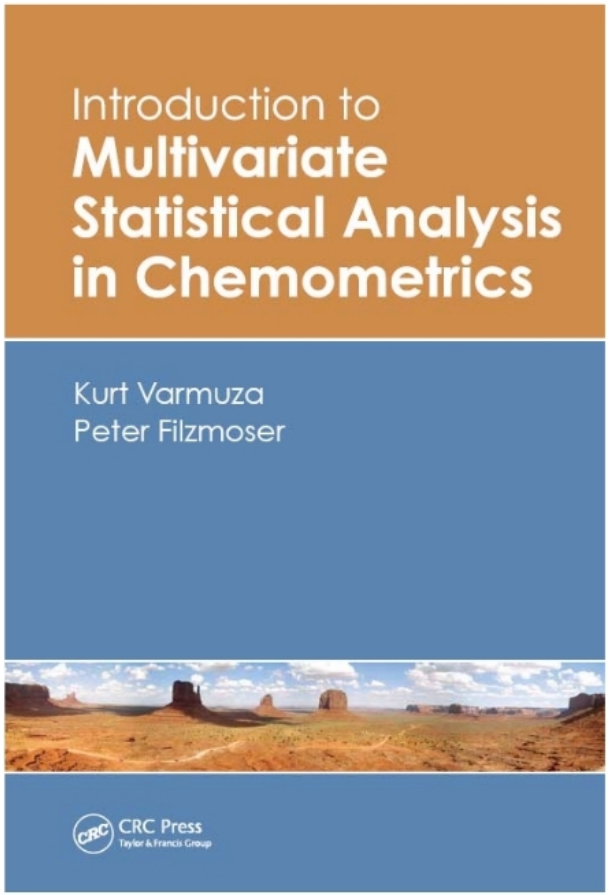
21. November 2013

Vienna University of Technology

Overview

321 + xiii pages; around 100 figures; appeared in 2009.

Written by a **chemometrician** and a **statistician**, reflects both the **practical** approach to chemometrics and the more **formally oriented** one of statistics.



Peter.Filzmoser@tuwien.ac.at

Regression in high dimension

Consider the **multiple linear regression model**

$$y_i = \boldsymbol{x}_i^\top \boldsymbol{\beta} + \varepsilon_i \quad \text{for } i = 1, \ldots, n,$$

with
$\boldsymbol{y} = (y_1, \ldots, y_n)^\top$
$\boldsymbol{x}_i^\top = (1, x_{i1}, \ldots, x_{ip})$, forming the rows of $\boldsymbol{X}$,
$\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^\top$.

The **ordinary least-squares solution**

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

cannot be computed if $\boldsymbol{X}^\top \boldsymbol{X}$ is (near) singular. This happens typically in case of **multicollinearity or** $n < p$.

Peter.Filzmoser@tuwien.ac.at

Principal Component Regression (PCR)

Way out: Dimension reduction in the $\boldsymbol{X}$ space.

Assume that the $n \times p$ matrix $\boldsymbol{X}$ is mean-centered.
Representation with the first k **principal components (PCs)**:

$$\boldsymbol{X} = \boldsymbol{Z}\boldsymbol{V}^\top + \boldsymbol{E}$$

with $\boldsymbol{Z}$ the $n \times k$ matrix of the first k PCs
$\boldsymbol{V}$ the $p \times k$ matrix with the first k PC loadings
$\boldsymbol{E}$ the error matrix ("reconstruction error")

Principal Component Regression:

$$\boldsymbol{y} = \boldsymbol{Z}\boldsymbol{\theta} + \varepsilon$$

with new regression coefficients $\boldsymbol{\theta}$ and errors ε.

Important: "tuning" parameter k

Peter.Filzmoser@tuwien.ac.at

Principal Component Regression (PCR)

PCR is a “technical” solution to the problem of multicollinearity:

- The PCs $\boldsymbol{Z}$ contain the “most important” information of $\boldsymbol{X}$. But is this also the most important part for explaining $\boldsymbol{y}$?
- Why take the first k PCs, and not a “variable selection” of PCs?
- The real problem is to find that information of $\boldsymbol{X}$ that is best suited for explaining $\boldsymbol{y}$.

Solution: Partial Least Squares (PLS) regression

Peter.Filzmoser@tuwien.ac.at

Partial Least Squares (PLS) regression

Use again a **"latent variable model"** to explain $\boldsymbol{X}$:

$$\boldsymbol{X} = \boldsymbol{T}\boldsymbol{A}^\top + \boldsymbol{E}$$

with *scores* $\boldsymbol{T}$ $(n \times k)$
and *loadings* $\boldsymbol{A}$ $(p \times k)$

The columns of $\boldsymbol{A}$ are derived by

$$\boldsymbol{a}_j = \underset{\boldsymbol{a}}{\text{argmax}}\, \text{Cov}(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{a})$$

$(j = 1, \ldots, k)$.
Then we have the **new regression model**:

$$\boldsymbol{y} = \boldsymbol{T}\boldsymbol{\gamma} + \varepsilon = (\boldsymbol{X}\boldsymbol{A})\boldsymbol{\gamma} + \varepsilon$$

$$= \boldsymbol{X}\underbrace{(\boldsymbol{A}\boldsymbol{\gamma})}_{\boldsymbol{\beta}} + \varepsilon = \boldsymbol{X}\boldsymbol{\beta} + \varepsilon$$

Peter.Filzmoser@tuwien.ac.at

Partial Least Squares (PLS) regression

Important for PLS: selection of the "optimal" number k of PLS components.

- k too small: underfit; poor prediction ability of the model
- k too large: overfit, and again poor prediction ability of the model

Model evaluation requires **training**, **validation** and **test** data.

Since we have **only one data set** available, we have to split appropriately.

Performance measures for regression

Popular in chemometrics: **Standard Error of Prediction (SEP):**

$$\text{SEP} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \hat{y}_i - \text{bias})^2}$$

with

$$\text{bias} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)$$

N is the number of test set observations.

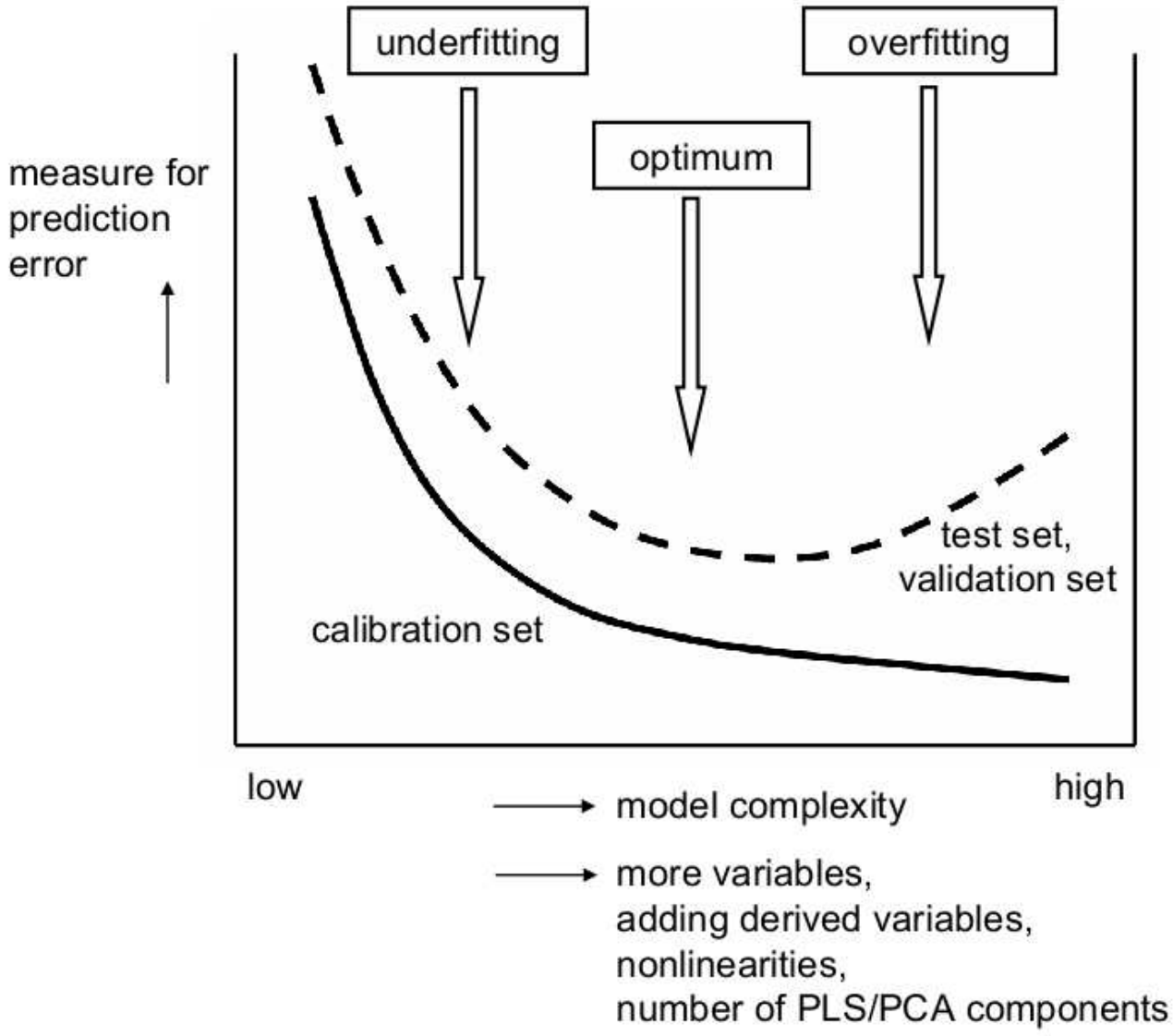
Popular in many fields: **Mean Squared Error of Prediction (MSEP):**

$$\text{MSEP} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

We see that $\text{SEP}^2 = \text{MSEP} - \text{bias}^2$

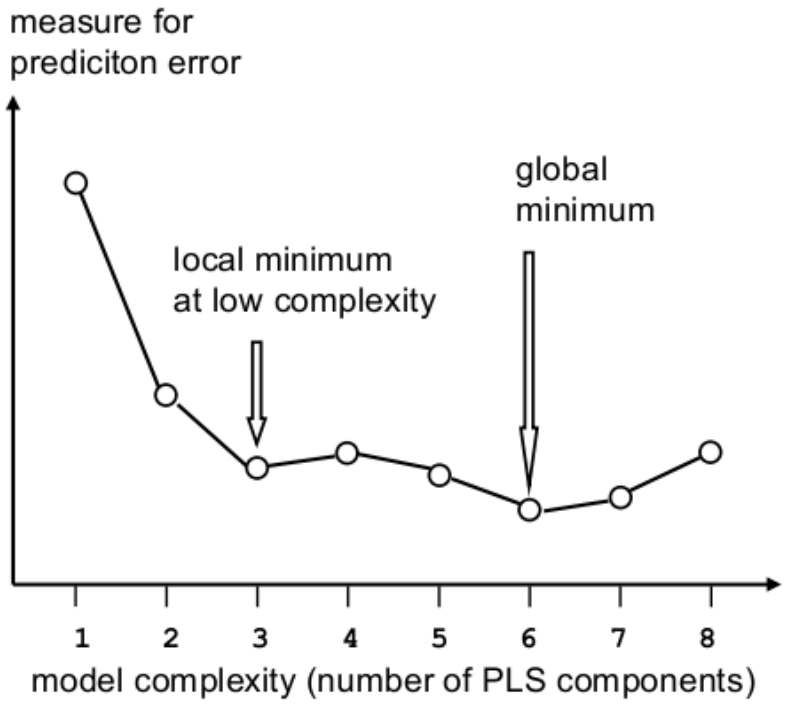
Peter.Filzmoser@tuwien.ac.at

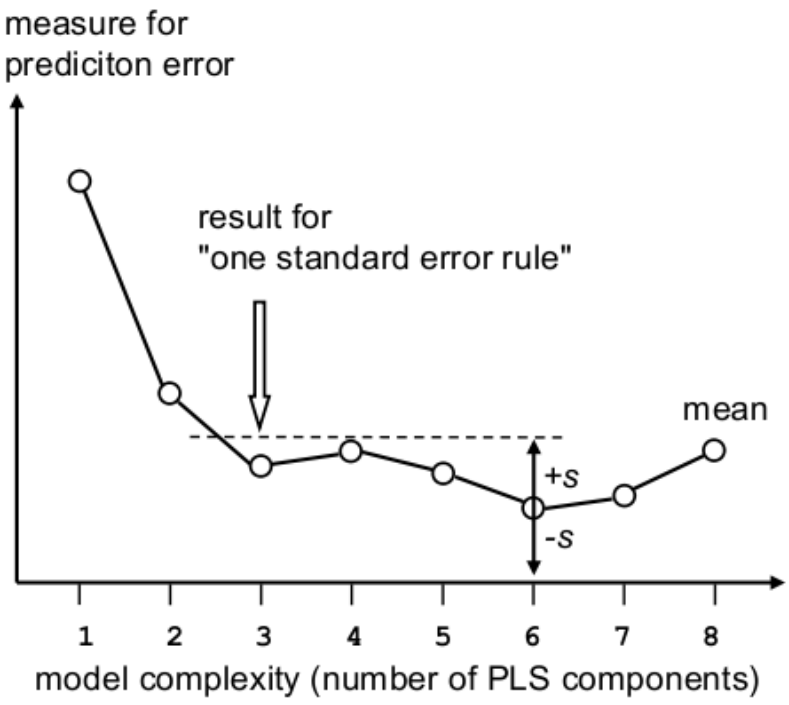
Optimal model complexity



Peter.Filzmoser@tuwien.ac.at

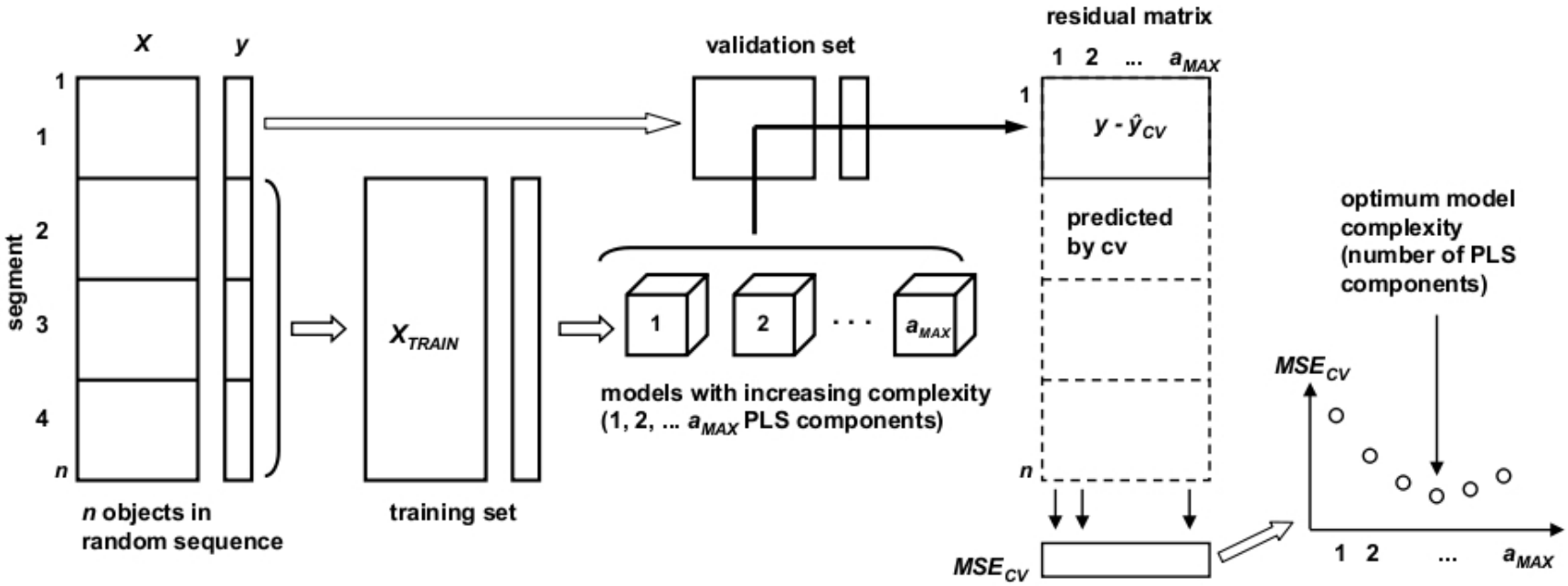
Optimal model complexity





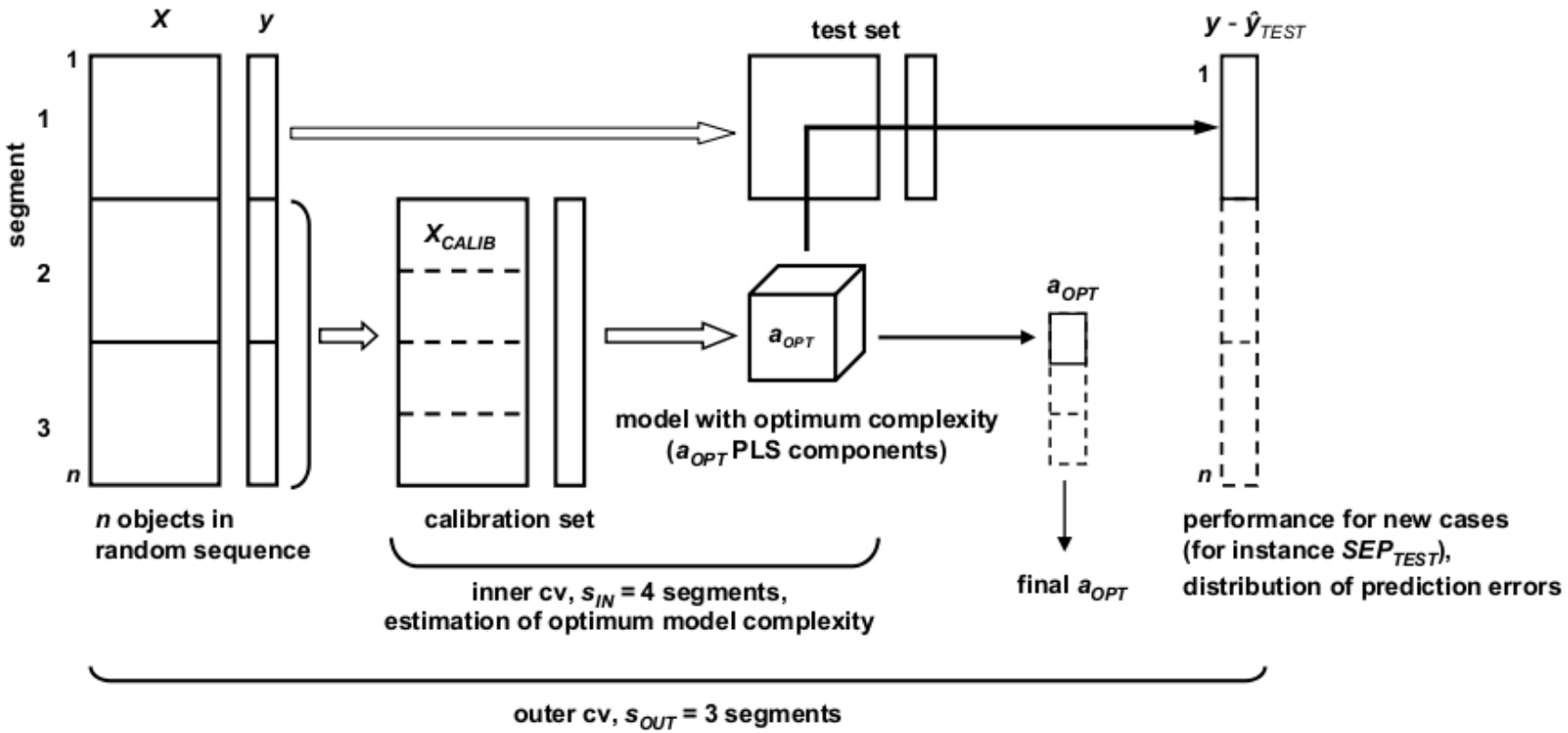
Peter.Filzmoser@tuwien.ac.at

Cross Validation



Peter.Filzmoser@tuwien.ac.at

Repeated Double Cross Validation



Peter.Filzmoser@tuwien.ac.at

Regression & Shrinkage

Ridge regression (Hoerl and Kennard, 1970) is a solution to the problem of multicollinearity. The Ridge estimator is

$$\widehat{\boldsymbol{\beta}}_{\text{Ridge}} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

and it solves the problem

$$\widehat{\boldsymbol{\beta}}_{\text{Ridge}} = \underset{\boldsymbol{\beta}}{\text{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right\} \quad \text{with} \quad \sum_{j=1}^{p} \beta_j^2 \leq \text{const.}$$

The Ridge parameter λ **shrinks the regression coefficients** (regularization).

Peter.Filzmoser@tuwien.ac.at

Regression & Shrinkage

Lasso regression (Tibshirani, 1996) solves the problem

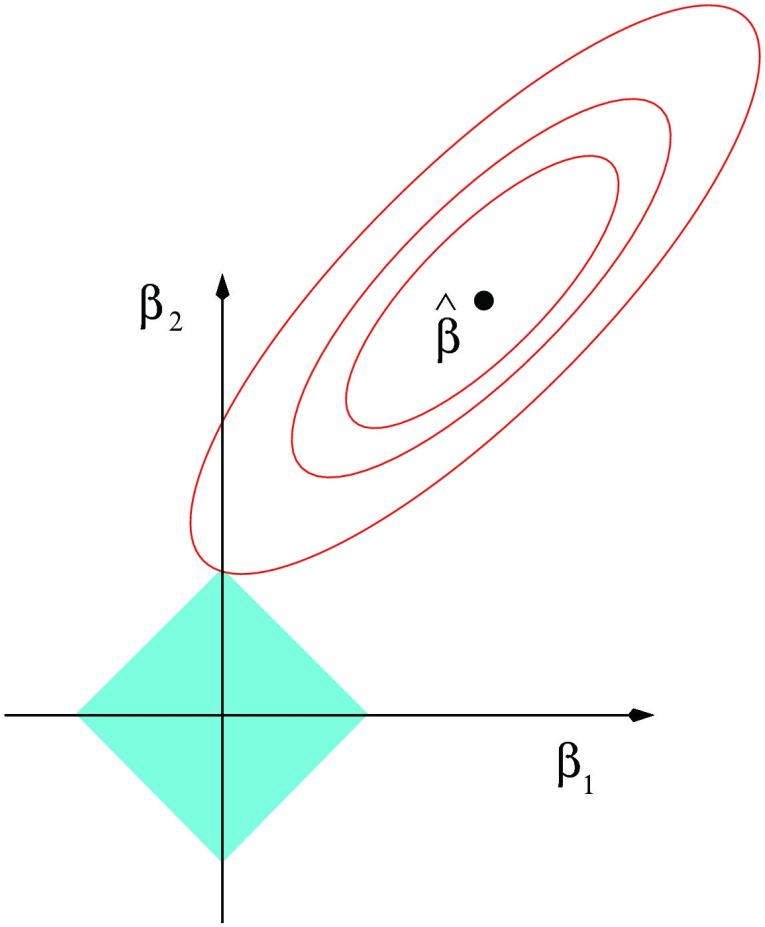
$$\hat{\boldsymbol{\beta}}_{\text{Lasso}} = \underset{\boldsymbol{\beta}}{\text{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right\} \quad \text{with} \quad \sum_{j=1}^{p} |\beta_j| \leq \text{const.}$$

No explicit formula for the solution, only numerical optimization.

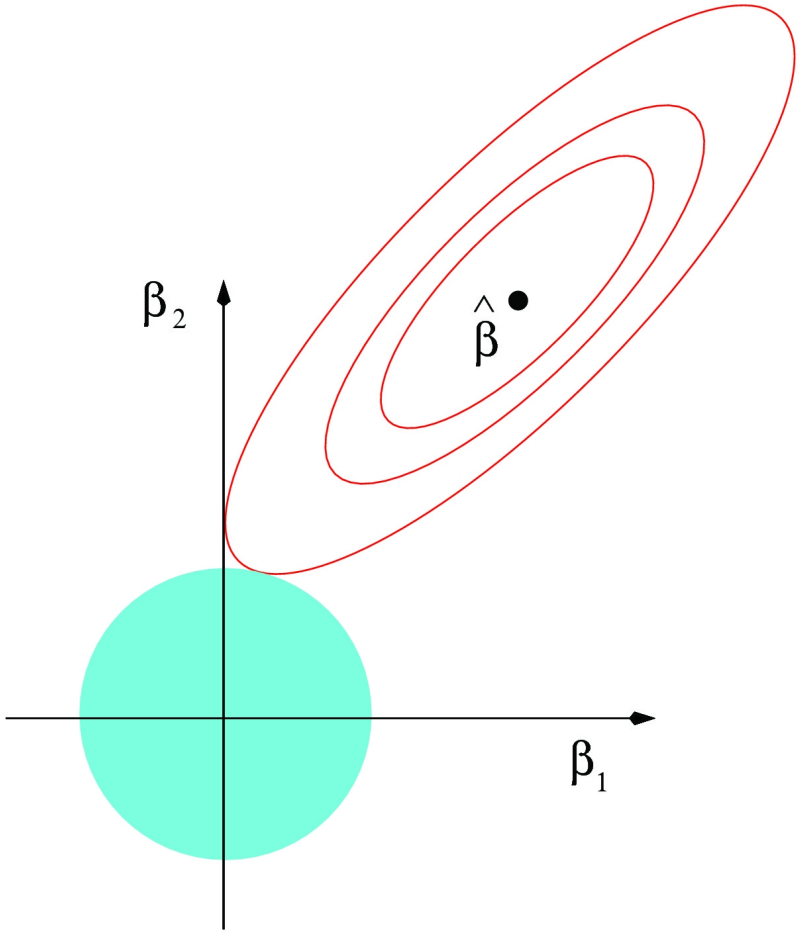
Peter.Filzmoser@tuwien.ac.at

Lasso versus Ridge

Lasso: $\sum_{j=1}^{p} |\beta_j| \leq \text{const}$



Ridge: $\sum_{j=1}^{p} \beta_j^2 \leq \text{const}$



Peter.Filzmoser@tuwien.ac.at

Software in R: package “chemometrics”

Freely available in R: `http://www.r-project.org`

Main contributions of the “chemometrics” package:

- Robustness (e.g. robust PLS)
- Model evaluation (e.g. repeated double cross-validation)
- Unified evaluation tools for parameter selection
- Diagnostic tools (e.g. for choice of number of components, visualizing effect of outliers)
- Example data sets

Peter.Filzmoser@tuwien.ac.at

Multivariate Calibration

Example data:

Gas chromatographic retention indices of polycyclic aromatic compounds:

We consider $n = 209$ polycyclic aromatic compounds (PAC):

y-vector: GC retention index;

X-matrix: $p = 467$ descriptors of the molecular structure (Corina, Dragon).

```
> library(chemometrics)
> data(PAC)
> str(PAC)

List of 2
 $ y: num [1:209] 197 197 197 200 201 ...
 $ X: num [1:209, 1:467] 6.51 6.51 6.01 7.12 8.95 6.62 7.32 7.6 7.6 6.77 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:209] "1" "2" "3" "4" ...
  .. ..$ : chr [1:467] "AMW" "Me" "Mp" "Ms" ...
```

PLS (and PCR)

Idea: reduce the number of regressor variables to a few components (PCR: using only the X-data; PLS: using both X and y data).

```
> pls_dcv <- mvr_dcv(y~X,ncomp=50,data=PAC,method="simpls")
             # PLS with repeated double cross validation
             # Default: 100 repetitions, 4 outer and 10 inner segments
             # for PCR use: method="svdpc"
```

Output:

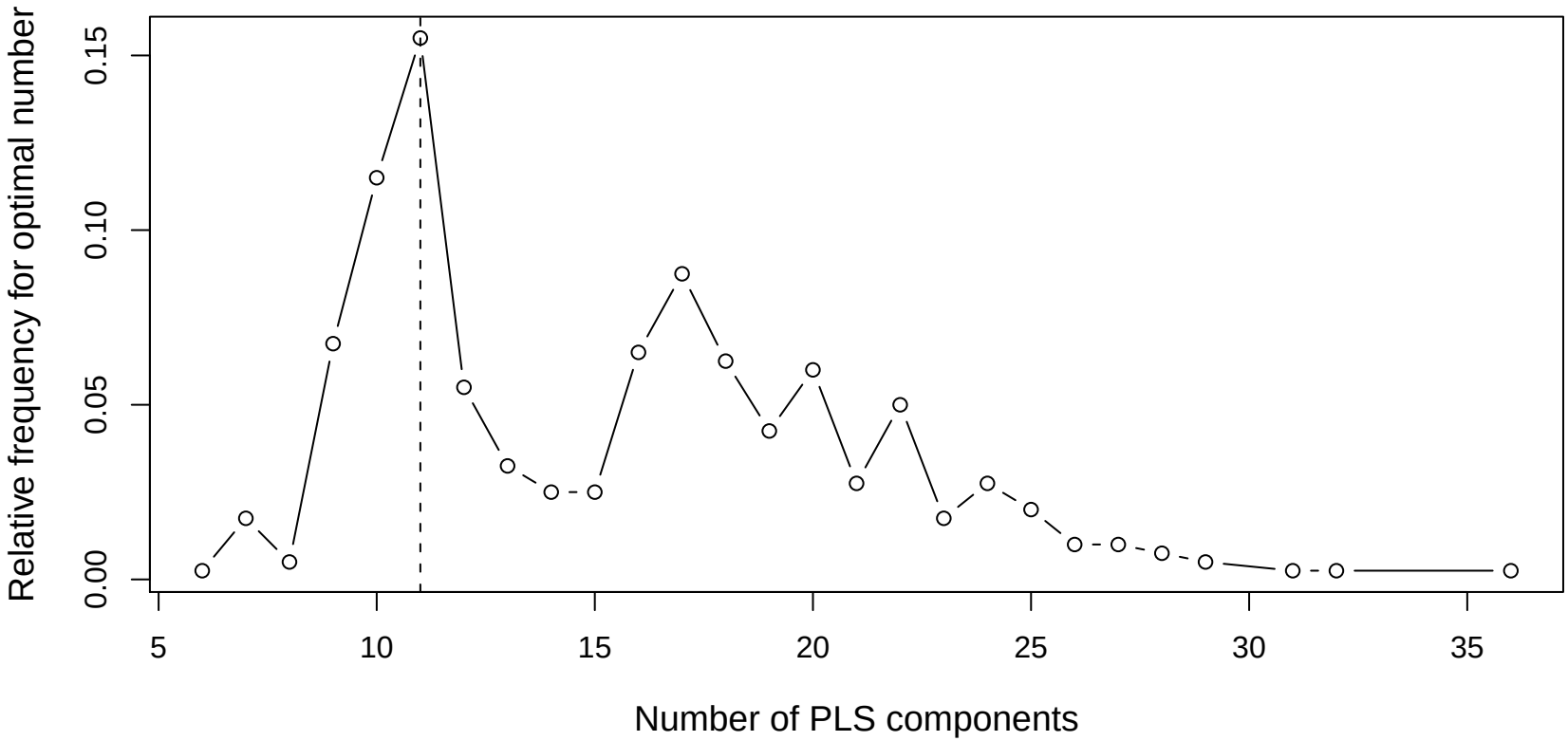
```
 $ resopt  : num [1:209, 1, 1:100]
 $ predopt : num [1:209, 1, 1:100]
 $ optcomp : int [1:4, 1:100]
 $ pred    : num [1:209, 1, 1:50, 1:100]
 $ SEPopt  : num 12
 $ sIQRopt : num 8.4
 $ sMADopt : num 8.39
 $ MSEPopt : num 144
 $ afinal  : num 11
 $ SEPfinal: Named num [1:50]
```

Peter.Filzmoser@tuwien.ac.at

PLS (and PCR)

Diagnostic plots: optimal number of components (based on 4×100 values)

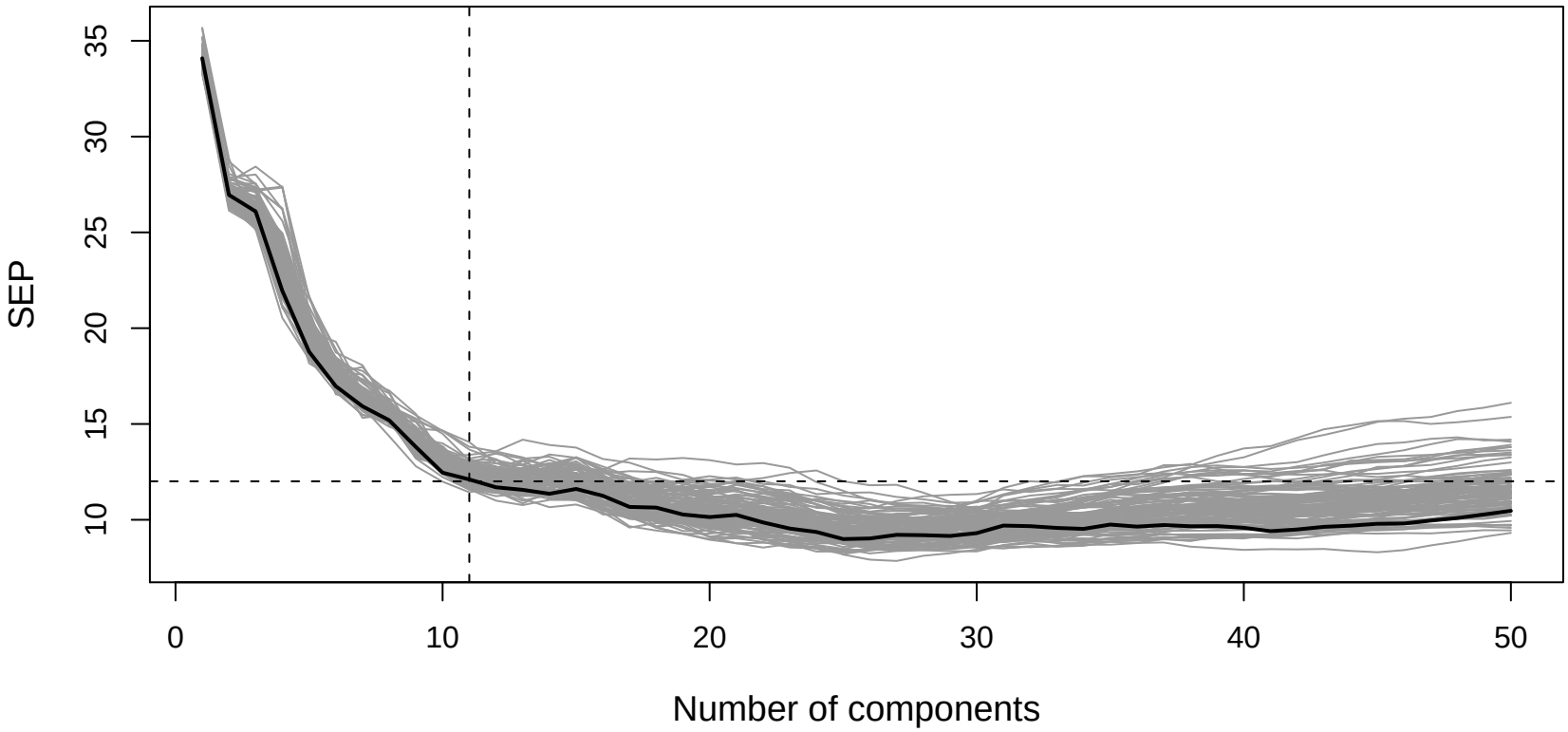
```
> plotcompmvr(pls_dcv)
```



PLS (and PCR)

Diagnostic plots: SEP for $1, \ldots, 50$ components

```
> plotSEPmvr(pls_dcv,optcomp=11,y=PAC$y,X=PAC$X,method="simpls")
```

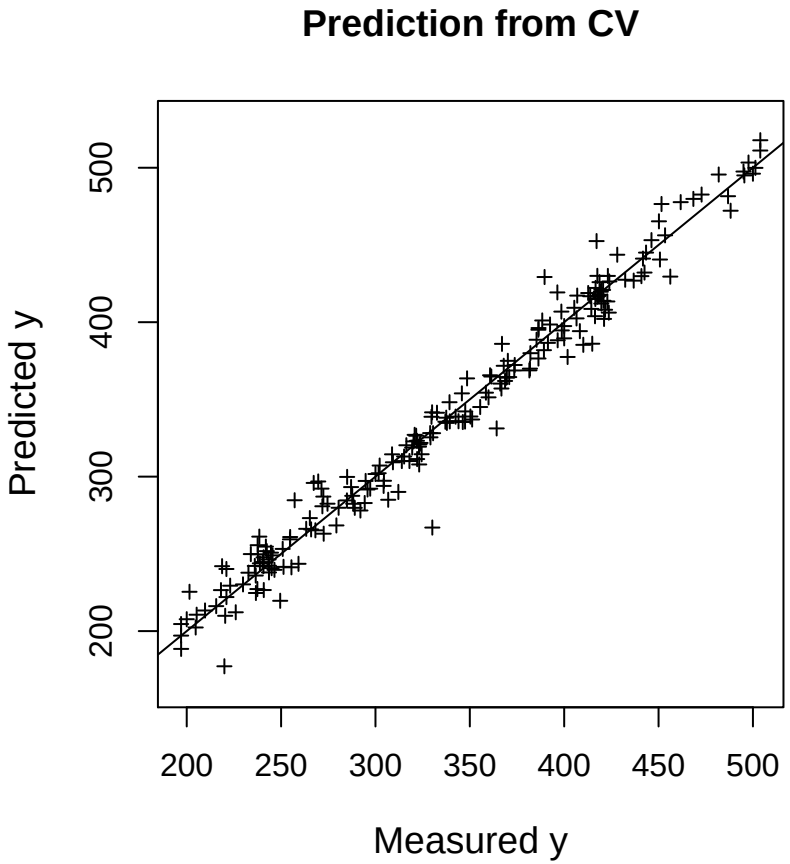


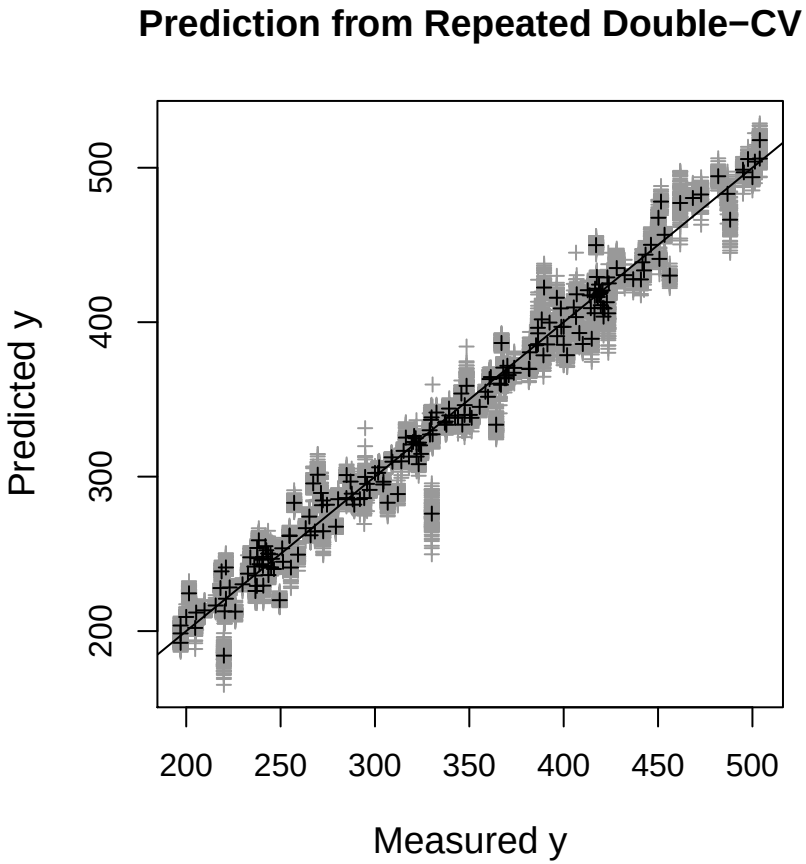
Peter.Filzmoser@tuwien.ac.at

PLS (and PCR)

Diagnostic plots: predicted values from 100 models with 11 components

```
> plotpredmvr(pls_dcv,optcomp=11,y=PAC$y,X=PAC$X,method="simpls")
```



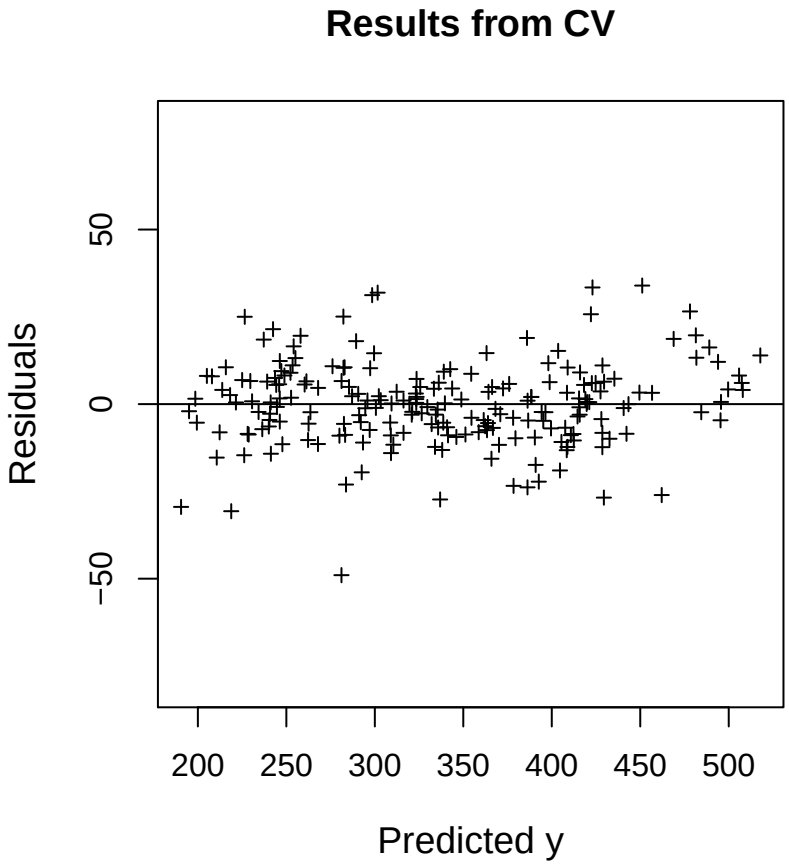


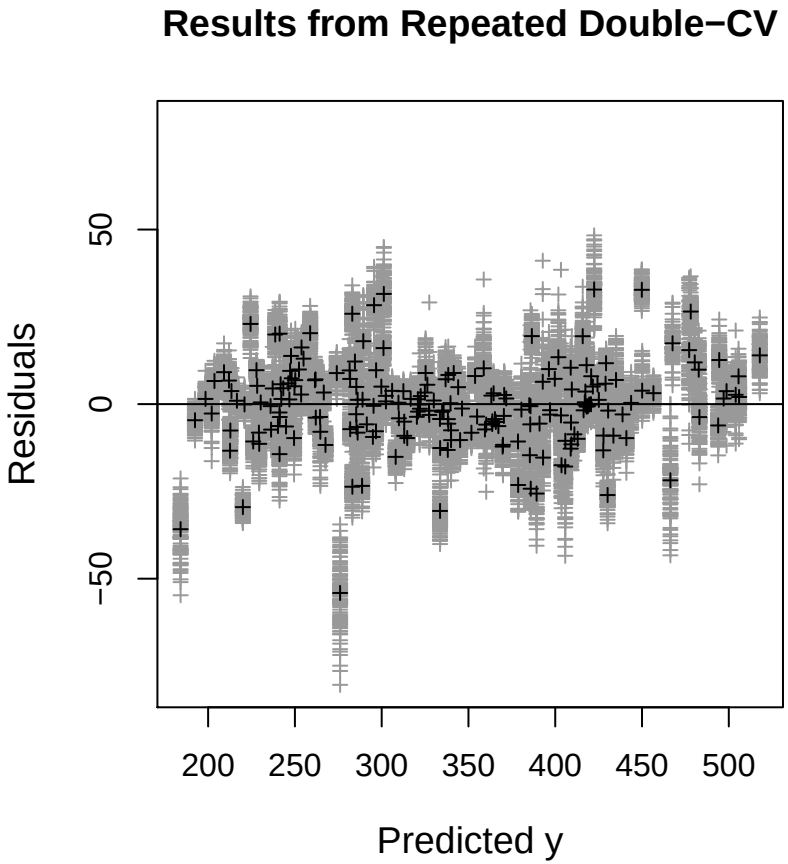
Peter.Filzmoser@tuwien.ac.at

PLS (and PCR)

Diagnostic plots: residuals from 100 models with 11 components

```
> plotresmvr(pls_dcv,optcomp=11,y=PAC$y,X=PAC$X,method="simpls")
```



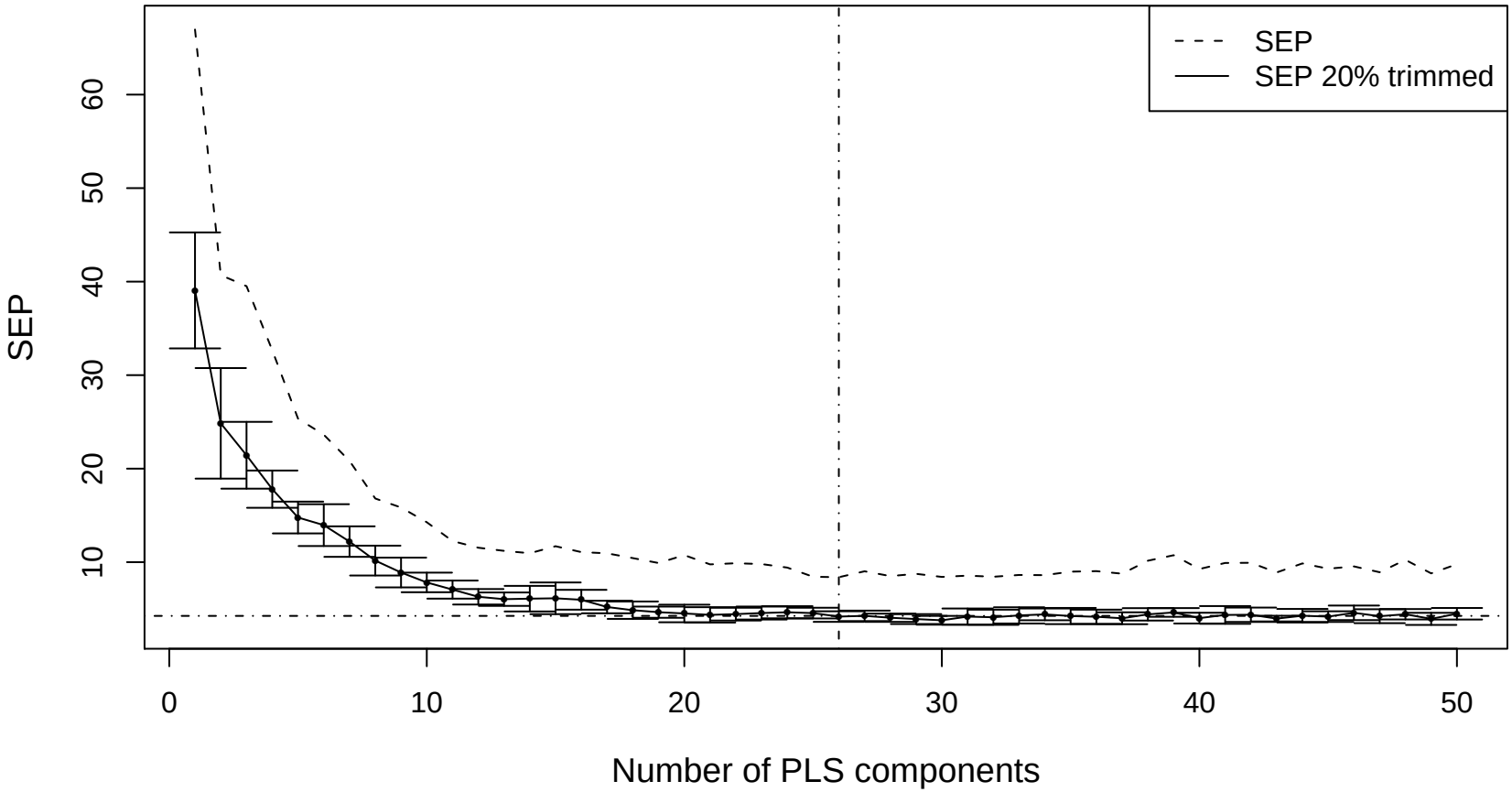


Peter.Filzmoser@tuwien.ac.at

Robust PLS

PRM: Serneels, Croux, Filzmoser, Van Espen (ChemoLab, 2005)

```
> prm_cv(PAC$X,PAC$y,a=50,trim=0.2,plot.opt=TRUE)
```

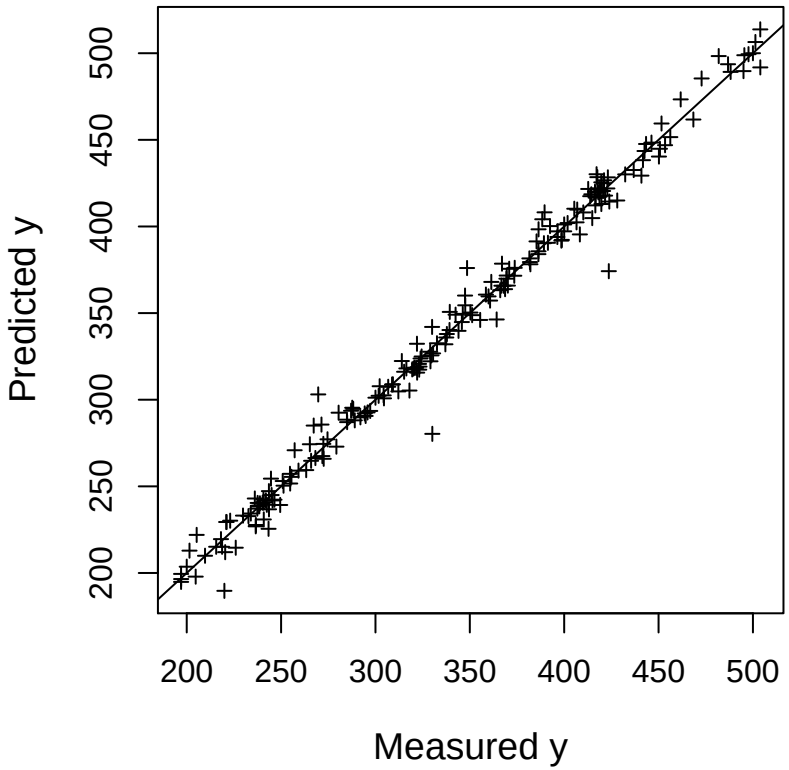


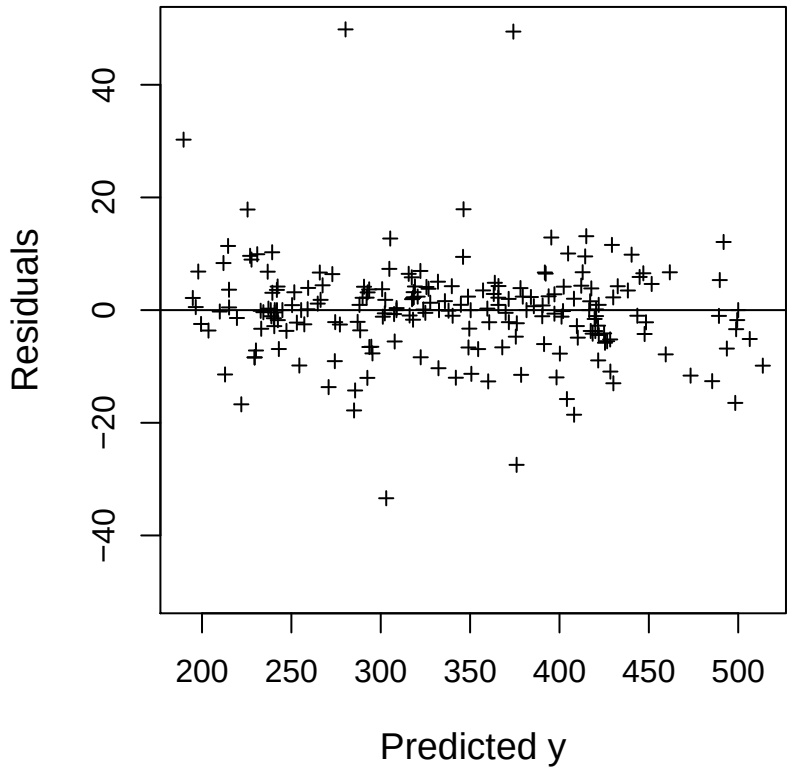
Peter.Filzmoser@tuwien.ac.at

Robust PLS

Diagnostic plots: predicted values and residuals using 26 components

```
> plotprm(resprmcv,PAC$y)
```



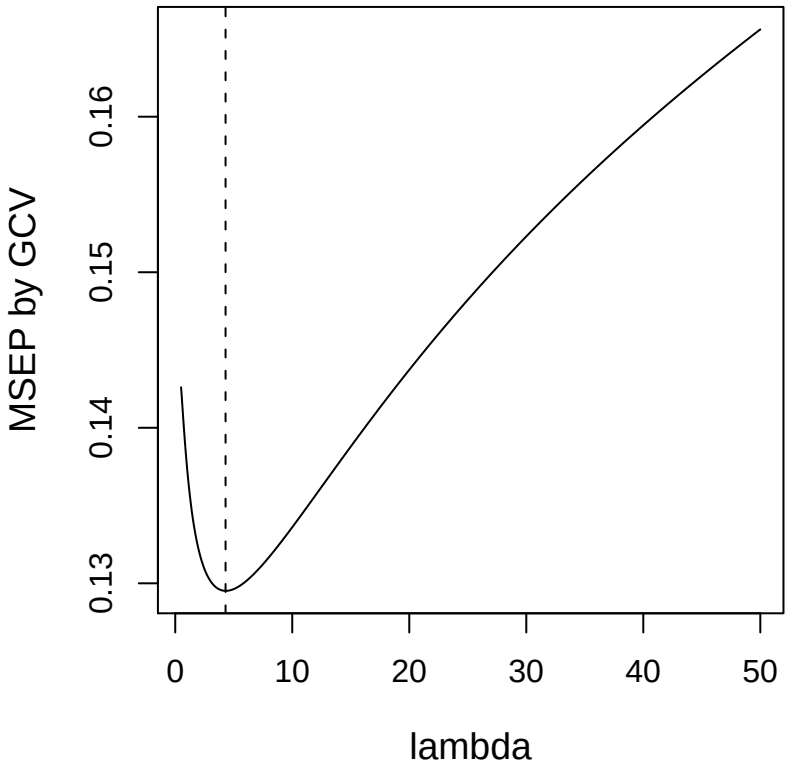


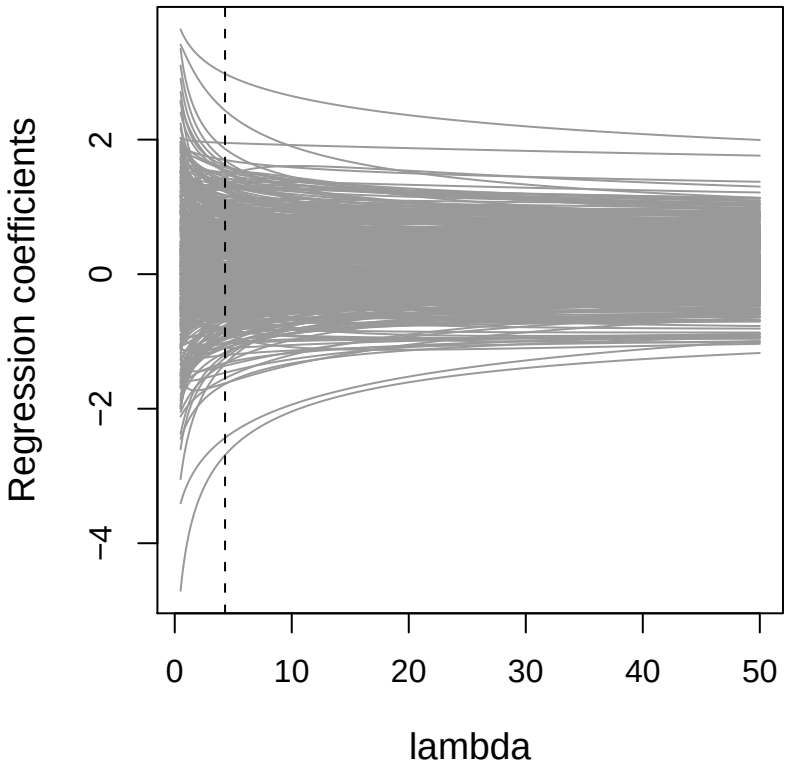
Peter.Filzmoser@tuwien.ac.at

Ridge Regression

Diagnostic plot: choice of ridge parameter

```
> resR <- plotRidge(y~X,data=PAC,lambda=seq(0.5,50,by=0.05))
```



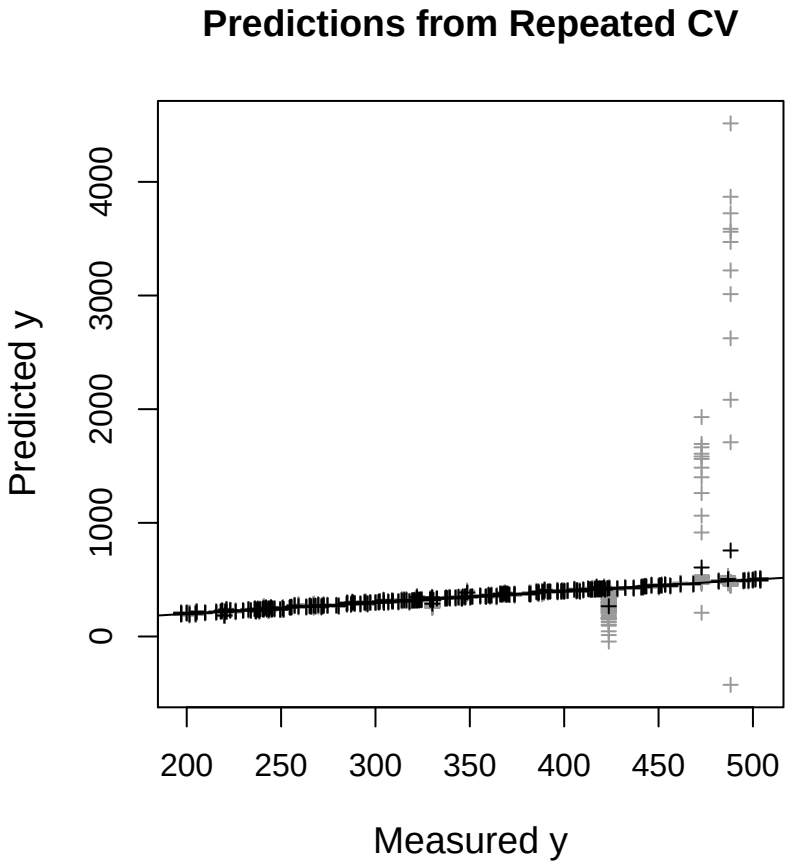


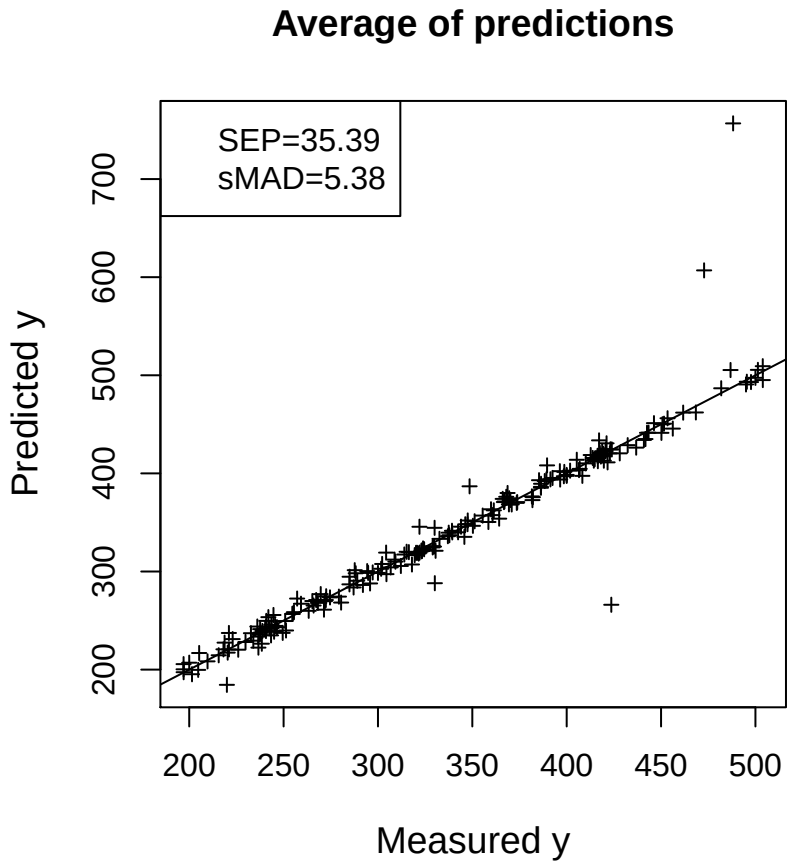
Peter.Filzmoser@tuwien.ac.at

Ridge Regression

Diagnostic plots: from repeated cross validation

```
> resRcv <- ridgeCV(y~X,data=PAC,repl=100,lambda=resR$lambdaopt)
```



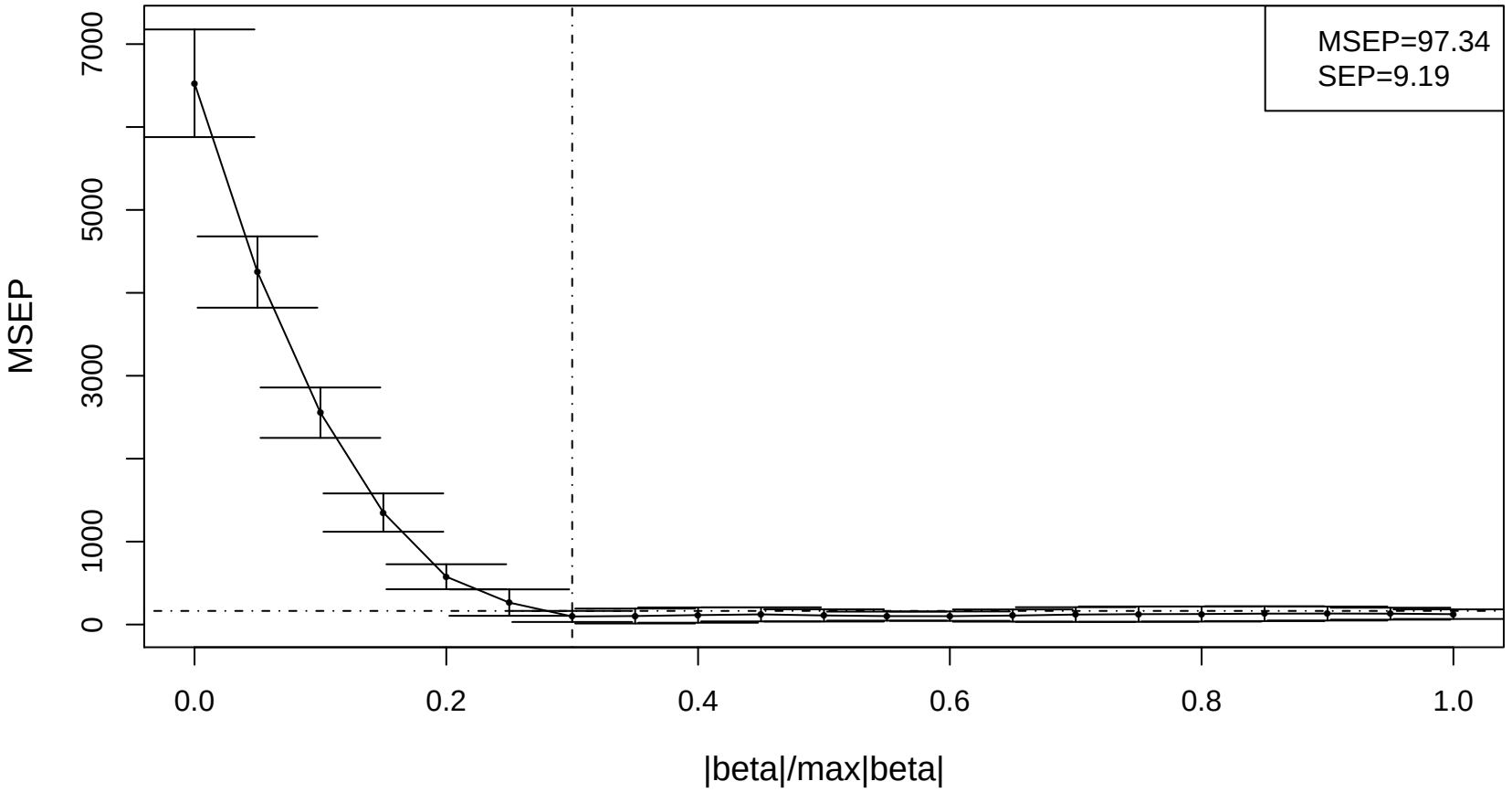


Peter.Filzmoser@tuwien.ac.at

Lasso Regression

Diagnostic plot: choice of Lasso parameter

```
> resL <- lassoCV(y~X,data=PAC,K=10,fraction=seq(0,1,by=0.05))
```

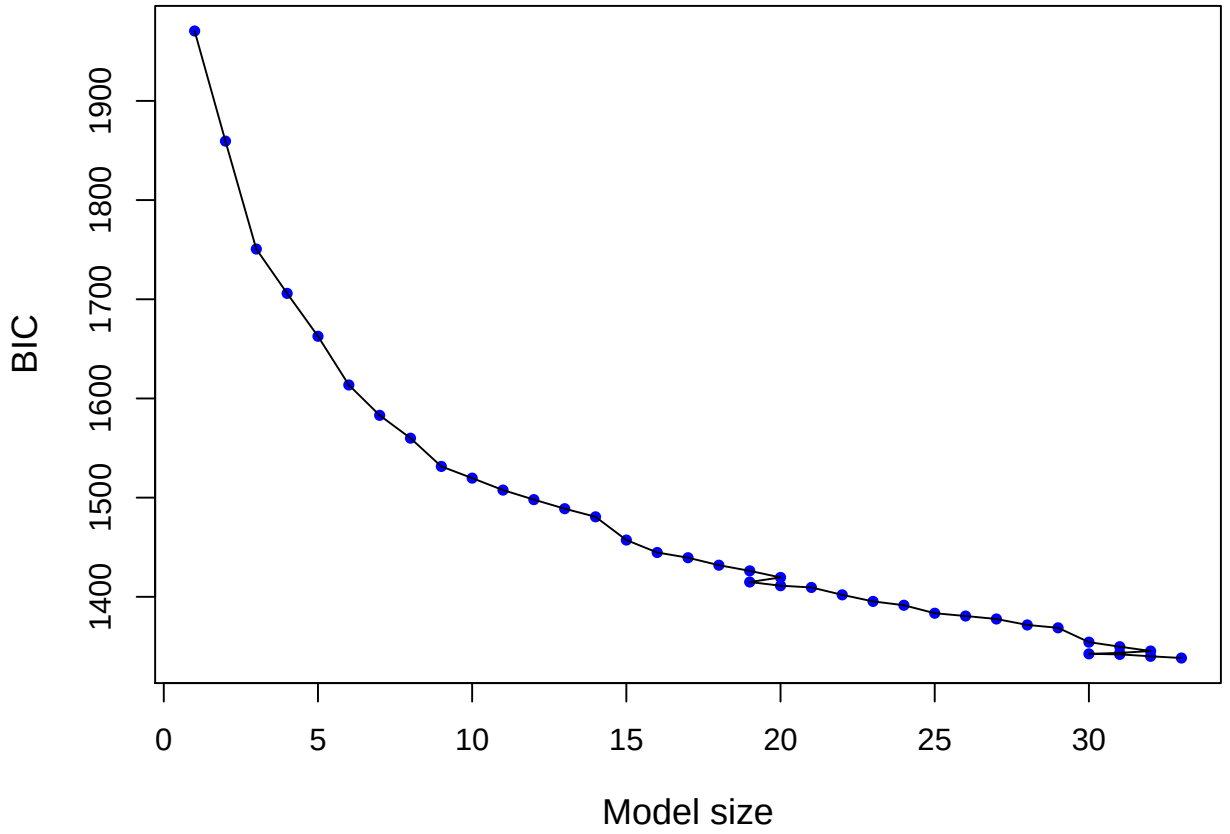


Peter.Filzmoser@tuwien.ac.at

Variable Selection by Stepwise Regression

Diagnostic plot: choice of model

```
> resS <- stepwise(y~X,data=PAC)
```

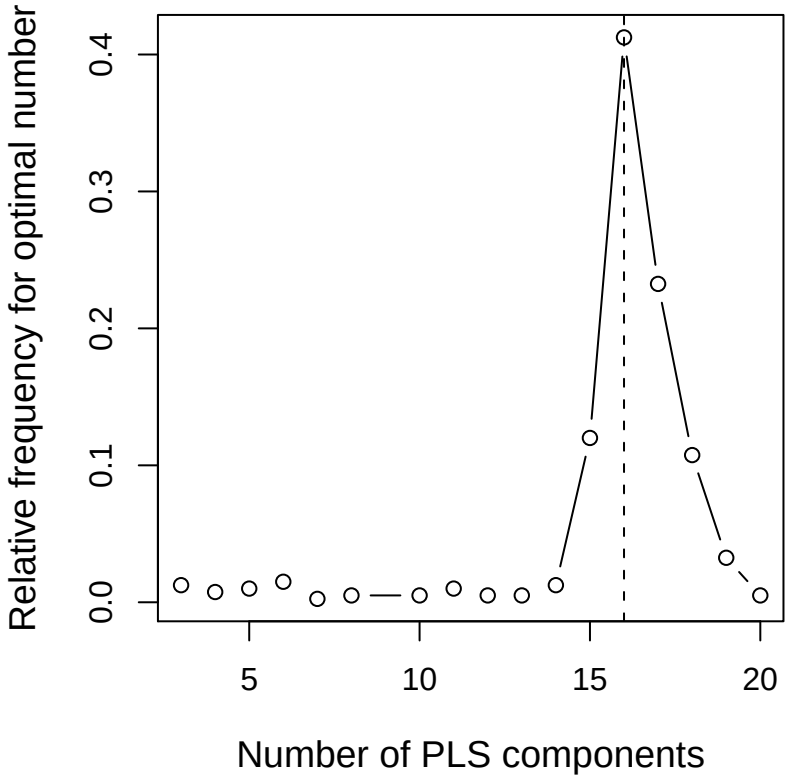


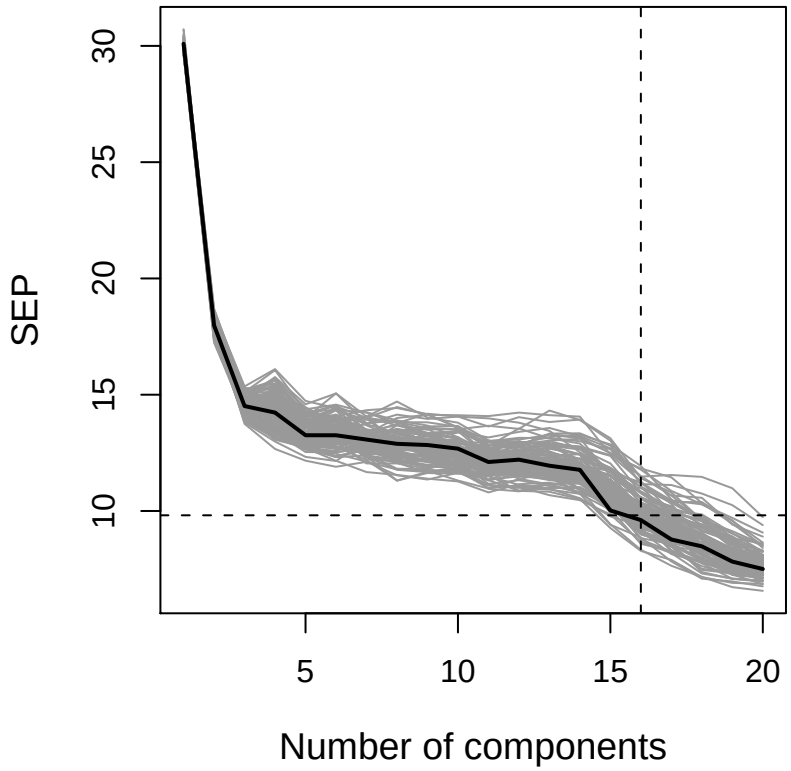
Peter.Filzmoser@tuwien.ac.at

Stepwise Regression + PLS

Diagnostic plot: choice of number of PLS components

```
> resSdcv <- mvr_dcv(y~.,ncomp=20,data=PACred,method="simpls")
```





Peter.Filzmoser@tuwien.ac.at

Comparison of Results

Method	p^*	k	SEP_{Test}	SEP_{CV}	$SEP^{0.2}$
PCR	467	21	14.2	—	7.9
PLS	467	11	12.0	—	5.7
Robust PLS	467	26	—	8.9	4.0
Ridge regression	467	—	—	28.4	4.0
Lasso regression	145	—	—	7.7	5.0
Stepwise variable selection + PLS	33	16	9.6	—	4.4

Peter.Filzmoser@tuwien.ac.at

Example data: Origin of glass samples:

- $n = 214$ glass samples
- 6 different glass types (e.g. windows, headlamps, tableware, containers)
- $p = 9$ variables (refractive index, mass-% of Al, Ba, Ca, Fe, K, Mg, Na, Si)

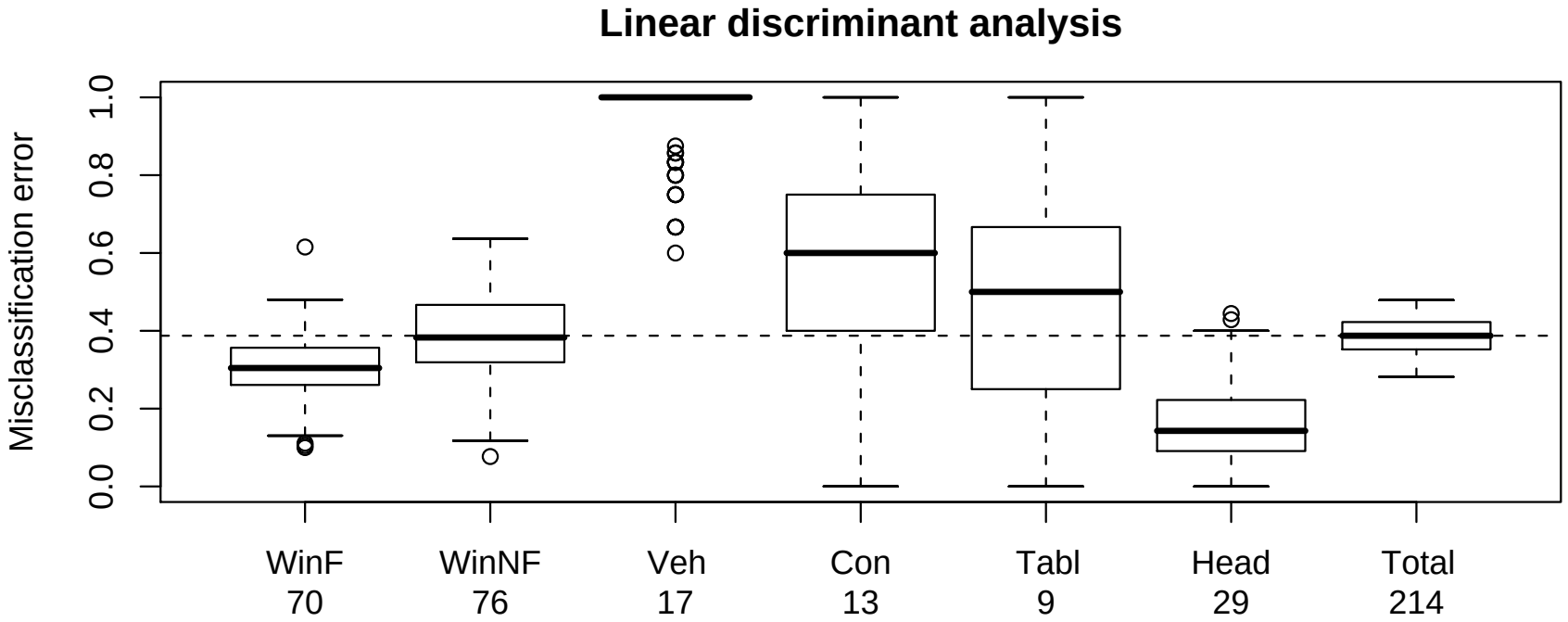
```
> library(MASS)
> data(fgl)
> grp=fgl$type
> X <- scale(fgl[,1:9])
> dim(X)

[1] 214   9
```

LDA (Linear Discriminant Analysis)

LDA: obtain LDA-rule for training data, apply to test data; repeat

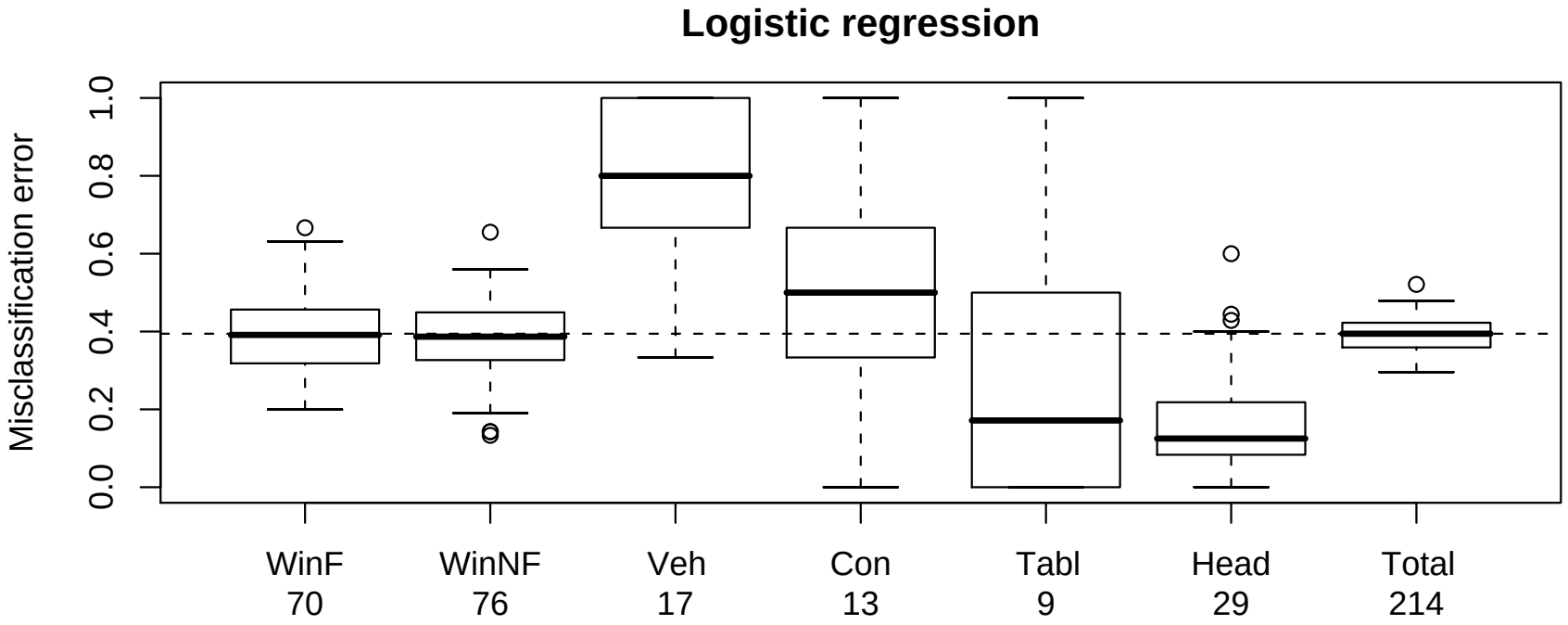
```
> train <- sample(1:n,ntrain)
> reslda <- lda(X[train,],grp[train])
> pred <- predict(reslda,newdata=X[-train,])$class
> tab <- table(grp[-train],pred)
```



LR (Logistic Regression)

LR: obtain LR-rule for training data, apply to test data; repeat

```
> train <- sample(1:n,ntrain)
> reslr <- vglm(grp ~ .,data=dat[train,],family=multinomial)
> predmix <- predict(reslr,dat[-train,],type="response")
> predgrp <- apply(predmix,1,which.max)
> tab <- table(grp[-train],predgrp)
```

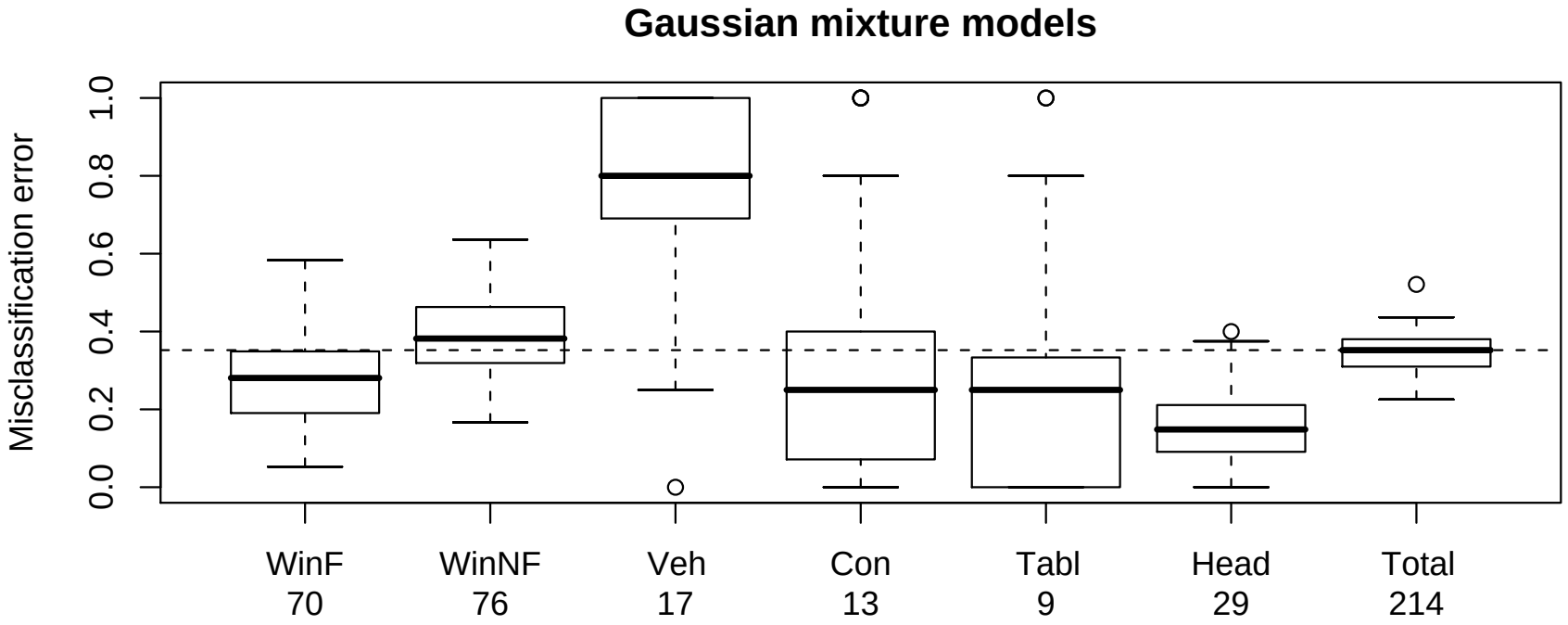


Peter.Filzmoser@tuwien.ac.at

Mix: Gaussian Mixture Models

Mix: obtain models for training data, apply to test data; repeat

```
> train <- sample(1:n,ntrain)
> resgmm <- mda(grp ~ .,data=dat[train,])
> predgmm <- predict(resgmm,dat[-train,],type="post")
> predgrp <- apply(predgmm,1,which.max)
> tab <- table(grp[-train],predgrp)
```

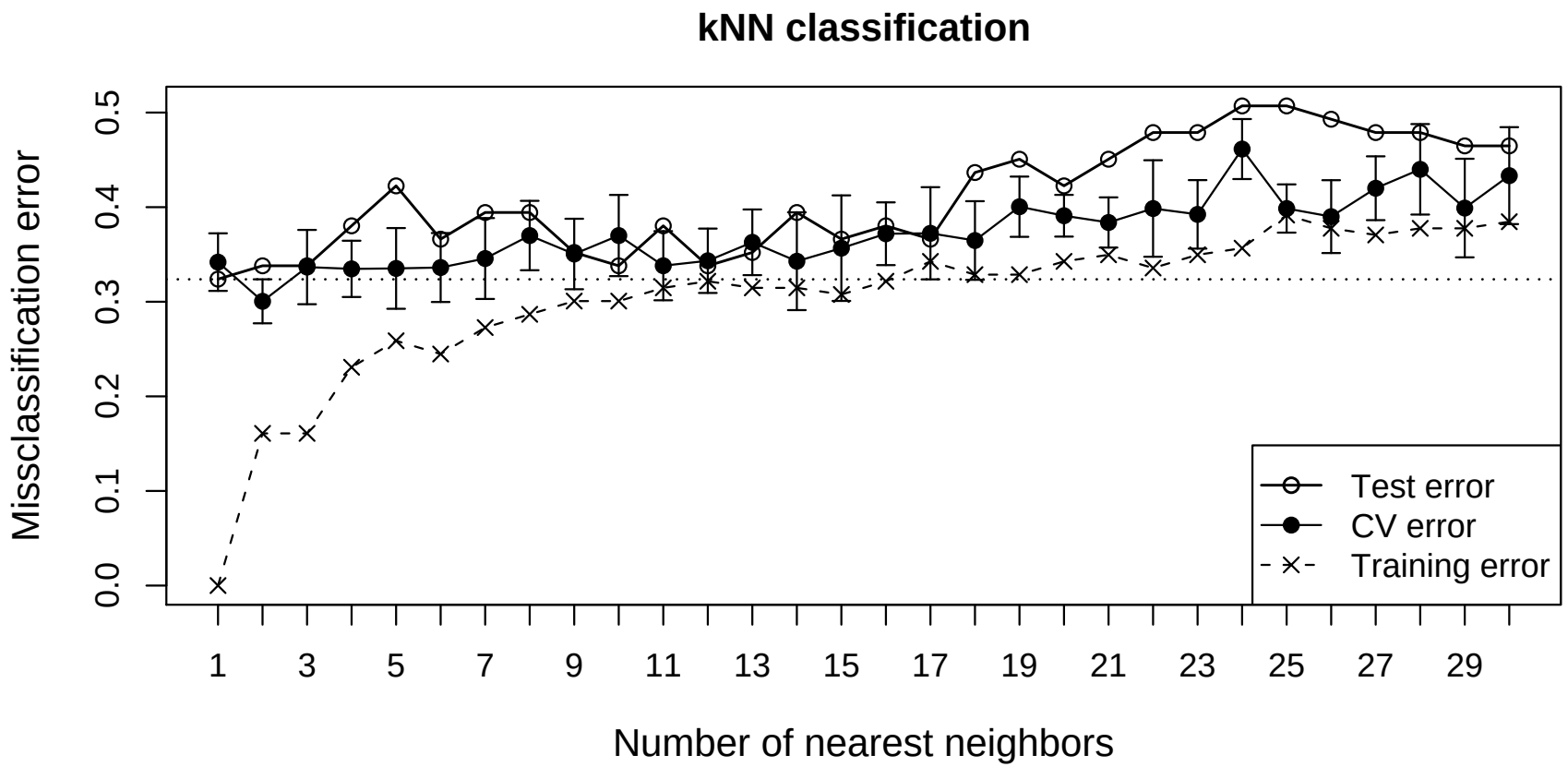


Peter.Filzmoser@tuwien.ac.at

kNN: k-nearest-neighbor classification

kNN: select tuning parameter “k” (number of neighbors)

```
> train <- sample(1:n,ntrain)
> resknn <- knnEval(X,grp,train,knnvec=seq(1,30,by=1))
```

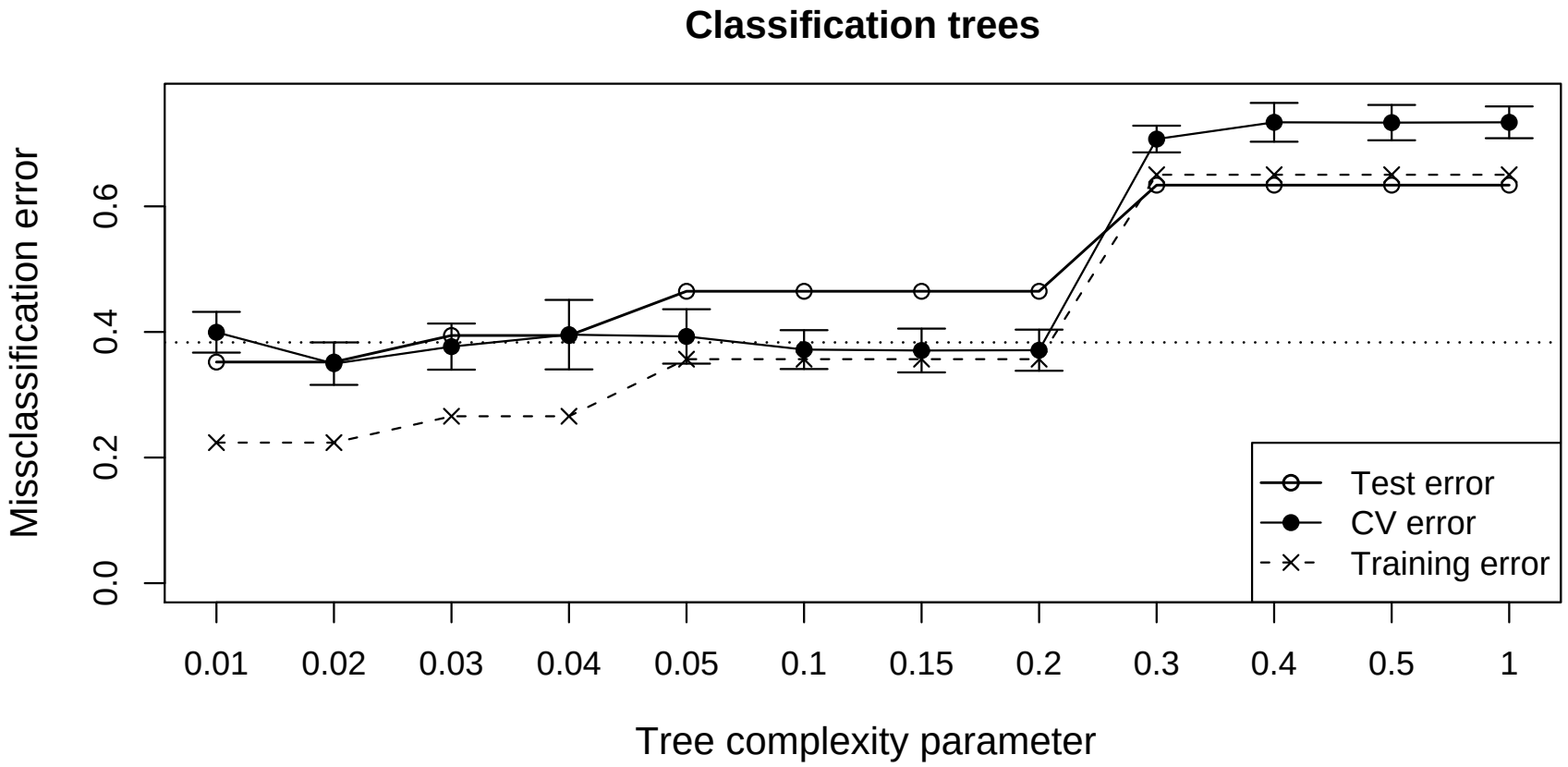


Peter.Filzmoser@tuwien.ac.at

Tree: classification trees

Tree: select tuning parameter “cp” (tree complexity)

```
> train <- sample(1:n,ntrain)
> cptry <- c(0.01,0.02,0.03,0.04,0.05,0.1,0.15,0.2,0.3,0.4,0.5,1)
> restree <- treeEval(X,grp,train,cp=cptry)
```

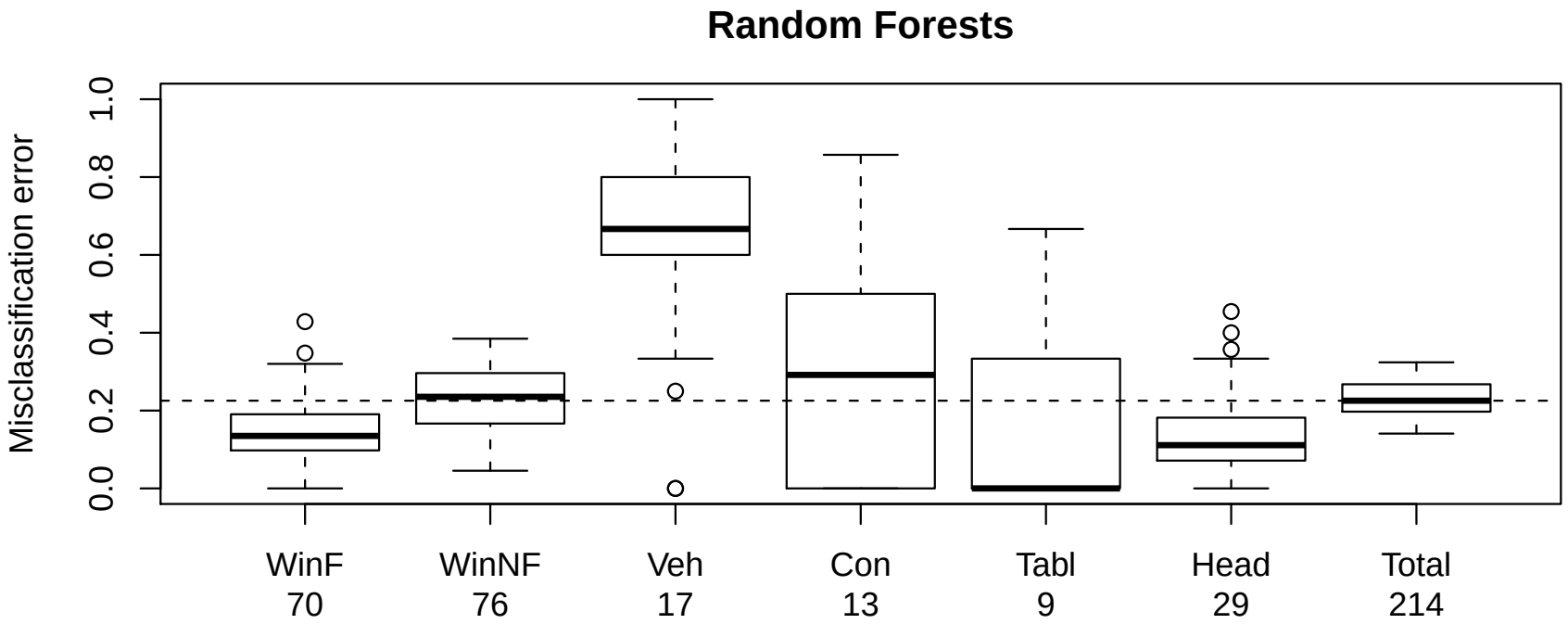


Peter.Filzmoser@tuwien.ac.at

RF: Random Forests

RF: obtain rule for training data, apply to test data; repeat

```
> train <- sample(1:n,ntrain)
> resRF <- randomForest(grp~.,data=dat,subset=train)
> predRF <- predict(resRF, dat[-train,])
> table(grp[-train],predRF)
```

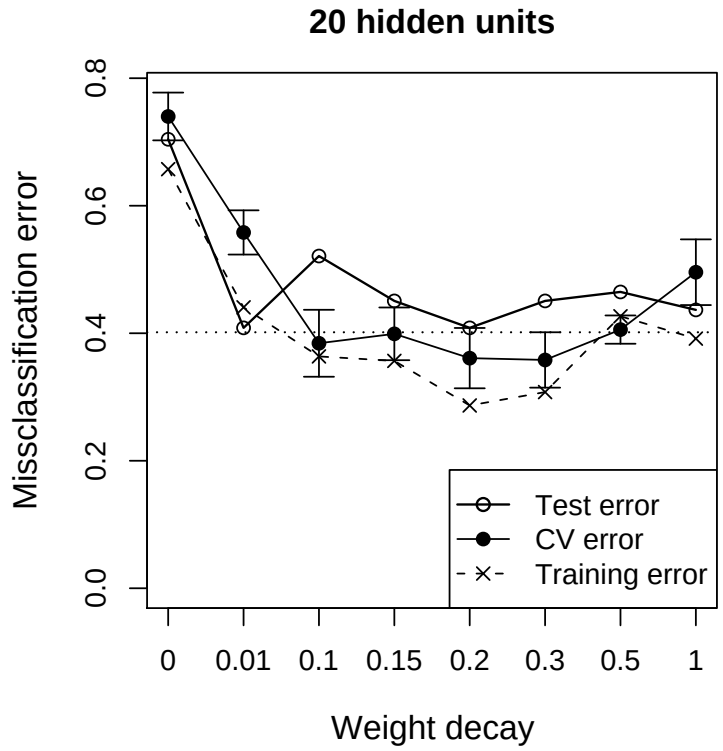


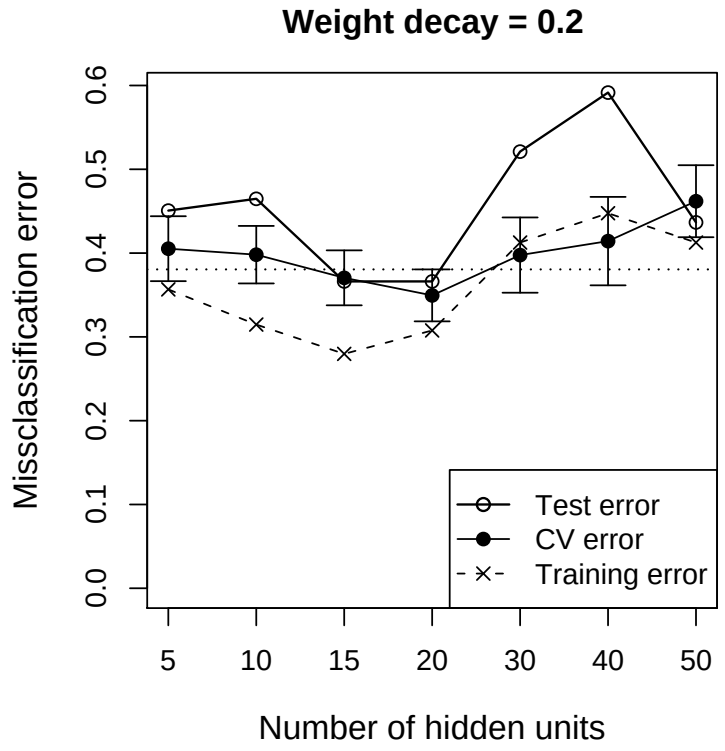
Peter.Filzmoser@tuwien.ac.at

ANN: Artifician Neural Networks

ANN: select tuning parameters “weight decay” and “number of hidden units”

```
> train <- sample(1:n,ntrain)
> wd <- c(0,0.01,0.1,0.15,0.2,0.3,0.5,1)
> sz <- c(5,10,15,20,30,40,50)
> resnnet=nnetEval(X,grp,train,decay=wd,size=20)
> resnnet=nnetEval(X,grp,train,decay=0.2,size=sz)
```



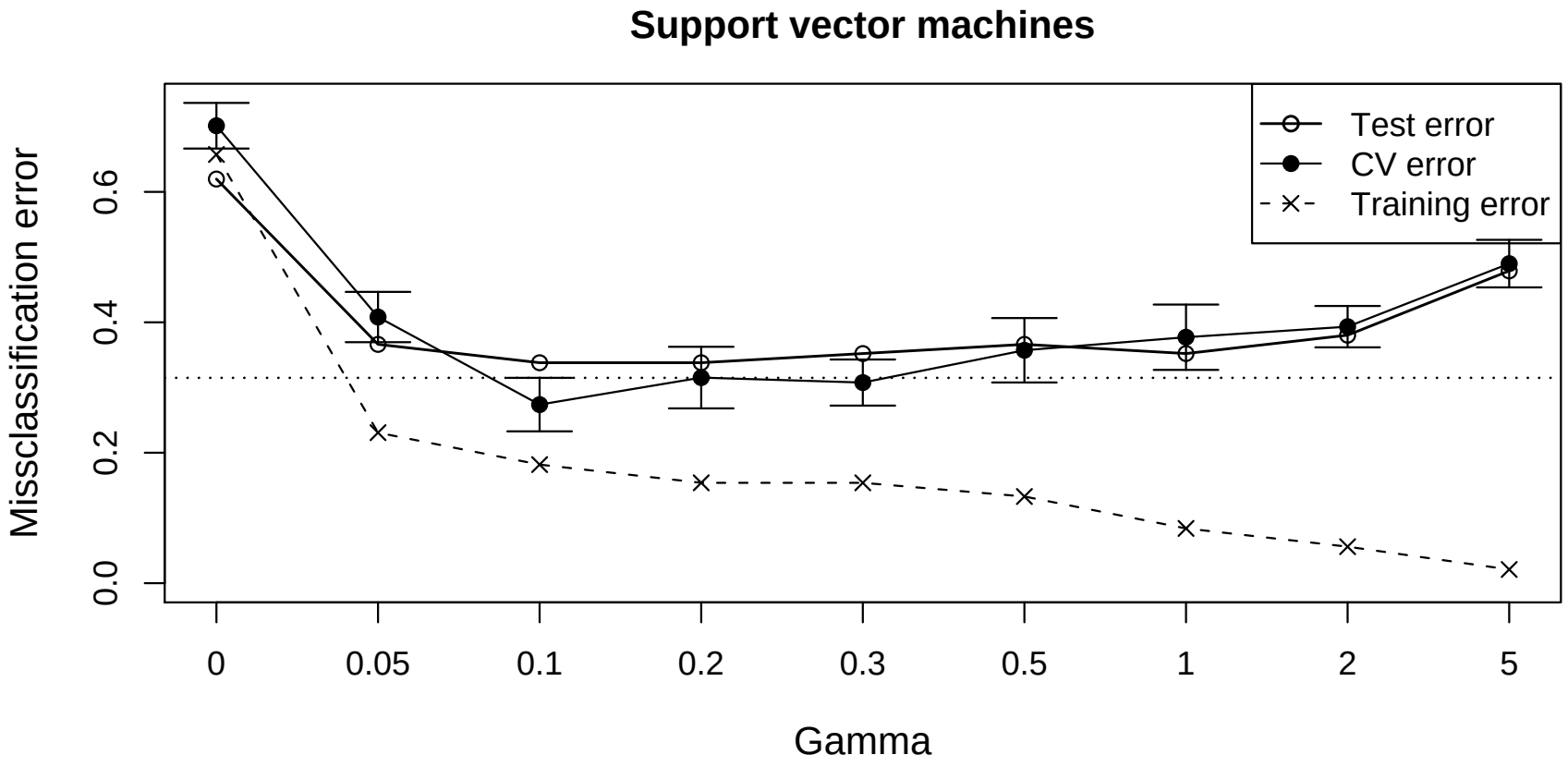


Peter.Filzmoser@tuwien.ac.at

SVM: Support Vector Machines

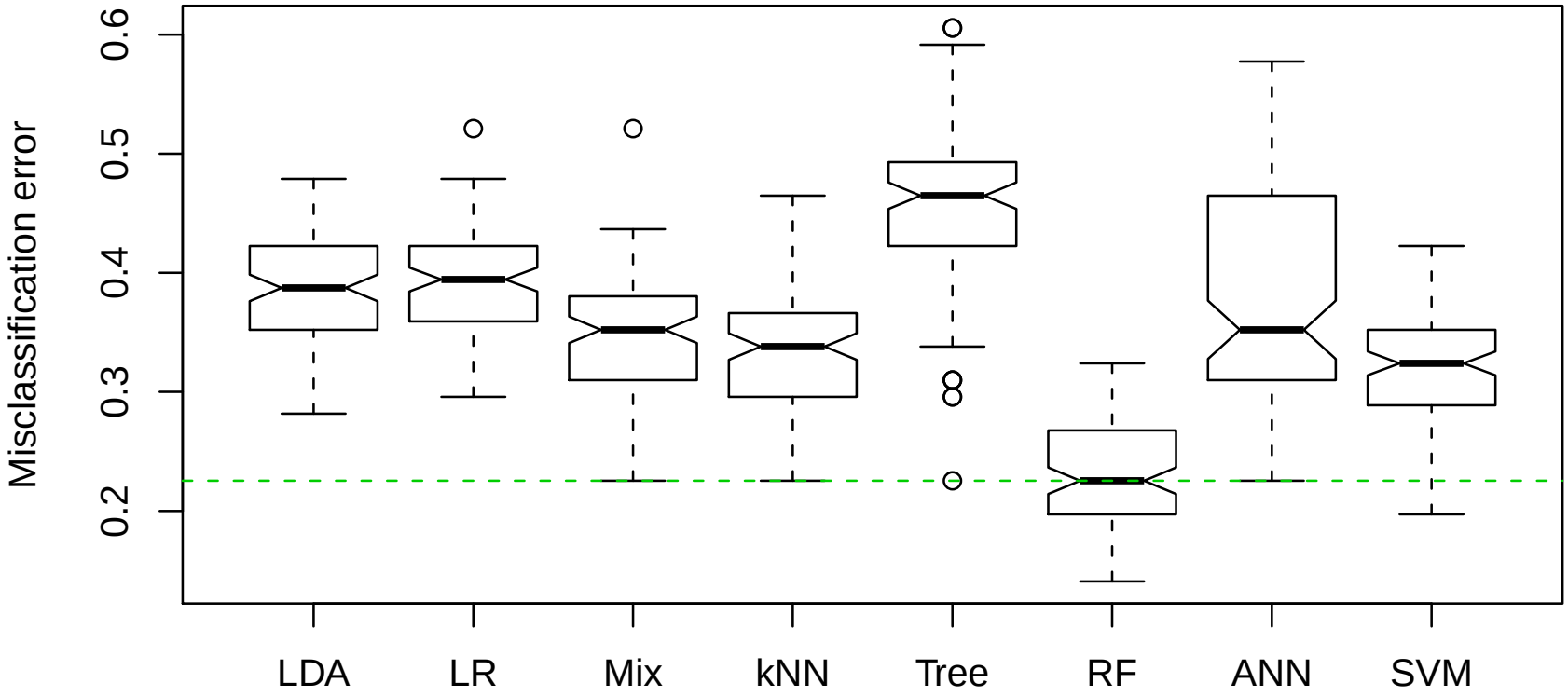
SVM: select tuning parameter "γ" (size constraint of slack variables)

```
> train <- sample(1:n,ntrain)
> gv <- c(0,0.05,0.1,0.2,0.3,0.5,1,2,5)
> ressvm <- svmEval(X,grp,train,gamvec=gv)
```



Peter.Filzmoser@tuwien.ac.at

Overall Comparison



Peter.Filzmoser@tuwien.ac.at

Conclusions

- The book is not only suitable for chemometricians, but for all who want to learn about **multivariate statistical methods** from a theoretical and practical perspective.

- Data sets and methods treated in the book are included in the R package **chemometrics**. The different evaluation procedures (*repeated cross validation*) are implemented in a unified manner.

- We already received a lot of positive feedback to both, the book and the R package.

Peter.Filzmoser@tuwien.ac.at