Davies, Laurie

Working Paper

Lasso, knockoff and Gaussian covariates: a comparison

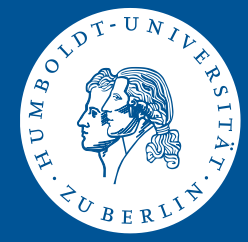
IRTG 1792 Discussion Paper, No. 2018-019

Provided in Cooperation with:
Humboldt University Berlin, International Research Training Group 1792 "High Dimensional Nonstationary Time Series"

Suggested Citation: Davies, Laurie (2018) : Lasso, knockoff and Gaussian covariates: a comparison, IRTG 1792 Discussion Paper, No. 2018-019, Humboldt-Universität zu Berlin, International Research Training Group 1792 "High Dimensional Nonstationary Time Series", Berlin

This Version is available at:
https://hdl.handle.net/10419/230730

IRTG 1792 Discussion Paper 2018-019



International Research Training Group 1792

Lasso, knockoff and Gaussian covariates: a comparison

Laurie Davies *

* University Duisburg-Essen, Germany

This research was supported by the Deutsche Forschungsgemeinschaft through the International Research Training Group 1792 "High Dimensional Nonstationary Time Series".

http://irtg1792.hu-berlin.de
ISSN 2568-5619

Lasso, knockoff and Gaussian covariates: a comparison

Laurie Davies[1]

Abstract

Given data $\boldsymbol{y}$ and k covariates $\boldsymbol{x}_j$ one problem in linear regression is to decide which if any of the covariates to include when regressing the dependent variable $\boldsymbol{y}$ on the covariates $\boldsymbol{x}_j$. In this paper three such methods, lasso, knockoff and Gaussian covariates are compared using simulations and real data. The Gaussian covariate method is based on exact probabilities which are valid for all $\boldsymbol{y}$ and $\boldsymbol{x}_j$ making it model free. Moreover the probabilities agree with those based on the F-distribution for the standard linear model with i.i.d. Gaussian errors. It is conceptually, mathematically and algorithmically very simple, it is very fast and makes no use of simulations. It outperforms lasso and knockoff in all respects by a considerable margin.

1 Introduction

There are many papers on lasso from the first [Tibshirani, 1996] to a very recent one [Bellec et al., 2017], a span of 21 years. As no theoretical comparisons are made in this paper we give no further references. The software required for the comparison is the R package glmnet which can be downloaded from

`https://CRAN.R-project.org/package=glmnet`

Knockoff is much more recent. Theoretical work is to be found in [Candes et al., 2017]. The software is obtainable from R

[1]Faculty of Mathematics, University Duisburg-Essen, 45117 Essen, Federal Republic of Germany, email: laurie.davies@uni-due.de

https://cran.r-project.org/web/packages/knockoff/index.html

Part of the comparison is based on the Tutorials 1 and 2 of

https://web.stanford.edu/group/candes/knockoffs/software/knockoff/

The present version of the Gaussian covariate method is new but it is based on previous attempts, see [Davies, 2017]. It is described in Section 2. There is as yet no R package but the software is available as an ancillary file.
The real data used in the comparison includes three of the data sets used in [Dettling and Bühlmann, 2003], colon cancer ([Alon et al., 1999]), leukemia ([Golub et al., 1999]) and lymphoma ([Alizadeh et al., 2000]) available from

http://microarray.princeton. edu/oncology/
http://microarray.princeton. edu/oncology/
http://llmpp.nih.gov/lymphoma/data/figure1).

respectively. The prostate cancer data is available from the *lasso2* CRAN R package

https://CRAN.R-project.org/package=lasso2

and the red wine data ([Cortez et al., 2009]) from

https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/

The Boston housing data and the Brownlee stack loss data are available from the R package MASS

https://CRAN.R-project.org/package=MASS

A further data set is that considered in [Cox and Battey, 2017] on osteoarthritis available from the Gene Expression Omnibus under accession number GDS5363 available from

https://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS5363

As in [Cox and Battey, 2017] the males have been excluded. All the data sets are the same but with permutations so that replicating the results here will require exactly the same data set. The one used here is that used in [Cox and Battey, 2017] and is, I think, the *DataSet SOFT file*.
For a short discussion of L_1 regression use is made of the R package quantreg available from

https://CRAN.R-project.org/package=quantreg

The comparisons require the latest version of R [R Development Core Team, 2008]. The results of this paper can be reproduced by running the file runcomp.R. This requires the FORTRAN file selvar.f and the R files comp.R and selvar.R. The output is given in the Appendix 5. The running time was eight hours 40 minutes.

2 A description of the Gaussian covariate method

2.1 The basic idea

Consider data consisting of a dependent variable $\boldsymbol{y} = \boldsymbol{y}(n) = (y_1, \ldots, y_n)$ and an explanatory variable $\boldsymbol{x} = (x_1, \ldots, x_n)$. The problem is to decide if $\boldsymbol{x}$ is indeed an explanatory variable for $\boldsymbol{y}$ in the sense that the values of $\boldsymbol{y}$ can to some extent be explained by those of $\boldsymbol{x}$. A standard method of deciding this is to postulate a linear model

$$\boldsymbol{y} = \beta \boldsymbol{x} + \sigma \boldsymbol{\varepsilon} \tag{1}$$

with $\boldsymbol{\varepsilon}$ i.i.d. standard Gaussian noise and to test the null hypothesis $\beta = 0$. The standard test is the F-test based on the F-statistic

$$F = (ss_y - ss_{r,x})/(ss_{r,x}/(n-1)) \stackrel{D}{=} F_{1,n-1} \tag{2}$$

where $ss_y = \sum_{i=1}^{n} y_i^2$ and $ss_{r,x}$ is the sum of the squared residuals after regressing $\boldsymbol{y}$ on $\boldsymbol{x}$. The null hypothesis is rejected if the P-value

$$1 - F_{1,n-1}(F) \tag{3}$$

is less than the specified size of the test α.

The Gaussian covariate approach is to compare $\boldsymbol{x}$ with a Gaussian covariate $\boldsymbol{Z}$ with i.i.d. $N(0,1)$ components. The comparison is done through the sum of the squared residuals. The covariate $\boldsymbol{Z}$ is clearly not an explanatory covariate so that if $\boldsymbol{x}$ is no better than $\boldsymbol{Z}$ in respect of the sum of squared residuals it is concluded that $\boldsymbol{x}$ is also not an explanatory covariate: the null hypothesis $\beta = 0$ is replaced by the question, is $\boldsymbol{Z}$ better than $\boldsymbol{x}$?

Denote the sum of squared residuals after regressing $\boldsymbol{y}$ on $\boldsymbol{Z}$ by $ss_{r,Z}$. It has been shown by Lutz Dümbgen ([Davies and Dümbgen,]) that

$$B = 1 - ss_{r,Z}/ss_y \stackrel{D}{=} B_{1/2,(n-1)/2} \tag{4}$$

with P-value

$$1 - B_{1/2,(n-1)/2}(B) \tag{5}$$

and that the two P-values are equal

$$1 - B_{1/2,(n-1)/2}(B) = 1 - F_{1,n-1}(F). \tag{6}$$

This is a remarkable result even if it has a simple proof. It is remarkable because both P-values are exact and uniformly distributed over $[0,1]$ but whereas the P-value on the left is valid for all (non-zero) $\boldsymbol{y}$ and $\boldsymbol{x}$ that on the right depends on the model (1).

We need a generalization of this result also due to Lutz Dümbgen ([Davies and Dümbgen,]). Given $\boldsymbol{y}$ and ℓ linearly independent covariates $\boldsymbol{x}_j, j = 1, \ldots, \ell$ and $\ell' - \ell \geq 1$ i.i.d. $N(0,1)$ additional random covariates we have

$$1 - ss_{\ell'}/ss_{\ell} \stackrel{D}{=} B_{(\ell'-\ell)/2,(n-\ell')/2} \tag{7}$$

where ss_ℓ is the sum of squared residuals after regressing on the $\boldsymbol{x}_j, j = 1, \ldots, \ell$ and $ss_{\ell'}$ the sum of squared residuals after regressing on all ℓ' covariates. The case $\ell' = \ell + 1$ is the one required for stepwise regression.

2.2 Gaussian covariate stepwise regression

Regress $\boldsymbol{y}$ on $\boldsymbol{x}_j$ including an offset by default, put

$$ss_y = \sum_{i=1}^{n} (y_i - \text{mean}(\boldsymbol{y}))^2$$

and denote the sum of squared residuals by ss_j. The best of the $\boldsymbol{x}_j$ is the one with the smallest ss_j given by

$$ss_{(1)} = \min_j ss_j.$$

Denote the corresponding quantities for the Gaussian covariates by SS_j and $SS_{(1)}$ respectively. From (4)

$$1 - SS_j/ss_y \stackrel{D}{=} B_{1/2,(n-1)/2}. \tag{8}$$

As the Gaussian covariates are independent it follows that

$$\boldsymbol{P}(SS_{(1)} \leq ss_{(1)}) = 1 - B_{1/2,(n-1)/2}(1 - ss_{(1)}/ss_y)^k. \tag{9}$$

If the best of the $\boldsymbol{x}_j$ is $\boldsymbol{x}_{j_1}$ the right hand side is referred to as the P-value of $\boldsymbol{x}_{j_1}$. The smaller the P-value the more relevant $\boldsymbol{x}_{j_1}$. This corresponds to the role of testing $\beta_{j_1} = 0$ in the linear model where the smaller the P-value the more significant the covariate $\boldsymbol{x}_{j_1}$.

In some applications it is useful to be less strict when selecting variables. This may be seen as a trade-off between reducing the number of false negatives, not selecting variables with some explanatory value, at the risk of more false positives, selecting variables with no explanatory value. This may be done as follows.

The random variables $B_{1/2,(n-1)/2}(1 - SS_j/ss_y)$ are i.i.d. $U(0,1)$ so that

$$\begin{aligned} \boldsymbol{P}(SS_{(1)} \leq ss_{(1)}) &= 1 - B_{1/2,(n-1)/2}(1 - ss_{(1)}/ss_y)^k \\ &= 1 - B_{k,1}(B_{1/2,(n-1)/2}(1 - ss_{(1)}/ss_y)). \end{aligned} \tag{10}$$

This can be extended to the νth order $SS_{(\nu)}$ to give

$$\boldsymbol{P}(SS_{(\nu)} \leq ss_{(1)}) = 1 - B_{k-\nu+1,\nu}(B_{1/2,(n-1)/2}(1 - ss_{(1)}/ss_y)). \tag{11}$$

Comparing $ss_{(1)}$ with $SS_{(\nu)}$ is less strict than comparing it with $SS_{(1)}$. Although ν has a direct interpretation when an integer this is not necessary in (11) other values may be chosen.

We state once again that all the above probabilities are exact and hold for all $\boldsymbol{y} \neq \boldsymbol{0}$ and for all covariates $\boldsymbol{x}_j, j = 1, \ldots, k$.

To incorporate this into a stepwise procedure a cut-off value α for the P-value must be specified, for example $\alpha = 0.05$. Suppose at stage ℓ of the procedure

ℓ covariates have been selected. We denote the sum of the squared residuals by $ss_{r,\ell}$ with $ss_{r,0} = ss_y$ and the set of selected covariates by $\mathcal{S}_\ell$. For each $j \notin \mathcal{S}_\ell$ regress $\boldsymbol{y}$ on the covariates in $\mathcal{S}_\ell \cup \boldsymbol{x}_j$ and denote the smallest sum of squared residuals taken over j by $ss_{\ell,(1)}$. This is now compared with the smallest sum of squares obtainable by considering $k - \ell$ random Gaussian covariates $\boldsymbol{Z}_\kappa, \kappa = 1, \ldots, k - \ell$. These are so to speak chosen anew for each ℓ. This is asking the question as to whether the remaining covariates are better than Gaussian noise. Regressing $\boldsymbol{y}$ on the covariates in $\mathcal{S}_\ell \cup \boldsymbol{Z}_\kappa$ gives a random sum of squared residuals $SS_{\ell,\kappa}$. From (7) we have

$$1 - SS_{\ell,\kappa}/ss_{r,\ell} \stackrel{D}{=} \text{Beta}(1/2, (n - \ell - 1)/2), \tag{12}$$

and the P-value for the best of the $\boldsymbol{x}_j \notin \mathcal{S}_\ell$ is given by

$$\boldsymbol{P}(SS_{\ell,(1)} \leq ss_{\ell,(1)}) = 1 - \text{B}(1 - ss_{\ell,(1)}/ss_{r,\ell}, 1/2, (n - \ell - 1)/2)^{k-\ell}. \tag{13}$$

If the P-value is greater than the cut-off value α the procedure terminates. Otherwise the best covariate is included and the procedure continues. The command is

```
fstepwisey,x,alpha,kmax,nu=1,kexk=0,offset=TRUE,time=FALSE,verbose=FALSE)
```

Applying this to the leukemia data mentioned in Section 1 with $(n, k) = (72, 3571)$ gives

```
> fstepwise(ly.original,lx.original,0.05,10,time=T)[[1]]
   user  system elapsed
  0.022   0.000   0.021
     [,1]         [,2]
[1,] 1182 0.0000000000
[2,] 1219 0.0008577131
[3,] 2888 0.0035805523
```

Here 1182, 1219 and 2888 are the selected covariates with P-values 0.0000000000, 0.0008577131 and 0.0035805523 respectively.

The extension of the above to the νth order statistic is

$$\boldsymbol{P}(SS_{\ell,(\nu)} \leq ss_{\ell,(1)}) = \\ 1 - B_{k-\ell+1-\nu,\nu}(B_{1/2,(n-\ell-1)/2}(1 - ss_{\ell,(1)}/ss_{r,\ell})) \tag{14}$$

which is the probability that the νth best of the Gaussian covariates is better than the best of the remaining covariates.

2.3 False positives and ν

The left-hand side of (14) requires ν to be an integer. The right-hand side does not require this and the P-value

$$1 - B_{k-\ell+1-\nu,\nu}(B_{1/2,(n-\ell-1)/2}(1 - ss_{\ell,(1)}/ss_{r,\ell}))$$

is well defined for non-integer ν but then loses its interpretation in terms of the left-hand side.

Even when ν is an integer it is not clear what effect the choice of ν has on the results. This can be done by relating ν to the concept of false positives. A false positive is selecting a covariate which is no better than Gaussian noise. This may be quantified by using Gaussian noise as the covariates and then counting the number of covariates selected all of which are false positives. This is shown in Table 1 and may be of help in choosing a value of ν for any given data set. The 90% quantiles and medians in Table 1 are based on 1000 simulations.

ν	2	3	4	5	6	7	8	9	10
$\alpha = 0.05$	1	2	3	4	5	6	7	8	10
	0	1	1	2	3	3	4	5	6
$\alpha = 0.01$	1	1	2	3	4	5	6	7	7
	0	0	1	1	2	2	3	4	4

Table 1: 90% quantiles (first lines) and medians (second lines) of the number of false positives as a function of ν and α for $(n, k) = (1000, 1000)$

If n is small and k large even small values of ν can lead to as many as n false positives. This is the case for the leukemia data with $\nu = 5$. For $\nu = 2, \ldots, 7$ the 90% quantiles are respectively

$$1,\ 3,\ 5,\ 8,\ 14,\ 22.$$

Putting $\nu = 3$ gives

```
> fstepwise(ly.original,lx.original,0.05,10,nu=3,time=T)[[1]]
   user  system elapsed
   0.03    0.00    0.03
     [,1]         [,2]
[1,] 1182 0.000000e+00
[2,] 1219 1.051452e-10
[3,] 2888 7.664817e-09
[4,] 1946 3.353905e-03
[5,] 2102 6.026398e-04
```

Thus $\nu = 3$ leads to two more covariates but as the 90% quantile is 3 there is no evidence that these additional covariates are correct positives. This can be seen more precisely by regressing ly.original on the covariates 1-5 above to give the following regression P-values.

```
> b<-lm(ly.original~lx.original[,c(1182,1219,2888,1946,2102)])
>  as.double(summary(b)[[4]][2:6,4])
[1] 2.286646e-15 4.207564e-10 1.549639e-06 1.041936e-06 4.478674e-05
```

We refer to these as regression P-values to distinguish them from the P-values of (9). Typically the regression P-values are much smaller. The small regression P-values for 1946 and 2102 suggests that they may be relevant. They may indeed be relevant but not on the basis of the small regression P-values. To see this replace lx.original by Gaussian noise tmpx except for 1182, 1219 and 2888 which are the first three covariates of tmpx. Running the stepwise procedure gives

```
> set.seed(2345)
> tmpx<-rnorm(lx.original)
> dim(tmpx)<-dim(lx.original)
> tmpx[,1:3]<-lx.original[, c(1182, 1219, 2888)]
> fstepwise(ly.original,tmpx,0.05,10,nu=3)[[1]]
     [,1]         [,2]
[1,]    1 0.000000e+00
[2,]    2 1.051452e-10
[3,]    3 7.664817e-09
[4,] 1698 5.235440e-03
```

and regress ly.original on these 4 covariates to give the regression P-values

```
> b<-lm(ly.original~tmpx[,c(1:3,1698)])
>  as.double(summary(b)[[4]][2:5,4])
[1] 3.630659e-21 1.242047e-08 9.651025e-07 9.631336e-05
```

The Gaussian covariate 1698 has a regression P-value of 9.631336e-05 in spite of being a false positive. We use this idea below when evaluating selected covariates for real data.

It might seem that the calculation of the regression P-values given above relies on the model (1) but this is not the case. It is done for convenience. Exactly the same P-values can be obtained from (7) which is based on Gaussian covariates and applies for all $\boldsymbol{y}$ and all covariates .

Most of the data sets considered here have either k small or n small and k large. For such data sets only the default value $\nu = 1$ and possibly $\nu = 2$ are appropriate. The exceptions are Tutorials 1 and 2 with $(n, k) = (1000, 1000)$ where Table 1 suggests $\nu = 10$ can be a reasonable choice.

The definition of false postive given above is an emprical one. In simulations based on the linear model a false positive is the selection of a covariate $\boldsymbol{x}_j$ with $\beta_j = 0$. A false negative is the ommission of a covariate $\boldsymbol{x}_j$ with $\beta_j \neq 0$. This is the defintion we use in simulations, Tables 2 and 3. In the simulations of graphs in Section 3.5 a false negative is the omission of an edge where the true graph has an edge. A false postive is the inclusion of an edge where the true graph has no edge.

2.4 Repeated Gaussian covariate stepwise regression

Once the first covariate has been chosen the stepwise procedure is conditional on this covariate. More generally once a subset has been chosen the next covariate to be chosen is dependent on this subset. To illustrate this we consider the colon cancer data with $(n, k) = (62, 2000)$. The stepwise procedure results in

```
fstepwise(colon.y,colon.x,1,2,time=T)[[1]]
   user  system elapsed
  0.004   0.001   0.005
     [,1]          [,2]
```

```
[1,]  493 7.402367e-08
[2,]  175 4.311166e-01
```

For any cut-off P-value $\alpha < 0.43$ only one covariate is chosen, namely 493. This does not mean that only covariate 493 is relevant but that given 493 the remaining 1999 are in a sense no better than white Gaussian noise.

If covariate 493 is eliminated and the stepwise procedure applied to the remaining covariates again just one covariate is chosen, 377 with a P-value 1.355905e-07. The covariates 493 and 377 are highly correlated with correlation coefficient of 0.778. This explains why 377 is no longer considered once 493 has been included.

Now 377 can be excluded and the procedure continued in this manner until the P-value of the best of the remaining covariates exceed the specified cut-off value α. The value $\alpha = 0.05$ results in 82 covariates being selected. The time required was 0.22 seconds. The first ten are

```
     [,1] [,2]         [,3]
 [1,]    1  493 7.402367e-08
 [2,]    2  377 1.355905e-07
 [3,]    3  249 1.134563e-06
 [4,]    4 1635 2.283614e-06
 [5,]    4  576 2.815082e-02
 [6,]    5 1423 2.760755e-05
 [7,]    5  353 7.996075e-04
 [8,]    6  625 3.166107e-05
 [9,]    6  739 8.354274e-04
[10,]    7  245 1.931080e-04
```

The value $\alpha = 0.01$ results in 45 selected covariates in 0.15 seconds. The command is

```
fstepstepwise(y,x,alpha,kmax,nu=1,offset=TRUE,time=FALSE,verb=FALSE)
```

The Gaussian covariate procedure may be interpreted as offering a linear approximation to the dependent covariate $\boldsymbol{y}$ whereby the inclusion of an additional covariate does not lead to a better approximation as this additional covariate is no better than Gaussian noise. This was demonstrated above using the leukemia data. The repeated Gaussian procedure may be seen as offering a series of such approximations or attempts to specify all those covariates which are explanatory for $\boldsymbol{y}$.

Lasso offers a single regression-type relation, a point mentioned in [Cox and Battey, 2017]. It can also be seen in Section 5 of [Bellec et al., 2017] where a true β^* in (1) is assumed and compared with lasso estimators.

This completes the description of the Gaussian covariate method. It is extremely simple and there is no mention of regression parameters β or the variance σ^2 of (1). This contrasts with the treatment of lasso in [Bellec et al., 2017] where all the values of the tuning parameter λ considered involve σ. Indeed the estimation of σ is one of the main problems with lasso as many optimality results for the choice of λ depend on the value of σ.

2.5 L_1 regression

The idea is not restricted to least squares. It can be applied to L_1 regression but then simulations are necessary. We take the Brownlee stack loss data as an example. Suppose the covariates Air Flow and Water Temperature have been selected. Including the covariate Acid Concentration reduces the sum of the absolute residuals from 43.69355 to 42.08116. In a simulation with Gaussian covariates replacing Acid Concentration the Gaussian covariates

was better in 20% of the simulations giving a P-value of 0.204 for Acid Concentration. This uses the R package quantreg. The corresponding P-value for least squares regression is 0.344.

For robust regression with a smooth ψ function and for non-linear regression such as logistic a chi-squared approximation is available (see [Davies, 2017]) removing the need for simulations.

2.6 Why Gaussian?

The method started with the question as to whether it was possible to judge the relevance of a covariate without assuming the model (1). The initial data set was the Brownlee stack loss data and the unfortunate covariate was Acid Concentration. This was replaced by the cosine of the average daily temperature in Berlin on the first 21 days of January 2013. The Acid Concentration won but only just. Relying on empirical alternatives to the covariates was not a promising option but from there it was but a short step to simulating alternatives. Initially an attempt was made to model the alternative covariates. If the covariate was 0-1 use the binomial, if integer valued a Poisson etc (see page 279 of [Davies, 2014]). The results showed that the modelling was not worth it. They were essentially the same when one used Gaussian alternatives.

In the case of one Gaussian covariate $\boldsymbol{Z} = (Z_1, \ldots, Z_n)$ the sum of squared residuals is

$$\frac{(\sum_{i=1}^n y_i Z_i)^2}{\sum_{i=1}^n Z_i^2} \overset{D}{\approx} \frac{\sum_{i=1}^n y_i^2}{n} \chi_1^2. \tag{15}$$

If the Z_i are not Gaussian but say Bernoulli ± 1 then (15) will still hold

asymptotically but this will require conditions on $\boldsymbol{y}$. More generally if the Z_i have finite variance then subject to conditions on the y_i (15) will hold. In this case there seems to be no reason not to use the exact result for Gaussian covariates.

We start from $\sum_{i=1}^n y_i^2$ so when calculating the percentage reduction in the sum of squares due to regression on the Z_i the expression $\sum_{i=1}^n y_i^2$ cancels out just as σ^2 cancels out in the F-test. This is why the calculated probabilities hold for all $\boldsymbol{y}$. If the Z_i are replaced by i.i.d. Cauchy random variables C_i then (15) becomes

$$\frac{(\sum_{i=1}^n y_i C_i)^2}{\sum_{i=1}^n C_i^2} = \left(\sum_{i=1}^n |y_i|\right)^2 \frac{\tilde{C}^2}{\sum_{i=1}^n C_i^2}$$

where $\tilde{C}$ is a standard Cauchy random variable. There is no cancelling out and the distribution will depend on $\boldsymbol{y}$ even asymptotically. Moreover $(\sum_{i=1}^n |y_i|)^2$ is larger than $\sum_{i=1}^n y_i^2$ indicating that Cauchy random variables are less exacting than Gaussian random variables.

3 Comparison of Lasso, knockoff and the Gaussian covariate procedure

The comparisons given here are purely empirical. It is possible to prove theoretical results on the Gaussian covariate method as a first attempt in [Davies, 2017] shows but this will not be pursued further. The comparisons are given in detail and it should be possible to repeat then using the software available as an auxiliary file. The version of lasso to be used is the cross-validation option cv.glmnet provided in the R package glmnet.

Lasso and knonckoff involve randomization so that repeated use on the same data set will in general produce different selected covariates. In such cases we take the union of the selected covariates. The Gaussian covariate method does not involve randomization and gives the same result each time. The default values of α and ν are 0.05 and 1 but others will be occasionally used.

3.1 Tutorials 1 and 2

The simulations of Table 2 are based on the Tutorials 1 and 2 respectively of with the parameters as given there.

The default choice $\nu = 1$ avoids false positives possibly at the cost of false negatives. Using this value of ν in Tutorial 1 results in on average 15 covariates being chosen with no false positives. Putting $\nu = 5$ results in 53 covariates being chosen. Table 1 suggests that of 38 additional covariates at most 4 will be false positives. Putting $\nu = 10$ results 62 being chosen of which about 10 are false positives. Given the 4 with $\nu = 5$ indicates that of the additional 9 at most 6 will be false negatives. In all $\nu = 10$ suggests at most 10 false negatives (Table 1) giving about 52 correctly identified covariates. Table 2 shows that these estimates are quite good. Similar calculations apply to Tutorial 2.

It follows from Table 2 that lasso selects on average approximately 140 covariates in Tutorial 1 and 100 in Tutorial 2 which is somewhat excessive.

In terms of the sum of false positives and false negatives knockoff and the choices $\nu = 5$ and $\nu = 10$ are comparable. The default value of the false discovery rate fdr is 0.1 but in Tutorial 2 it is set to 0.2. If the default value 0.1 is used the false positive and negative values become 2.48 and 44.6

	Tutorial 1			Tutorial 2		
method	fp	fn	time	fp	fn	time
lasso	82.4	0.52	7.89	56.0	15.9	9.79
knockoff	5.58	10.0	63.9	7.00	35.1	53.9
$\nu = 1$	0.00	46.2	0.25	0.00	56.5	0.04
$\nu = 5$	3.36	11.6	2.29	2.78	42.5	0.44
$\nu = 10$	7.02	5.82	3.33	7.00	35.4	0.94

Table 2: Comparison of lasso, knockoff and Gaussian covariates with $\alpha = 0.05$ based on Tutorials 1 and 2.

respectively. Also in Tutorial 2 the covariance matrix Sigma was used to construct the knockoff variables. This seems to improve the performance if only slightly. If the knockoff filter is used as in Tutorial 1 the average numbers of false positives and negatives become 6.32 and 36.72 respectively. The main difference between knockoff and the Gaussian covariate method is the computing time. In Tutorial 1 it is 20 times slower that the Gaussian method with $\nu = 10$ and 50 times slower in Tutorial 2. For real data the difference can be much larger with factors over 1000.
The commands for Tutorial 1 and Tutorial 2 are as follows:

```
ftut(1,1000,1000,60,4.5,0.1,0.05,50)
ftut(2,1000,1000,60,7.5,0.2,0.05,50)
```

3.2 Red wine data

For the red wine data with $(n, k) = (1599, 11)$ the dependent variable is variable 12 and gives subjective evaluations of the quality of the wine.

Five applications of lasso gave 4, 4, 6, 6 and 7 selected covariates. Their union was $\{1, 2, 5, 7, 9, 10, 11\}$. Five applications of knockoff gave 10, 7, 10, 11 and 10 selected covariates. The Gaussian method with cut-off P-value $\alpha = 0.05$ gives

```
> fstepwise(redwine[,12],redwine[,1:11],0.05,10)[[1]]
     [,1]         [,2]
[1,]   11 0.000000e+00
[2,]    2 0.000000e+00
[3,]   10 2.032414e-10
[4,]    7 1.027568e-04
[5,]    5 1.002491e-04
[6,]    9 1.096799e-03
```

The chosen covariates are in order alcohol, volatile-acidity, sulphates, total-sulfur-dioxide, chlorides, pH (see Table 5 of [Lockhart et al., 2014]).
The repeated Gaussian method of Section 2.4 gives

```
> fstepstepwise(redwine[,12],redwine[,1:11],0.05,10)[[1]]
      [,1] [,2]         [,3]
 [1,]    1   11 0.000000e+00
 [2,]    1    2 0.000000e+00
 [3,]    1   10 2.032414e-10
 [4,]    1    7 1.027568e-04
 [5,]    1    5 1.002491e-04
 [6,]    1    9 1.096799e-03
 [7,]    2    3 0.000000e+00
 [8,]    2    8 0.000000e+00
 [9,]    2    1 1.918132e-12
[10,]    2    4 4.110445e-07
[11,]    3    6 4.283397e-02
```

method	false pos.	false neg.
lasso	0.04	1.26
knockoff	1.15	0.09
$\nu = 1$	0.05	0.00
$\nu = 2$	0.46	0.00
$\nu = 3$	1.45	0.00

Table 3: Comparison of lasso, knockoff and Gaussian covariates based on the red wine data

This indicates that the covariates 3, 8, 1 and 4 are also strongly related to the quality of the wine but not conditional on the first six.
Finally we give the result of a simulation. The five covariates 3, 8, 1, 4 and 6 are replaced by i.i.d. $N(0, 1)$ variables. Choosing one of these is a false positive. Not choosing one of the remaining six covariates is defined as a false negative. The result of 100 simulations are given in Table 3.
The false discovery rate was set to 0.2 which is the best for this simulation. Nevertheless in over 40% of the cases a Gaussian covariate was chosen. The cut-off P-value for the Gaussian methods was $\alpha = 0.05$. If this is reduced to $\alpha = 0.01$ the average false positive numbers become 0.02, 0.18 and 0.70. The command is

```
>fredwine(redwine,nsim=100,alpha=0.05)
```

3.3 Cancer data

3.3.1 Colon data

The size of colon cancer data is $(n, k) = (62, 2000)$.

A first application of lasso results in the following 15 covariates

$$14, 249, 377, 493, 576, 625, 792, 1231, 1360, 1473, 1582, 1679, 1772, 1843, 1924.$$

The next immediate application of lasso gives the following

$$249, 377, 493, 625.$$

How are these results to be interpreted? The initial formulation of the problem in [Tibshirani, 1996] and the analysis given in [Bellec et al., 2017] suggest that the aim is to give a good parsimonious linear model for $\boldsymbol{y}$. This is also the interpretation of lasso given in [Cox and Battey, 2017]. The first application does not result in a parsimonious linear model but the second may do so. Regressing colon.y on these four covariates results in the regression P-values

$$0.02517,\ 00379,\ 0.58052,\ 0.018820.$$

Covariate 493 with a regression P-value of 0.58052 contributes little and will be eliminated. The model based on 249, 377 and 625 gives regression P-values 4.71e-03, 8.26e-05 and 1.10e-02. Covariate 625 is not very convincing and eliminating it leads to the model base solely on 249 and 377 with P-values 7.77e-05 and 9.10e-06 which looks more convincing.

All these P-values do not take into account that the four covariates were chosen out of 2000. This can be done by calculating the P-values as defined in (14) with $\nu = 1$. The P-value for 493 given 249, 377 and 625 is

$$1 - \text{pbeta}(\text{pbeta}(1 - ss_{1234}/ss_{124}, 0.5, (62-5)/2), (2001-5), 1)$$

where ss_{1234} and ss_{124} are the sums of squares residuals based on 249, 377, 493, 625 and 249, 377, 625 respectively. This gives a P-value of 1 in contrast

to the previous one of 0.58052. The P-values corresponding to the regression P-values 7.77e-05 and 9.10e-06 are 0.0214619 and 0.1639252 respectively. Doing this for all possible 15 models leads in the end to just the four models based on each individual covariate.

The reason for the above is that the covariates are correlated. Regressing 493 on 249 and 377 results in regression P-values 1.803856e-06 and 3.452139e-08 respectively.

We point out that the first three covariates for the repeated Gaussian covariate stepwise procedure given in Section 2.4 are 493, 377 and 249 in that order. The covariate 625 is number 8 but together with 739. This means that the linear model based on 625 and 739 fulfills the conditions on the P-values and shows that not all models are based on a single covariate.

On the basis of this is is better to interpret the output of lasso as an attempt to specify those covariates which are closely related to colon.y. This is the attitude taken in [Cox and Battey, 2017]for their method. This applies also to the output of knockoff.

With only four covariates it was possible to check all subsets. With 15 and more this becomes practically impossible and even if possible probably not worth the effort. We shall therefore only consider single covariates, pairs and triplets. For the pairs and triplets we calculate the P-value of each covariate in turn and then list all the pairs and triplets for which all these P-values are less than a cut-off value α taken here to be 0.01.

Five application of lasso resulted in 15, 4, 5, 25 and 29 variables. The union consisted of the following 32 variables

```
14,  164,  175,  249,  353,  377,  493,  576,  611,  625,  654,  788,
792,  823, 1073, 1094, 1221, 1231, 1256, 1346, 1360, 1400, 1473,
```

1549, 1570, 1582, 1668, 1679, 1772, 1843, 1873, 1924

Five applications of the knockoff filter with fdr =0.5 resulted in 12, 3, 6, 35 and 4 variables being selected. The union consisted of the 38 variables

14, 164, 175, 353, 377, 576, 611, 654, 788, 792, 823, 966, 1073, 1094, 1110, 1123, 1146, 1231, 1241, 1346, 1360, 1400, 1420, 1482, 1549, 1570, 1622, 1649, 1740, 1772, 1827, 1843, 1873, 1893, 1924, 1935, 1976, 1989.

The colon cancer data set was used in Section 2.4 as an example for the repeated Gaussian stepwise. The first ten covariates together with their P-values were listed there.

The computing times were two seconds for lasso, two hours fifty minutes for knockoff and 0.22 seconds for the Gaussian covariate method.

The chosen covariates will be evaluated by calculating the P-values for single, pairs and triplets of covariates as described at the beginning of this section. The cut-off P-value was set to $\alpha = 0.01$ and the smallest P-values are given. The result for lasso was

```
[1,] 1843 6.495852e-09
 [2,]  493 7.398644e-08
 [3,] 1772 1.078545e-07
 [4,]  377 1.355903e-07
 [5,]  249 1.135130e-06
 [6,] 1873 1.214555e-06
 [7,] 1231 2.822242e-06
 [8,] 1473 4.105941e-06
 [9,]   14 6.468307e-06
[10,] 1400 6.848481e-06
[11,] 1360 1.044163e-05
[12,]  576 1.516268e-05
[13,] 1256 1.737674e-05
```

```
[14,] 1679 3.054090e-05
[15,]  625 3.175638e-05
[16,] 1924 4.711134e-05
[17,] 1549 3.631080e-04
[18,] 1346 4.018934e-04
[19,] 1582 1.403567e-03
[20,]  792 3.259051e-03
[21,]  353 3.770271e-03
[22,]  175 4.906957e-03
[23,] 1668 8.633158e-03.
```

The corresponding result for knockoff was

```
[1,] 1843 6.495852e-09
 [2,]  377 1.355903e-07
 [3,]   14 2.778769e-07
 [4,] 1873 1.214555e-06
 [5,] 1772 2.463932e-06
 [6,] 1231 2.822242e-06
 [7,] 1360 1.044163e-05
 [8,]  576 1.516268e-05
 [9,] 1400 1.168938e-04
[10,] 1935 1.411647e-04
[11,] 1549 3.631080e-04
[12,] 1346 4.018934e-04
[13,] 1924 4.747120e-04
[14,]  175 4.906957e-03
[15,] 1110 8.805362e-03
```

The Gaussian stepwise procedure yields 71 covariates. The first 30 are

```
 [1,] 1423 1.106895e-10
 [2,] 1843 6.495852e-09
 [3,] 1494 2.047067e-08
 [4,]  493 7.398644e-08
 [5,] 1293 8.035832e-08
 [6,] 1772 1.078545e-07
 [7,]  377 1.355903e-07
 [8,]  513 1.999018e-07
 [9,] 1406 2.378230e-07
[10,]  992 4.819964e-07
[11,] 1634 5.697313e-07
[12,]  249 1.135130e-06
[13,] 1648 1.161793e-06
[14,] 1060 1.316599e-06
[15,] 1042 1.476826e-06
[16,] 1730 2.032950e-06
[17,] 1900 2.209020e-06
[18,] 1635 2.285901e-06
[19,]  365 2.311574e-06
[20,]  187 2.586279e-06
[21,] 1231 2.822242e-06
[22,]  625 4.630786e-06
[23,]  792 4.863052e-06
[24,]  111 4.885140e-06
[25,]   14 6.468307e-06
[26,] 1771 9.208768e-06
[26,] 1771 9.208768e-06
[27,]  138 1.209879e-05
[28,]  576 1.516268e-05
[29,] 1256 1.737674e-05
```

```
[30,] 1153 2.372798e-05
```

Comparing the complete results (see comp.res) it is seen that all the lasso and knockoff variables apart from 175, 1110, 1360, 1549, 1668, 1924 and 1935 are included in the list of the Gaussian covariate method. There are 10 lasso variables, 6 knockoff variables and 26 Gaussian covariate variables with P-values less than 1e-5.

3.3.2 Leukemia data

The size of the leukemia data is $(n, k) = (72, 3571)$.
The knockoff procedure with fdr=0.5 resulted in 31 covariates. The time required was five hours 20 minutes. In view of this no further analysis was carried out as it would require of the order of one day computing time.
The lasso is much faster requiring 0.3 seconds on average. Five applications of lasso resulted in 12, 24, 14, 24 and 11 covariates, being selected. Their union was the following 26 covariates:

```
219,  456,  626,  657,  672,  888,  956,  979, 1099 ,1108, 1182, 1219,
1620, 1652, 1946, 2230, 2239, 2481, 2537, 2727, 2859, 2888, 3098,
3158, 3345, 3441.
```

The P-value procedure with cut-off value 0.01 reduced this to the following 25 covariates.

```
 [1,]  456 0.000000e+00
 [2,]  956 0.000000e+00
 [3,]  979 0.000000e+00
 [4,] 1182 0.000000e+00
 [5,] 1652 0.000000e+00
```

```
 [6,] 2481 0.000000e+00
 [7,] 3441 0.000000e+00
 [8,] 1099 5.548895e-12
 [9,]  626 3.804956e-11
[10,] 1219 5.152545e-11
[11,] 2230 2.852119e-10
[12,]  672 6.273695e-07
[13,]  219 3.663922e-06
[14,] 1946 2.117592e-05
[15,] 2727 4.329760e-05
[16,]  657 4.817129e-05
[17,] 1620 5.316102e-05
[18,] 3158 1.001275e-04
[19,] 3098 1.013139e-03
[20,] 2239 1.258528e-03
[21,] 2537 1.984932e-03
[22,] 3345 3.509971e-03
[23,] 2888 3.579551e-03
[24,] 2859 6.230513e-03
[25,]  888 7.516447e-03
```

The repeated Gauss procedure with $\alpha = 0.05$ resulted in 420 covariates which is probably too many to be useful. Setting $\alpha = 0.000001$ gave 41 covariates all of which survived the P-value procedure.

```
     [,1]         [,2]
 [1,]  435 0.000000e+00
 [2,]  456 0.000000e+00
 [3,]  874 0.000000e+00
 [4,]  956 0.000000e+00
 [5,]  979 0.000000e+00
 [6,] 1182 0.000000e+00
```

```
 [7,] 1356 0.000000e+00
 [8,] 1652 0.000000e+00
 [9,] 2049 0.000000e+00
[10,] 2481 0.000000e+00
[11,] 2789 0.000000e+00
[12,] 3441 0.000000e+00
[13,]  436 3.963496e-13
[14,] 1099 5.548895e-12
[15,] 3038 7.134293e-12
[16,]  626 3.804956e-11
[17,] 1219 5.152545e-11
[18,] 3216 2.413769e-10
[19,]  907 5.513223e-10
[20,] 3162 6.512024e-10
[21,] 2198 2.000377e-09
[22,] 2226 2.471636e-09
[23,] 2079 6.085156e-09
[24,]  918 7.884187e-09
[25,] 2220 1.093568e-08
[26,] 2230 1.228208e-08
[27,] 2145 1.499945e-08
[28,] 1249 2.457288e-08
[29,]  951 4.670623e-08
[30,] 2141 6.791133e-08
[31,]  851 8.359766e-08
[32,] 1014 1.493699e-07
[33,] 2546 2.594952e-07
[34,] 1053 2.742668e-07
[35,] 2911 3.024876e-07
[36,] 1104 3.150270e-07
```

```
[37,] 1001 5.385431e-07
[38,] 2449 6.480607e-07
[39,] 1020 8.243764e-07
[40,] 3466 9.369273e-07
[41,]  990 9.641691e-07
```

Of the lasso covariates with P-values less than 1e-6 only the covariate 672 is not on the Gaussian procedure list.

3.3.3 Prostate cancer and osteoarthritis

The size of the prostate data $(n, k) = (102, 6033)$ which suggest a very long time using the knockoff filter. It will not be considered.
Lasso was applied 5 times with 18, 3, 6, 14, and 14 covariates being selected giving in all 18 covariates. The P-value procedure reduced this to the following 12 with associated P-values

```
[1,] 1839 0.000000e+00
 [2,] 2619 0.000000e+00
 [3,] 4288 0.000000e+00
 [4,] 5016 0.000000e+00
 [5,] 5035 0.000000e+00
 [6,] 3423 2.008727e-12
 [7,] 4898 1.936399e-07
 [8,] 2377 4.988987e-06
 [9,] 1788 1.328393e-04
[10,] 1903 2.919642e-04
[11,] 2388 3.082575e-04
[12,] 2003 4.721405e-04
```

Of these 6 have regression P-values less than 1e-15.

The Gaussian covariate procedure with $\alpha = 0.00001$ gave 33 covariates all of which survived the P-value procedure.

```
[1,] 2619 0.000000e+00
 [2,] 5016 0.000000e+00
 [3,] 1839 0.000000e+00
 [4,] 4701 0.000000e+00
 [5,] 4155 0.000000e+00
 [6,] 4212 0.000000e+00
 [7,] 3705 0.000000e+00
 [8,] 2746 0.000000e+00
 [9,] 5035 0.000000e+00
[10,] 3392 0.000000e+00
[11,] 3006 0.000000e+00
[12,] 4898 6.694645e-13
[13,] 1640 6.695755e-13
[14,] 4335 1.339151e-12
[15,] 5808 4.017453e-12
[16,] 3934 9.374057e-12
[17,] 3423 1.205236e-11
[18,] 3833 3.883538e-11
[19,] 2425 8.838397e-11
[20,] 4261 3.722840e-10
[21,] 4001 3.248780e-09
[22,] 5249 4.436541e-09
[23,] 5230 8.823666e-09
[24,] 5639 2.574384e-08
[25,] 4849 2.764811e-08
[26,] 2386 7.471659e-08
[27,] 5783 8.060352e-08
[28,] 1540 1.117588e-07
```

```
[29,] 2428 1.809616e-07
[30,] 2377 1.011739e-06
[31,] 4448 1.288842e-06
[32,] 3366 3.940713e-06
[33,] 4262 7.863072e-06
```

Of the lasso covariates with P-values less than 1e-5 only 4288 is not on the Gaussian procedure list.
The osteoarthritis data set was analysed in ([Cox and Battey, 2017]). The authors selected the following 17 covariates:

```
7235 11643 25125 25470 25744 27642 27920 29679 33385
36409 37443 44276 45991 46771 48415 48433 48549
```

The P-value procedure with cut-off value 0.01 reduces this to the following 15 covariates

```
 [1,] 37443 2.925660e-10
 [2,] 27642 3.684164e-10
 [3,] 33385 2.505773e-08
 [4,] 27920 1.694824e-07
 [5,] 36409 5.585733e-07
 [6,] 25744 7.581411e-07
 [7,] 48415 1.663886e-06
 [8,] 46771 2.609815e-06
 [9,] 25470 3.000447e-06
[10,]  7235 3.314328e-06
[11,] 25125 2.407158e-05
[12,] 48549 1.922365e-04
[13,] 29679 1.984815e-04
[14,] 45991 3.831023e-04
[15,] 48433 6.429226e-04
```

Lasso applied 5 times selected 43, 50, 23, 29 and 40 covariates giving 50 in all. The P-value procedure reduces this to 35

```
 [1,]   939 0.000000e+00
 [2,]  3630 0.000000e+00
 [3,] 11499 0.000000e+00
 [4,] 44902 5.417777e-12
 [5,] 26207 7.043111e-11
 [6,] 43951 4.009237e-10
 [7,] 29522 6.176266e-10
 [8,] 41799 1.246114e-09
 [9,] 23983 2.936435e-09
[10,] 24232 4.624073e-08
[11,] 38561 8.087823e-08
[12,] 30451 1.315761e-07
[13,] 33321 5.230429e-07
[14,] 23259 8.717633e-07
[15,] 35023 1.555876e-06
[16,]  8669 2.285539e-06
[17,] 44758 5.331563e-06
[18,] 23038 6.508366e-06
[19,] 46172 9.159328e-06
[20,] 45991 1.382485e-05
[21,]  3685 3.096310e-05
[22,] 13650 9.284205e-05
[23,] 21757 2.463220e-04
[24,] 30997 3.918151e-04
[25,]  5939 4.768018e-04
[26,] 36417 6.450063e-04
[27,]  4216 9.412395e-04
[28,] 14724 1.664579e-03
```

```
[29,]  6661 2.026067e-03
[30,] 26917 3.145650e-03
[31,] 46700 3.281872e-03
[32,] 24462 4.818816e-03
[33,] 48035 4.884352e-03
[34,] 17062 4.888099e-03
[35,]  5210 8.442471e-03
```

The repeated Gaussian covariate method with $\alpha = 0.0001$ gave 32 covariates all of which survived the P-value procedure

```
[1,] 11499 0.000000e+00
 [2,] 31848 0.000000e+00
 [3,] 44902 0.000000e+00
 [4,] 10546 0.000000e+00
 [5,] 34803 0.000000e+00
 [6,] 30451 0.000000e+00
 [7,]   939 0.000000e+00
 [8,]  7896 0.000000e+00
 [9,] 22705 0.000000e+00
[10,] 37443 0.000000e+00
[11,] 28816 0.000000e+00
[12,] 29522 0.000000e+00
[13,] 26207 5.417888e-12
[14,]  3630 1.625333e-11
[15,] 46979 2.167155e-11
[16,] 25744 5.417888e-11
[17,] 10374 9.210221e-11
[18,] 43951 4.009237e-10
[19,] 25125 6.393108e-10
[20,] 41799 1.647038e-09
[21,] 39141 2.557243e-09
```

```
[22,] 24232 2.806466e-09
[23,] 26795 4.930278e-09
[24,]  6740 5.791723e-09
[25,] 44664 6.095124e-09
[26,] 25241 1.106875e-08
[27,] 21350 1.783677e-07
[28,] 24266 8.452877e-07
[29,] 38561 1.122988e-06
[30,] 22929 2.326038e-06
[31,] 44758 5.331563e-06
[32,] 34076 8.007249e-05
```

The Gaussian list which has 19 covariates with a P-value less than 1e-9. The Cox-Battey list has two one of which 27642 is not on the Gaussian list. The lasso procedure has 7 such covariates all of which are on the Gaussian list. The osteoarthritis data proved too large for the knockoff filter requiring 17.7 GB of memory.

3.4 Boston housing data and interactions

For the Boston housing data $(n, k) = (506, 13)$. Allowing for interactions of order up to and including eight increases the number of covariates from 13 to 203490 to give $(n, k) = (506, 203490)$.
Lasso often failed with the laptop complaining that it could not allocate a vector of the size 155.3 MB. One application did succeed giving 39 covariates in 3 minutes:

```
94 1913 2051 2083 2129 2656 7783 7855 9859 25423 25529 30644 30789
31504 31964 91725 92955 92956 121490 121701 156301 160277 161551
187192 187214 187259 188898 188907 189367 189830 190613 192106 193822
```

```
194153 197373 197578 197585 197823 197834
```

Regressing the dependent variable on these 39 resulted in three NAs. Of the remaining 36 covariates 21 had a regression P-value exceeding 0.1. The sum of squared residuals was 5031.45.

The Gaussian covariate procedure with $\alpha = 0.05$ results in the following 10 interactions with a computing time of 13 seconds. The first interaction is boston$[,6]^3$, the second boston$[,6]^7$boston$[,13]$ and so on. The sum of squared residuals was 5690 compared with 11079 for the linear regression on all 13 covariates.

```
> bostoninter<-fgeninter(boston[,1:13],8)[[1]]
[1] "number of interactions: 203490"
> bostonsv<-fstepwise(boston[,14],bostoninter,0.05,20,offset=F,time=T)[[1]]
   user  system elapsed
 15.300   0.716  16.588
> bostonsv
        [,1]         [,2]
 [1,]    441 0.000000e+00
 [2,] 197063 0.000000e+00
 [3,] 197166 7.455263e-10
 [4,] 120886 1.006557e-05
 [5,] 118685 2.726566e-07
 [6,] 121659 1.140579e-03
 [7,] 193641 1.650804e-02
 [8,]     10 1.607415e-03
 [9,]   7472 4.518208e-11
[10,] 192007 9.425253e-06
> decomp(bostonsv[,1],14,8)
[[1]]
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
```

```
 [1,]    0    0    0    0    0    6    6    6
 [2,]    6    6    6    6    6    6    6   13
 [3,]    6    6    6    6    6   11   11   11
 [4,]    1    4    7    9    9    9    9    9
 [5,]    1    4    5    5    6    7    9    9
 [6,]    1    5    5    5    6    6    8   13
 [7,]    5    6    6    6    6    6    8   10
 [8,]    0    0    0    0    0    0    0    9
 [9,]    0    0    0    5    6    6    9   13
[10,]    5    5    6    6    6    9   13   13
```

```
> b<-lm(boston[,14]~bostoninter[,bostonsv[,1]])
> print(sum(b$res^2))
[1] 5690.012
> b<-lm(boston[,14]~boston[,1:13])
> print(sum(b$res^2))
[1] 11078.78
```

Regressing the dependent variable on these 10 covariates results in the following regression P-values

```
> as.double(summary(b)[[4]][2:11,4])
 [1] 1.519294e-110  2.158545e-07  9.677670e-19  3.369443e-09  3.530752e-07
 [6]  2.279109e-06  9.988516e-09  7.841906e-28  1.991632e-19  6.770982e-11
```

3.5 Graphs

Given covariates $\boldsymbol{x}_j, j = 1, \ldots, k$ a graph is calculated as follows. Each $\boldsymbol{x}_j$ is regressed on the remaining covariates and connected to those covariates found to be relevant

As an example we take $\boldsymbol{x}$ to be a set of covariates generated in Tutorial 1 of

https://web.stanford.edu/group/candes/knockoffs/software/knockoff/

The graph is determined by the matrix Sigma which is a Toeplitz matrix with $\rho = 0.25$. As the elements decrease rapidly in size as one moves away from the diagonal it is effectively a tridiagonal matrix with the side diagonal elements being 0.25. The graph will have edges connecting each $\boldsymbol{x}_j$ to $\boldsymbol{x}_{j+1}$ for $i = j, \ldots, 999$.
One application of lasso resulted in seven false negative and 53 false positives. The time requires was about 150 minutes.

```
> xc<-tut1x(1000,1000)[[1]]
> #
> tmp<-gralss(xc)
user.self
 10902.11
> tmp[[1]]
[1] 1045
> fn<-999-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])==1)
> fp<-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])>=2)
> print(c("(fn,fp)",fn,fp))
[1] "(fn,fp)" "7"       "53"
```

Knockoff gave the following error

```
Fehler in chol.default(Sigma_k) :
  der fuehrende Minor der Ordnung 1 ist nicht positiv definit
```

in English

```
Error in  chol.default(Sigma_k) :
 the leading minor of order 1 is not postive definite.
```

The Gaussian covariate method is applied k times and to account for this the cut-off P-value α is replaced by α/k. The command is

```
fgraphst(x,alpha,kmax,nu=1,time=FALSE,verbose=FALSE,offset=TRUE)
```

Applying fgraphst to the same data set gives

```
> tmp<-fgraphst(xc,0.05,10,time=T)
   user  system elapsed
 19.156   0.000  19.147
> tmp[[1]]
[1] 994
> fn<-999-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])==1)
> fp<-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])>=2)
> print(c("(fn,fp)",fn,fp))
[1] "(fn,fp)" "5"       "0"
> tmp[[2]][1:10,]
      [,1] [,2]
 [1,]    1    2
 [2,]    2    3
 [3,]    3    4
 [4,]    4    5
 [5,]    5    6
 [6,]    6    7
 [7,]    7    8
 [8,]    8    9
 [9,]    9   10
[10,]   10   11
```

tmp[[1]] is the number of edges and tmp[[2]] gives the edges. There are no false positives and 5 false negatives. The time requires was about 16 seconds. In [Meinshausen and Bühlmann, 2006] lasso was used to calculate graphs for large$k$. In a simulation in Section 4 of that paper with $(n, k) = (600, 1000)$ the method resulted in two false positives and 638 false negatives. The description of the generation of the graph of Figure 1 is incorrect and has

been altered to reproduce graphs with about 1800 edges. One application of lasso gave 571 false positives and 13 false negatives. The time taken was 46 minutes.

```
> frgraph(st=F,lasso=T)
[1] 1829
user.self
  3036.04
[1] 571  13
    user   system  elapsed
3036.728    0.032 3034.852
```

For the same data the Gaussian covariate procedure with $\alpha = 0.05$ resulted in zero false positives and 118 false negatives.

```
> frgraph()
[1] 1829
[1]   0 118
   user  system elapsed
 18.800   0.000  18.797
```

The default setting of fgraphst is to divide α by the number of variables k so that the probability of one false positive edge is at most α. If this is judged to be too severe at the cost of false negatives the value of α can be increased to give so to speak a specified expected number of false positives. Putting $\alpha = 5$ in fgraphst with $k = 1000$ increases the expected number of false positives to about $1000 * (0.05 * 100/1000) = 5$. In the simulation just reported this results in 5 false positives and 12 false negatives.

```
> frgraph(alpha=5)
[1] 1829
[1]  5 12
   user  system elapsed
 21.284   0.000  21.271
```

Graphs can also be constructed for real data sets. The colon cancer data with $(n, k) = (62, 2000)$. Lasso requires 10 minutes and produces a graph with 23322 edges.

```
> tmp<-gralss(colon.x)
user.self
  703.448
> tmp[[1]]
[1] 23322
```

The Gaussian covariate method with $\alpha = 0.05$ gives a graph with 1634 edges in 2.8 seconds. The first ten edges are given.

```
> tmp<-fgraphst(colon.x,0.05,10,time=T)
   user  system elapsed
  2.772   0.000   2.769
> tmp[[1]]
[1] 1634
> tmp[[2]][1:10,]
      [,1] [,2]
 [1,]    1   23
 [2,]    1   98
 [3,]    2    3
 [4,]    2  126
 [5,]    2  156
 [6,]    4  108
 [7,]    4  131
 [8,]    4  367
 [9,]    5   15
[10,]    5 1201
```

The repeated Gaussian covariate method with $\alpha = 0.05$ gives a graph with 24475 edges in 69 seconds.

```
> tmp<-fgraphstst(colon.x,0.05,10,nedge=1e5,time=T)
   user  system elapsed
 62.068   0.012  62.044
> tmp[[1]]
[1] 24475
```

Putting $\alpha = 1e-7$ gives a graph with 1521 edges. The first 10 are

```
     [,1] [,2]
 [1,]    2    3
 [2,]    2  126
 [3,]    2  156
 [4,]    2  162
 [5,]    3  126
 [6,]    4  108
 [7,]    4  131
 [8,]    4  367
 [9,]    4  416
[10,]    4  457
[11,]    4  697
[12,]    4  864
[13,]    4 1070
[14,]    5   15
```

It is seen that the repeated method picks up many more highly significant dependencies than the single method so the recommendation would seem to be to use the repeated method with a smaller value of α.
Finally for the osteoarthritis data with $(n, k) = (129, 48802)$ lasso requires 11 seconds for one node resulting in an estimated 11*48802/(24*3600)=6.2 days for the whole graph. The Gaussian covariate method requires about 0.24 seconds for each node giving an estimated time of 3 hours 15 minutes

for the whole graph. A specially written Fortran programme gives a graph with 38986 edges in 52 minutes.

```
> tmp<-fgraphst(gse3.x,0.05,10,time=T)
    user   system  elapsed
2941.296    0.052 2939.496
> tmp[[1]]
[1] 38986
```

This data set shows the limitations of the recommendation made just above. Regress the first covariate on the remaining 44801 where α has been divided by the number of variables as in the default version of fgraphst. This results in 4009 selected covariates. If a graph were to be constructed the first covariate alone would be connected to 4009 other covariates. The calculated P-values for the first 152 covariates are zero. Such a graph would be much too large to be useful.

```
> tmp<-fstepstepwise(gse3.x[,1],gse3.x[,2:48802],alpha=0.05/48801,10,time=TRUE)[[1]]
user.self
  744.776
> length(tmp[,1])
[1] 4009
```

In this particular case one can make use of the problem and restrict the construction of the graph to those covariates chosen in the first step as in Section 3.3.3. We take the 74 covariates selected by the Gaussian covariate method and apply the repeated Gaussian procedure to calculate a graph with $\alpha = 1e-6$. It has 517 edges. The first covariate 11499 is connected with 18 of the 74 covariates.

```
> tmp<-fgraphstst(gse3.x[,stost],1e-6,10,time=T)
   user  system elapsed
  0.096   0.000   0.095
```

```
> tmp[[1]]
[1] 517
      [,1]  [,2]
 [1,] 11499  3630
 [2,] 11499 41799
 [3,] 11499 26207
 [4,] 11499 10546
 [5,] 11499 46979
 [6,] 11499 25241
 [7,] 11499 34803
 [8,] 11499   939
 [9,] 11499 10374
[10,] 11499  7896
[11,] 11499 39141
[12,] 11499 26795
[13,] 11499 44664
[14,] 11499  6740
[15,] 11499 28816
[16,] 11499 24232
[17,] 11499 29522
[18,] 11499 25125
[19,] 31848 44902
[20,] 31848 25241
```

4 Acknowledgment

We thank Oliver Maclaren for comments on earlier versions.

References

[Alizadeh et al., 2000] Alizadeh, A., Eisen, M., Davis, R., Ma, C., Lossos, I., Rosenwald, A., Boldrick, J., Sabet, H., Tran, T., Yu, X., Powell, J., Yang, L., Marti, G., Moore, T., Hudson, J., Lu, L., Lewis, D., Tibshirani, R., Sherlock, G., Chan, W., Greiner, T., Weisenburger, D., Armitage, J., Warnke, R., Levy, R., Wilson, W., Grever, M., Byrd, J., Botstein, D., Brown, P., and Staudt, L. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511.

[Alon et al., 1999] Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., and Levine, A. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS*, 96(12).

[Bellec et al., 2017] Bellec, P. C., Lecué, G., and Tsybakov, A. B. (2017). Slope meets Lasso: improved oracle bounds and optimality. arXiv:1605.08651v3 [math.ST].

[Candes et al., 2017] Candes, E., Fan, Y., Janson, L., and Lv, J. (2017). Panning for gold: Model-free knockoffs for high-dimensional controlled variable selection. arXiv:1610.02351v3 [math.ST].

[Cortez et al., 2009] Cortez, P., Cerdeira, A., Almeida, F., Matos, T., and Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems, Elsevier*, 47(4):547–553.

[Cox and Battey, 2017] Cox, D. R. and Battey, H. S. (2017). Large numbers of explanatory variables, a semi-descriptive analysis. *Proc. Natl. Acad. Sci. USA*, 114(32):85928595.

[Davies, 2014] Davies, L. (2014). *Data Analysis and Approximate Models*. Monographs on Statistics and Applied Probability 133. CRC Press.

[Davies, 2017] Davies, L. (2017). Stepwise choice of covariates in high dimensional regression. arXiv:1610.05131v4 [math.ST].

[Davies and Dümbgen,] Davies, L. and Dümbgen, L. A model-free approach to linear least squares regression with exact probabilities. In preparation.

[Dettling and Bühlmann, 2003] Dettling, M. and Bühlmann, P. (2003). Boosting for tumor classification with gene expression data. *Bioinformatics*, 19(9):1061–1069.

[Golub et al., 1999] Golub, T., Slonim, D., P., T., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C., and Lander, E. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(15):531–537.

[Lockhart et al., 2014] Lockhart, R., Taylor, J., Tibshirani, R. J., and Tibshirani, R. (2014). A significance test for the lasso. *Ann. Statist.*, 42(2):413–468.

[Meinshausen and Bühlmann, 2006] Meinshausen, N. and Bühlmann, P. (2006). High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462.

[R Development Core Team, 2008] R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, 58(1):267–288.

5 Appendix: Results of runcomp.R

```
> library(knockoff)
> library(glmnet)
> library(quantreg)
> #
> load("comp_data.rda")
> #load("gse3x.rda")
> #load("gsey.rda")
> source("selvar.R")
> source("comp.R")
> dyn.load("selvar.so")
> time1<-proc.time()
> #
> #    SECTION 2.2
> #
> fstepwise(ly.original,lx.original,0.05,10,time=T)[[1]]
   user  system elapsed
  0.008   0.000   0.010
     [,1]         [,2]
[1,] 1182 0.0000000000
[2,] 1219 0.0008577131
[3,] 2888 0.0035805523
```

```
> #
> print("Table 1")
[1] "Table 1"
> #
> tmp<-fsimords(1000,1000,0.05,10,1000)
> tmp[[1]]
[1]  1  2  3  4  5  6  7  8 10
> tmp[[2]]
[1] 0 1 1 2 3 3 4 5 6
> #
> tmp<-fsimords(1000,1000,0.01,10,1000)
> tmp[[1]]
[1] 1 1 2 3 4 5 6 7 7
> tmp[[2]]
[1] 0 0 1 1 2 2 3 4 4
> #
> #
> #
> fstepwise(ly.original,lx.original,0.05,10,nu=3,time=T)[[1]]
   user  system elapsed
  0.016   0.000   0.016
     [,1]         [,2]
[1,] 1182 0.000000e+00
[2,] 1219 1.051452e-10
[3,] 2888 7.664817e-09
[4,] 1946 3.353905e-03
[5,] 2102 6.026398e-04
> #
> b<-lm(ly.original~lx.original[,c(1182,1219,2888,1946,2102)])
> as.double(summary(b)[[4]][2:6,4])
```

```
[1] 2.286646e-15 4.207564e-10 1.549639e-06 1.041936e-06 4.478674e-05
> #
> set.seed(2345)
> tmpx<-rnorm(lx.original)
> dim(tmpx)<-dim(lx.original)
> tmpx[,1:3]<-lx.original[, c(1182, 1219, 2888)]
> fstepwise(ly.original,tmpx,0.05,10,nu=3)[[1]]
     [,1]         [,2]
[1,]    1 0.000000e+00
[2,]    2 1.051452e-10
[3,]    3 7.664817e-09
[4,] 1698 5.235440e-03
> b<-lm(ly.original~tmpx[,c(1:3,1698)])
> as.double(summary(b)[[4]][2:5,4])
[1] 3.630659e-21 1.242047e-08 9.651025e-07 9.631336e-05
> rm(tmpx)
> #
> #
> #  SECTION 2.4
> #
> #
> fstepwise(colon.y,colon.x,1,2,time=T)[[1]]
   user  system elapsed
  0.004   0.000   0.002
     [,1]         [,2]
[1,]  493 7.402367e-08
[2,]  175 4.311166e-01
> #
> #
> tmp<-fstepstepwise(colon.y,colon.x,0.05,10,time=T)[[1]]
```

```
user.self
    0.188
> stcolon<-sort(tmp[,2])
> tmp
       [,1] [,2]         [,3]
 [1,]    1  493 7.402367e-08
 [2,]    2  377 1.355905e-07
 [3,]    3  249 1.134563e-06
 [4,]    4 1635 2.283614e-06
 [5,]    4  576 2.815082e-02
 [6,]    5 1423 2.760755e-05
 [7,]    5  353 7.996075e-04
 [8,]    6  625 3.166107e-05
 [9,]    6  739 8.354274e-04
[10,]    7  245 1.931080e-04
[11,]    7 1772 1.716390e-02
[12,]    8 1771 2.260242e-04
[13,]    8 1897 1.455075e-03
[14,]    9  765 3.477144e-04
[15,]    9 1346 2.734873e-02
[16,]   10  267 3.972433e-04
[17,]   11 1884 5.095652e-04
[18,]   12 1843 5.866773e-04
[19,]   12 1549 4.759877e-04
[20,]   13  897 5.955879e-04
[21,]   14   66 7.261918e-04
[22,]   14 1993 4.473896e-02
[23,]   15 1494 8.133834e-04
[24,]   15  228 3.064624e-02
[25,]   16  822 1.351272e-03
```

```
[26,]   17 1582 1.386727e-03
[27,]   17  663 1.746740e-02
[28,]   18 1730 1.476773e-03
[29,]   19  513 1.690501e-03
[30,]   19 1210 4.037406e-04
[31,]   20 1042 1.878823e-03
[32,]   20   14 3.475249e-03
[33,]   21  812 3.459078e-03
[34,]   21  187 1.345395e-02
[35,]   22  137 3.674447e-03
[36,]   22  627 5.586127e-03
[37,]   23  138 4.232452e-03
[38,]   23 1231 9.372632e-03
[39,]   24 1406 4.979419e-03
[40,]   24  823 5.203994e-03
[41,]   25 1060 5.208778e-03
[42,]   25  411 8.464091e-03
[43,]   26  780 5.390096e-03
[44,]   27  964 6.032253e-03
[45,]   28  365 6.171878e-03
[46,]   28  611 5.032157e-03
[47,]   29  391 6.241313e-03
[48,]   29 1873 3.182333e-03
[49,]   30  824 6.930950e-03
[50,]   30 1400 4.309922e-02
[51,]   31 1900 8.925496e-03
[52,]   32 1256 1.244725e-02
[53,]   32  892 1.598585e-03
[54,]   33  802 1.401268e-02
[55,]   34 1648 1.604443e-02
```

```
[56,]   34  189 9.831067e-04
[57,]   35 1153 1.645393e-02
[58,]   35  792 7.972154e-04
[59,]   36   67 1.726898e-02
[60,]   36  989 7.152085e-03
[61,]   37  992 1.934046e-02
[62,]   37  140 8.214284e-03
[63,]   38 1674 2.088915e-02
[64,]   39 1892 2.088136e-02
[65,]   40 1293 2.138831e-02
[66,]   40  229 1.104997e-03
[67,]   41 1634 2.475598e-02
[68,]   41 1981 2.595704e-03
[69,]   42 1511 2.519410e-02
[70,]   43   26 2.808292e-02
[71,]   43  211 2.133406e-02
[72,]   44  581 2.882589e-02
[73,]   44   49 3.194865e-02
[74,]   45 1867 3.050095e-02
[75,]   46  111 3.316820e-02
[76,]   46  622 3.071545e-02
[77,]   47 1671 3.352217e-02
[78,]   47 1473 4.698842e-02
[79,]   48 1960 4.191835e-02
[80,]   48  293 9.836906e-03
[81,]   49 1679 4.885719e-02
[82,]   49 1451 1.181720e-02
> #
> #
> # Section 2.5
```

```
> #
> #
> fl1stack(stackloss,1000)
[1] 0.2040000 0.3440461
> #
> #
> # Section 3.1
> #
> #
> set.seed(1234)
> ftut(1,1000,1000,60,4.5,0.1,0.05,50)
    user   system  elapsed
4029.124    9.236 4036.249
[1] 82.40  0.52  7.94
[1]  0.000 46.160  0.256
[1]  3.36 11.60  2.35
[1] 7.08 5.64 3.19
[1]  5.58 10.02 64.88
> #
> set.seed(1234)
> ftut(2,1000,1000,60,7.5,0.2,0.05,50)
    user   system  elapsed
3360.472   11.544 3370.011
[1] 56.02 15.86  9.91
[1]  0.0000 56.5200  0.0418
[1]  2.780 42.460  0.449
[1]  7.000 35.400  0.957
[1]  7.0 35.1 54.2
> #
> #
```

```
> # Section 3.2
> #
> #
> fstepwise(redwine[,12],redwine[,1:11],0.05,10)[[1]]
     [,1]         [,2]
[1,]   11 0.000000e+00
[2,]    2 0.000000e+00
[3,]   10 2.032414e-10
[4,]    7 1.027568e-04
[5,]    5 1.002491e-04
[6,]    9 1.096799e-03
> #
> fstepstepwise(redwine[,12],redwine[,1:11],0.05,10)[[1]]
      [,1] [,2]         [,3]
 [1,]    1   11 0.000000e+00
 [2,]    1    2 0.000000e+00
 [3,]    1   10 2.032414e-10
 [4,]    1    7 1.027568e-04
 [5,]    1    5 1.002491e-04
 [6,]    1    9 1.096799e-03
 [7,]    2    3 0.000000e+00
 [8,]    2    8 0.000000e+00
 [9,]    2    1 1.918132e-12
[10,]    2    4 4.110445e-07
[11,]    3    6 4.283397e-02
> #
> fredwine(redwine,200,alpha=0.05)
[1] "results"
[1] 0.035 1.260
[1] 0.05 0.00
```

```
[1] 0.455 0.000
[1] 1.445 0.000
[1] 1.15 0.03
> #
> #
> # Section 3.3.1
> #
> #
> set.seed(1234)
> tmp<-cv.glmnet(colon.x,colon.y)
> tmplas1<-(1:2000)[abs(coef(tmp)[2:2001,1])>0]
> tmplas1
 [1]   14  249  377  493  576  625  792 1231 1360 1473 1582 1679 1772 1843 1924
> tmp<-cv.glmnet(colon.x,colon.y)
> tmplas1<-(1:2000)[abs(coef(tmp)[2:2001,1])>0]
> tmplas1
[1] 249 377 493 625
> b<-lm(colon.y~colon.x[,tmplas1])
> as.double(summary(b)[[4]][2:5,4])
[1] 0.025170468 0.003786664 0.580520055 0.018821024
> b<-lm(colon.y~colon.x[,tmplas1[c(1,2,4)]])
> as.double(summary(b)[[4]][2:4,4])
[1] 4.712717e-03 8.257605e-05 1.097590e-02
> b<-lm(colon.y~colon.x[,tmplas1[c(1,2)]])
> as.double(summary(b)[[4]][2:3,4])
[1] 7.772988e-05 9.095636e-06
> #
> b<-lm(colon.y~colon.x[,c(249,377,493,625)])
> ss1234<-sum(b$res^2)
> b<-lm(colon.y~colon.x[,c(249,377,625)])
```

```
> ss124<-sum(b$res^2)
> pval3<-pbeta(1-ss1234/ss124,0.5,(62-5)/2)
> pval3<-1-pbeta(pval3,2001-5,1)
> pval3
[1] 1
> #
> b<-lm(colon.y~colon.x[,c(249,377)])
> ss12<-sum(b$res^2)
> b<-lm(colon.y~colon.x[,377])
> ss1<-sum(b$res^2)
> pval1<-pbeta(1-ss12/ss1,0.5,(62-3)/2)
> pval1<-1-pbeta(pval1,2001-3,1)
> pval1
[1] 0.1438506
> b<-lm(colon.y~colon.x[,249])
> ss2<-sum(b$res^2)
> pval2<-pbeta(1-ss12/ss2,0.5,(62-3)/2)
> pval2<-1-pbeta(pval2,2001-3,1)
> pval2
[1] 0.01800903
> #
> b<-lm(colon.x[,493]~colon.x[,c(249,377)])
> as.double(summary(b)[[4]][2:3,4])
[1] 1.803856e-06 3.452139e-08
> #
> #
> # lacolon and kncolon
> #
> set.seed(1234)
> tmpcol<-fvarch(colon.y,colon.x)
```

```
[1] "lasso start"
[1] 15
 [1]   14  249  377  493  576  625  792 1231 1360 1473 1582 1679 1772 1843 1924
[1] 4
[1] 249 377 493 625
[1] 5
[1]  249  377  493  625 1772
[1] 25
 [1]   14  164  175  249  353  377  493  576  788  792 1073 1094 1221 1231 1346
[16] 1360 1400 1473 1549 1582 1668 1679 1772 1843 1924
[1] 29
 [1]   14  164  175  353  377  576  611  654  788  792  823 1073 1094 1221 1231
[16] 1256 1346 1360 1400 1473 1549 1570 1582 1668 1679 1772 1843 1873 1924
   user  system elapsed
  1.644   0.000   1.641
 [1]   14  164  175  249  353  377  493  576  611  625  654  788  792  823 1073
[16] 1094 1221 1231 1256 1346 1360 1400 1473 1549 1570 1582 1668 1679 1772 1843
[31] 1873 1924
[1] "lasso end"
[1] "knockoff start"
[1] 1
[1] 12
 [1]   14  175  353  788  792  823 1360 1400 1570 1740 1843 1924
[1] 2
[1] 3
[1]  377  792 1772
[1] 3
[1] 6
[1]  175  377  792 1772 1843 1924
[1] 4
```

```
[1] 35
 [1]   14  164  175  353  576  611  654  788  792  823  966 1073 1094 1110 1123
[16] 1146 1231 1241 1346 1360 1400 1420 1482 1549 1570 1622 1649 1827 1843 1873
[31] 1893 1924 1935 1976 1989
[1] 5
[1] 4
[1]  353  792 1843 1924
 [1]   14  164  175  353  377  576  611  654  788  792  823  966 1073 1094 1110
[16] 1123 1146 1231 1241 1346 1360 1400 1420 1482 1549 1570 1622 1649 1740 1772
[31] 1827 1843 1873 1893 1924 1935 1976 1989
    user   system  elapsed
6595.920    9.940 6601.908
[1] "knockoff end"
> lacolon<-tmpcol[[1]]
> kncolon<-tmpcol[[2]]
> #load("lacolon.rda")
> #load("kncolon.rda")
> # Lasso
> fpval(colon.y,colon.x,lacolon,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,] 1843 6.495852e-09
 [2,]  493 7.398644e-08
 [3,] 1772 1.078545e-07
 [4,]  377 1.355903e-07
 [5,]  249 1.135130e-06
 [6,] 1873 1.214555e-06
 [7,] 1231 2.822242e-06
```

```
 [8,] 1473 4.105941e-06
 [9,]   14 6.468307e-06
[10,] 1400 6.848481e-06
[11,] 1360 1.044163e-05
[12,]  576 1.516268e-05
[13,] 1256 1.737674e-05
[14,] 1679 3.054090e-05
[15,]  625 3.175638e-05
[16,] 1924 4.711134e-05
[17,] 1549 3.631080e-04
[18,] 1346 4.018934e-04
[19,] 1582 1.403567e-03
[20,]  792 3.259051e-03
[21,]  353 3.770271e-03
[22,]  175 4.906957e-03
[23,] 1668 8.633158e-03

> # Knockoff
> fpval(colon.y,colon.x,kncolon,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,] 1843 6.495852e-09
 [2,]  377 1.355903e-07
 [3,]   14 2.778769e-07
 [4,] 1873 1.214555e-06
 [5,] 1772 2.463932e-06
 [6,] 1231 2.822242e-06
 [7,] 1360 1.044163e-05
```

```
 [8,]  576 1.516268e-05
 [9,] 1400 1.168938e-04
[10,] 1935 1.411647e-04
[11,] 1549 3.631080e-04
[12,] 1346 4.018934e-04
[13,] 1924 4.747120e-04
[14,]  175 4.906957e-03
[15,] 1110 8.805362e-03

> # Repeated Gaussian stepwise
> fpval(colon.y,colon.x,stcolon,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,] 1423 1.106895e-10
 [2,] 1843 6.495852e-09
 [3,] 1494 2.047067e-08
 [4,]  493 7.398644e-08
 [5,] 1293 8.035832e-08
 [6,] 1772 1.078545e-07
 [7,]  377 1.355903e-07
 [8,]  513 1.999018e-07
 [9,] 1406 2.378230e-07
[10,]  992 4.819964e-07
[11,] 1634 5.697313e-07
[12,]  249 1.135130e-06
[13,] 1648 1.161793e-06
[14,] 1060 1.316599e-06
[15,] 1042 1.476826e-06
```

```
[16,] 1730 2.032950e-06
[17,] 1900 2.209020e-06
[18,] 1635 2.285901e-06
[19,]  365 2.311574e-06
[20,]  187 2.586279e-06
[21,] 1231 2.822242e-06
[22,]  625 4.630786e-06
[23,]  792 4.863052e-06
[24,]  111 4.885140e-06
[25,]   14 6.468307e-06
[26,] 1771 9.208768e-06
[27,]  138 1.209879e-05
[28,]  576 1.516268e-05
[29,] 1256 1.737674e-05
[30,] 1153 2.372798e-05
[31,]  137 2.812837e-05
[32,] 1679 3.054090e-05
[33,]  822 4.699079e-05
[34,]  391 5.023164e-05
[35,] 1400 6.746418e-05
[36,]  211 1.127294e-04
[37,] 1346 1.912821e-04
[38,] 1897 1.919767e-04
[39,]  245 1.938838e-04
[40,]  765 3.498139e-04
[41,]  353 3.837240e-04
[42,]  267 4.000444e-04
[43,] 1210 4.092701e-04
[44,] 1549 4.798292e-04
[45,]  189 4.816914e-04
```

```
[46,] 1884 5.134168e-04
[47,]   67 5.506242e-04
[48,]  897 6.009980e-04
[49,]  663 7.152961e-04
[50,]   66 7.331577e-04
[51,] 1473 7.841344e-04
[52,] 1960 7.868915e-04
[53,]  739 8.379427e-04
[54,]  892 9.096631e-04
[55,]  581 9.850568e-04
[56,]  229 1.140953e-03
[57,]  140 1.241931e-03
[58,]   26 1.286887e-03
[59,] 1582 1.403567e-03
[60,] 1981 2.682871e-03
[61,] 1873 3.255539e-03
[62,]  812 3.513470e-03
[63,]  989 4.410411e-03
[64,]  611 5.064780e-03
[65,]  823 5.301922e-03
[66,]  780 5.502652e-03
[67,]  627 5.679673e-03
[68,]  964 6.161314e-03
[69,]  411 6.684128e-03
[70,]  824 7.097239e-03
[71,]  228 8.232653e-03

> #
> #
> # Section 3.3.2
```

```
> #
> #
> laleuk<-fvarch(ly.original,lx.original,kn=F)[[1]]
[1] "lasso start"
[1] 12
 [1]  456  626  672  956  979 1182 1219 1652 1946 2481 3098 3441
[1] 24
 [1]  219  456  626  657  672  888  956  979 1099 1108 1182 1219 1620 1652 1946
[16] 2230 2239 2481 2537 2727 2859 2888 3098 3158
[1] 14
 [1]  456  626  672  956  979 1182 1219 1652 1946 2230 2481 3098 3158 3441
[1] 24
 [1]  219  456  626  657  672  888  956  979 1099 1182 1219 1620 1652 1946 2230
[16] 2239 2481 2537 2727 2859 2888 3098 3158 3345
[1] 11
 [1]  456  626  672  956  979 1182 1219 1652 1946 2481 3441
   user  system elapsed
  2.576   0.000   2.574
 [1]  219  456  626  657  672  888  956  979 1099 1108 1182 1219 1620 1652 1946
[16] 2230 2239 2481 2537 2727 2859 2888 3098 3158 3345 3441
[1] "lasso end"
> laleuk
 [1]  219  456  626  657  672  888  956  979 1099 1108 1182 1219 1620 1652 1946
[16] 2230 2239 2481 2537 2727 2859 2888 3098 3158 3345 3441
> fpval(ly.original,lx.original,laleuk,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,]  456 0.000000e+00
```

```
 [2,]  956 0.000000e+00
 [3,]  979 0.000000e+00
 [4,] 1182 0.000000e+00
 [5,] 1652 0.000000e+00
 [6,] 2481 0.000000e+00
 [7,] 3441 0.000000e+00
 [8,] 1099 5.548895e-12
 [9,]  626 3.804956e-11
[10,] 1219 5.152545e-11
[11,] 2230 2.852119e-10
[12,]  672 6.273695e-07
[13,]  219 3.663922e-06
[14,] 1946 2.117592e-05
[15,] 2727 4.329760e-05
[16,]  657 4.817129e-05
[17,] 1620 5.316102e-05
[18,] 3158 1.001275e-04
[19,] 3098 1.013139e-03
[20,] 2239 1.258528e-03
[21,] 2537 1.984932e-03
[22,] 3345 3.509971e-03
[23,] 2888 3.579551e-03
[24,] 2859 6.230513e-03
[25,]  888 7.516447e-03

> stleuk<-fstepstepwise(ly.original,lx.original,0.000001,10,time=T)[[1]]
user.self
    0.192
> stleuk<-stleuk[,2]
> print(length(stleuk))
```

```
[1] 41
> fpval(ly.original,lx.original,stleuk,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,] 1182 0.000000e+00
 [2,] 1652 0.000000e+00
 [3,]  979 0.000000e+00
 [4,]  956 0.000000e+00
 [5,] 2481 0.000000e+00
 [6,] 3441 0.000000e+00
 [7,]  456 0.000000e+00
 [8,]  874 0.000000e+00
 [9,] 2789 0.000000e+00
[10,]  435 0.000000e+00
[11,] 1356 0.000000e+00
[12,] 2049 0.000000e+00
[13,]  436 3.963496e-13
[14,] 1099 5.548895e-12
[15,] 3038 7.134293e-12
[16,]  626 3.804956e-11
[17,] 1219 5.152545e-11
[18,] 3216 2.413769e-10
[19,]  907 5.513223e-10
[20,] 3162 6.512024e-10
[21,] 2198 2.000377e-09
[22,] 2226 2.471636e-09
[23,] 2079 6.085156e-09
[24,]  918 7.884187e-09
```

```
[25,] 2220 1.093568e-08
[26,] 2230 1.228208e-08
[27,] 2145 1.499945e-08
[28,] 1249 2.457288e-08
[29,]  951 4.670623e-08
[30,] 2141 6.791133e-08
[31,]  851 8.359766e-08
[32,] 1014 1.493699e-07
[33,] 2546 2.594952e-07
[34,] 1053 2.742668e-07
[35,] 2911 3.024876e-07
[36,] 1104 3.150270e-07
[37,] 1001 5.385431e-07
[38,] 2449 6.480607e-07
[39,] 1020 8.243764e-07
[40,] 3466 9.369273e-07
[41,]  990 9.641691e-07

> #
> #
> # Section 3.3.3
> #
> #
> laprost<-fvarch(prostate.y,prostate.x,kn=F)[[1]]
[1] "lasso start"
[1] 18
 [1] 1291 1735 1788 1839 1848 1903 2003 2377 2388 2450 2619 3423 4279 4288 4898
[16] 5016 5035 5663
[1] 3
[1] 1839 2619 5016
```

```
[1] 6
[1] 1839 2003 2619 3423 5016 5035
[1] 14
 [1] 1291 1735 1839 1848 1903 2003 2450 2619 3423 4288 4898 5016 5035 5663
[1] 14
 [1] 1291 1735 1839 1848 1903 2003 2450 2619 3423 4288 4898 5016 5035 5663
   user  system elapsed
  5.344   0.000   5.339
 [1] 1291 1735 1788 1839 1848 1903 2003 2377 2388 2450 2619 3423 4279 4288 4898
[16] 5016 5035 5663
[1] "lasso end"
> fpval(prostate.y,prostate.x,laprost,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,] 1839 0.000000e+00
 [2,] 2619 0.000000e+00
 [3,] 4288 0.000000e+00
 [4,] 5016 0.000000e+00
 [5,] 5035 0.000000e+00
 [6,] 3423 2.008727e-12
 [7,] 4898 1.936399e-07
 [8,] 2377 4.988987e-06
 [9,] 1788 1.328393e-04
[10,] 1903 2.919642e-04
[11,] 2388 3.082575e-04
[12,] 2003 4.721405e-04

> stprost<-fstepstepwise(prostate.y,prostate.x,0.00001,10,time=T)[[1]]
```

```
user.self
    0.316
> stprost<-stprost[,2]
> print(length(stprost))
[1] 33
> fpval(prostate.y,prostate.x,stprost,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
      [,1]         [,2]
 [1,] 2619 0.000000e+00
 [2,] 5016 0.000000e+00
 [3,] 1839 0.000000e+00
 [4,] 4701 0.000000e+00
 [5,] 4155 0.000000e+00
 [6,] 4212 0.000000e+00
 [7,] 3705 0.000000e+00
 [8,] 2746 0.000000e+00
 [9,] 5035 0.000000e+00
[10,] 3392 0.000000e+00
[11,] 3006 0.000000e+00
[12,] 4898 6.694645e-13
[13,] 1640 6.695755e-13
[14,] 4335 1.339151e-12
[15,] 5808 4.017453e-12
[16,] 3934 9.374057e-12
[17,] 3423 1.205236e-11
[18,] 3833 3.883538e-11
[19,] 2425 8.838397e-11
[20,] 4261 3.722840e-10
```

```
[21,] 4001 3.248780e-09
[22,] 5249 4.436541e-09
[23,] 5230 8.823666e-09
[24,] 5639 2.574384e-08
[25,] 4849 2.764811e-08
[26,] 2386 7.471659e-08
[27,] 5783 8.060352e-08
[28,] 1540 1.117588e-07
[29,] 2428 1.809616e-07
[30,] 2377 1.011739e-06
[31,] 4448 1.288842e-06
[32,] 3366 3.940713e-06
[33,] 4262 7.863072e-06

> #
> #
> #
> laost<-fvarch(gse.y,gse3.x,kn=F)[[1]]
[1] "lasso start"
[1] 43
 [1]   939  3630  3685  4216  5210  5939  6661  8669 11499 13650 14724 17062
[13] 19391 21667 21757 23038 23259 23796 23983 24232 24462 24765 26207 26917
[25] 29522 30451 30997 33321 35023 36417 38406 38561 41045 41799 43642 43951
[37] 44542 44758 44902 45991 46172 46700 48035
[1] 50
 [1]   939  1348  3630  3685  4216  5210  5939  6661  8669 11499 13650 14724
[13] 17062 19391 20877 21667 21757 21980 23038 23259 23796 23983 24232 24462
[25] 24765 26207 26917 29522 30451 30997 31218 33321 35023 36417 38406 38539
[37] 38561 41045 41799 42887 43642 43951 44542 44758 44902 45627 45991 46172
[49] 46700 48035
```

```
[1] 23
 [1]   939  3630  6661 11499 13650 14724 17062 21667 23038 23259 23983 26207
[13] 33321 35023 36417 38561 41045 41799 43951 44758 44902 45991 48035
[1] 29
 [1]   939  3630  3685  6661 11499 13650 14724 17062 21667 23038 23259 23983
[13] 24462 26207 29522 33321 35023 36417 38406 38561 41045 41799 43642 43951
[25] 44542 44758 44902 45991 48035
[1] 40
 [1]   939  3630  3685  5210  5939  6661  8669 11499 13650 14724 17062 19391
[13] 21667 21757 23038 23259 23796 23983 24232 24462 24765 26207 29522 30451
[25] 30997 33321 35023 36417 38406 38561 41045 41799 43642 43951 44542 44758
[37] 44902 45991 46172 48035
   user  system elapsed
 51.264  17.984  69.215
 [1]   939  1348  3630  3685  4216  5210  5939  6661  8669 11499 13650 14724
[13] 17062 19391 20877 21667 21757 21980 23038 23259 23796 23983 24232 24462
[25] 24765 26207 26917 29522 30451 30997 31218 33321 35023 36417 38406 38539
[37] 38561 41045 41799 42887 43642 43951 44542 44758 44902 45627 45991 46172
[49] 46700 48035
[1] "lasso end"
> fpval(gse.y,gse3.x,laost,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
       [,1]         [,2]
 [1,]   939 0.000000e+00
 [2,]  3630 0.000000e+00
 [3,] 11499 0.000000e+00
 [4,] 44902 5.417777e-12
 [5,] 26207 7.043111e-11
```

```
 [6,] 43951 4.009237e-10
 [7,] 29522 6.176266e-10
 [8,] 41799 1.246114e-09
 [9,] 23983 2.936435e-09
[10,] 24232 4.624073e-08
[11,] 38561 8.087823e-08
[12,] 30451 1.315761e-07
[13,] 33321 5.230429e-07
[14,] 23259 8.717633e-07
[15,] 35023 1.555876e-06
[16,]  8669 2.285539e-06
[17,] 44758 5.331563e-06
[18,] 23038 6.508366e-06
[19,] 46172 9.159328e-06
[20,] 45991 1.382485e-05
[21,]  3685 3.096310e-05
[22,] 13650 9.284205e-05
[23,] 21757 2.463220e-04
[24,] 30997 3.918151e-04
[25,]  5939 4.768018e-04
[26,] 36417 6.450063e-04
[27,]  4216 9.412395e-04
[28,] 14724 1.664579e-03
[29,]  6661 2.026067e-03
[30,] 26917 3.145650e-03
[31,] 46700 3.281872e-03
[32,] 24462 4.818816e-03
[33,] 48035 4.884352e-03
[34,] 17062 4.888099e-03
[35,]  5210 8.442471e-03
```

```
> #
> # Cox-Battey covariates
> #
> coxbat<-c(7235, 11643, 25125, 25470, 25744, 27642, 27920, 29679, 33385,
+ 36409, 37443, 44276, 45991, 46771, 48415, 48433, 48549)
> fpval(gse.y,gse3.x,coxbat,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
       [,1]         [,2]
 [1,] 37443 2.925660e-10
 [2,] 27642 3.684164e-10
 [3,] 33385 2.505773e-08
 [4,] 27920 1.694824e-07
 [5,] 36409 5.585733e-07
 [6,] 25744 7.581411e-07
 [7,] 48415 1.663886e-06
 [8,] 46771 2.609815e-06
 [9,] 25470 3.000447e-06
[10,]  7235 3.314328e-06
[11,] 25125 2.407158e-05
[12,] 48549 1.922365e-04
[13,] 29679 1.984815e-04
[14,] 45991 3.831023e-04
[15,] 48433 6.429226e-04

> #
> stost<-fstepstepwise(gse.y,gse3.x,0.0001,10,time=T)[[1]]
user.self
```

```
    4.084
> stost<-stost[,2]
> fpval(gse.y,gse3.x,stost,0.01)
[1] "start 2"
[1] "start 3"
[[1]]
       [,1]         [,2]
 [1,] 11499 0.000000e+00
 [2,] 31848 0.000000e+00
 [3,] 44902 0.000000e+00
 [4,] 10546 0.000000e+00
 [5,] 34803 0.000000e+00
 [6,] 30451 0.000000e+00
 [7,]   939 0.000000e+00
 [8,]  7896 0.000000e+00
 [9,] 22705 0.000000e+00
[10,] 37443 0.000000e+00
[11,] 28816 0.000000e+00
[12,] 29522 0.000000e+00
[13,] 26207 5.417888e-12
[14,]  3630 1.625333e-11
[15,] 46979 2.167155e-11
[16,] 25744 5.417888e-11
[17,] 10374 9.210221e-11
[18,] 43951 4.009237e-10
[19,] 25125 6.393108e-10
[20,] 41799 1.647038e-09
[21,] 39141 2.557243e-09
[22,] 24232 2.806466e-09
[23,] 26795 4.930278e-09
```

```
[24,]  6740 5.791723e-09
[25,] 44664 6.095124e-09
[26,] 25241 1.106875e-08
[27,] 21350 1.783677e-07
[28,] 24266 8.452877e-07
[29,] 38561 1.122988e-06
[30,] 22929 2.326038e-06
[31,] 44758 5.331563e-06
[32,] 34076 8.007249e-05

> #
> #
> # Section 3.4
> #
> #
> bostoninter<-fgeninter(boston[,1:13],8)[[1]]
[1] "number of interactions: 203490"
> bostonsv<-fstepwise(boston[,14],bostoninter,0.05,20,offset=F,time=T)[[1]]
   user  system elapsed
 14.496   0.364  15.144
> bostonsv
        [,1]         [,2]
 [1,]    441 0.000000e+00
 [2,] 197063 0.000000e+00
 [3,] 197166 7.455263e-10
 [4,] 120886 1.006557e-05
 [5,] 118685 2.726566e-07
 [6,] 121659 1.140579e-03
 [7,] 193641 1.650804e-02
 [8,]     10 1.607415e-03
```

```
 [9,]   7472 4.518208e-11
[10,] 192007 9.425253e-06
> decomp(bostonsv[,1],14,8)
[[1]]
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
 [1,]    0    0    0    0    0    6    6    6
 [2,]    6    6    6    6    6    6    6   13
 [3,]    6    6    6    6    6   11   11   11
 [4,]    1    4    7    9    9    9    9    9
 [5,]    1    4    5    5    6    7    9    9
 [6,]    1    5    5    5    6    6    8   13
 [7,]    5    6    6    6    6    6    8   10
 [8,]    0    0    0    0    0    0    0    9
 [9,]    0    0    0    5    6    6    9   13
[10,]    5    5    6    6    6    9   13   13

> b<-lm(boston[,14]~bostoninter[,bostonsv[,1]])
> print(sum(b$res^2))
[1] 5690.012
> b<-lm(boston[,14]~boston[,1:13])
> print(sum(b$res^2))
[1] 11078.78
> #
> rm(bostoninter)
> #
> #
> # Section 3.5
> #
> #
> xc<-tut1x(1000,1000)[[1]]
```

```
> #
> tmp<-gralss(xc)
user.self
 9243.556
> tmp[[1]]
[1] 1045
> fn<-999-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])==1)
> fp<-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])>=2)
> print(c("(fn,fp)",fn,fp))
[1] "(fn,fp)" "7"       "53"
>
> #
> tmp<-fgraphst(xc,0.05,10,time=T)
   user  system elapsed
 17.356   0.004  17.352
> tmp[[1]]
[1] 994
> fn<-999-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])==1)
> fp<-sum(abs(tmp[[2]][,2]-tmp[[2]][,1])>=2)
> print(c("(fn,fp)",fn,fp))
[1] "(fn,fp)" "5"       "0"
> tmp[[2]][1:10,]
      [,1] [,2]
 [1,]    1    2
 [2,]    2    3
 [3,]    3    4
 [4,]    4    5
 [5,]    5    6
 [6,]    6    7
 [7,]    7    8
```

```
 [8,]    8    9
 [9,]    9   10
[10,]   10   11
> #
> #
> rm(xc)
> #
> frgraph(st=F,lasso=T)
[1] 1829
user.self
 2831.744
[1] 571  13
    user   system  elapsed
2832.240    0.176 2830.770
> #
> #
> frgraph()
[1] 1829
[1]   0 118
   user  system elapsed
 18.516   0.000  18.506
> #
> #
> frgraph(alpha=5)
[1] 1829
[1]  5 12
   user  system elapsed
 21.056   0.004  21.049
> #
> #
```

```
> tmp<-gralss(colon.x)
user.self
      657
> tmp[[1]]
[1] 23322
> #
> tmp<-fgraphst(colon.x,0.05,10,time=T)
   user  system elapsed
  2.660   0.004   2.661
> tmp[[1]]
[1] 1634
> tmp[[2]][1:10,]
      [,1] [,2]
 [1,]    1   23
 [2,]    1   98
 [3,]    2    3
 [4,]    2  126
 [5,]    2  156
 [6,]    4  108
 [7,]    4  131
 [8,]    4  367
 [9,]    5   15
[10,]    5 1201
> #
> tmp<-fgraphstst(colon.x,0.05,10,nedge=1e5,time=T)
   user  system elapsed
 59.624   0.008  59.595
> tmp[[1]]
[1] 24475
> #
```

```
> tmp<-fgraphstst(colon.x,1e-7,10,time=T)
   user  system elapsed
  5.212   0.000   5.207
> tmp[[1]]
[1] 1521
> tmp[[2]][1:10,]
      [,1] [,2]
 [1,]    2    3
 [2,]    2  126
 [3,]    2  156
 [4,]    2  162
 [5,]    3  126
 [6,]    4  108
 [7,]    4  131
 [8,]    4  367
 [9,]    4  416
[10,]    4  457
> #
> #
> #
> #
> tmp<-fgraphst(gse3.x,0.05,10,time=T)
    user   system  elapsed
3007.720    0.472 3006.784
> tmp[[1]]
[1] 38986
> #
> #
> tmp<-fstepstepwise(gse3.x[,1],gse3.x[,2:48802],alpha=0.05/48801,10,time=TRUE)[[1]]
user.self
```

```
  767.292
> length(tmp[,1])
[1] 4009
> #
> #
> tmp<-fgraphstst(gse3.x[,stost],1e-6,10,time=T)
   user  system elapsed
  0.012   0.000   0.010
>
> tmp[[1]]
[1] 160
> tmpi1<-tmp[[2]][,1]
> tmpi2<-tmp[[2]][,2]
> edg<-cbind(stost[tmpi1],stost[tmpi2])
> edg[1:20,]
       [,1]  [,2]
 [1,] 11499  3630
 [2,] 11499 41799
 [3,] 11499 26207
 [4,] 11499 10546
 [5,] 11499 46979
 [6,] 11499 25241
 [7,] 11499 34803
 [8,] 11499   939
 [9,] 11499 10374
[10,] 11499  7896
[11,] 11499 39141
[12,] 11499 26795
[13,] 11499 44664
[14,] 11499  6740
```

```
[15,] 11499 28816
[16,] 11499 24232
[17,] 11499 29522
[18,] 11499 25125
[19,] 31848 44902
[20,] 31848 25241
> time2<-proc.time()
> print(time2[[1]]-time1[[1]])
[1] 31958.58
> q()
```

IRTG 1792 Discussion Paper Series 2018

For a complete list of Discussion Papers published, please visit irtg1792.hu-berlin.de.

001 "Data Driven Value-at-Risk Forecasting using a SVR-GARCH-KDE Hybrid" by Marius Lux, Wolfgang Karl Härdle and Stefan Lessmann, January 2018.

002 "Nonparametric Variable Selection and Its Application to Additive Models" by Zheng-Hui Feng, Lu Lin, Ruo-Qing Zhu asnd Li-Xing Zhu, January 2018.

003 "Systemic Risk in Global Volatility Spillover Networks: Evidence from Option-implied Volatility Indices " by Zihui Yang and Yinggang Zhou, January 2018.

004 "Pricing Cryptocurrency options: the case of CRIX and Bitcoin" by Cathy YH Chen, Wolfgang Karl Härdle, Ai Jun Hou and Weining Wang, January 2018.

005 "Testing for bubbles in cryptocurrencies with time-varying volatility" by Christian M. Hafner, January 2018.

006 "A Note on Cryptocurrencies and Currency Competition" by Anna Almosova, January 2018.

007 "Knowing me, knowing you: inventor mobility and the formation of technology-oriented alliances" by Stefan Wagner and Martin C. Goossen, February 2018.

008 "A Monetary Model of Blockchain" by Anna Almosova, February 2018.

009 "Deregulated day-ahead electricity markets in Southeast Europe: Price forecasting and comparative structural analysis" by Antanina Hryshchuk, Stefan Lessmann, February 2018.

010 "How Sensitive are Tail-related Risk Measures in a Contamination Neighbourhood?" by Wolfgang Karl Härdle, Chengxiu Ling, February 2018.

011 "How to Measure a Performance of a Collaborative Research Centre" by Alona Zharova, Janine Tellinger-Rice, Wolfgang Karl Härdle, February 2018.

012 "Targeting customers for profit: An ensemble learning framework to support marketing decision making" by Stefan Lessmann, Kristof Coussement, Koen W. De Bock, Johannes Haupt, February 2018.

013 "Improving Crime Count Forecasts Using Twitter and Taxi Data" by Lara Vomfell, Wolfgang Karl Härdle, Stefan Lessmann, February 2018.

014 "Price Discovery on Bitcoin Markets" by Paolo Pagnottoni, Dirk G. Baur, Thomas Dimpfl, March 2018.

015 "Bitcoin is not the New Gold - A Comparison of Volatility, Correlation, and Portfolio Performance" by Tony Klein, Hien Pham Thu, Thomas Walther, March 2018.

016 "Time-varying Limit Order Book Networks" by Wolfgang Karl Härdle, Shi Chen, Chong Liang, Melanie Schienle, April 2018.

017 "Regularization Approach for NetworkModeling of German EnergyMarket" by Shi Chen, Wolfgang Karl Härdle, Brenda López Cabrera, May 2018.

018 "Adaptive Nonparametric Clustering" by Kirill Efimov, Larisa Adamyan, Vladimir Spokoiny, May 2018.

019 "Lasso, knockoff and Gaussian covariates: a comparison" by Laurie Davies, May 2018.

IRTG 1792, Spandauer Straße 1, D-10178 Berlin
http://irtg1792.hu-berlin.de

This research was supported by the Deutsche Forschungsgemeinschaft through the IRTG 1792.