E. Heinzen, J. Sinnwell, E. Atkinson, T. Gunderson, and G. Dougherty. *Arsenal: An Arsenal of 'R' Functions for Large-Scale Statistical Summaries*, 2019. URL https://CRAN.R-project.org/package=arsenal. R package version 2.0.0. [p]

P. Hendricks. *Describer: Describe Data in R Using Common Descriptive Statistics*, 2015. URL https://CRAN.R-project.org/package=describer. R package version 0.2.0. [p]

X. Horn. *autoEDA: Automated Univariate and Bivariate Exploratory Data Analysis*, 2018a. R package version 1.0. [p]

X. Horn. Automated exploratory data analysis in r. https://www.linkedin.com/pulse/automated-exploratory-data-analysis-r-xander-horn/, 2018b. Retrieved on 14 March 2019. [p]

K. Hu, D. Orghian, and C. Hidalgo. Dive: A mixed-initiative system supporting integrated data exploration workflows. In *ACM SIGMOD Workshop on Human-In-the-Loop Data Analytics (HILDA)*. ACM, 2018. URL https://doi.org/10.1145/3209900.3209910. [p]

H. Jin, Q. Song, and X. Hu. Auto-Keras: Efficient Neural Architecture Search with Network Morphism. *arXiv e-prints*, 2018. [p]

A. Kassambara and M. Kosinski. *Survminer: Drawing Survival Curves Using 'ggplot2'*, 2018. URL https://CRAN.R-project.org/package=survminer. R package version 0.4.3. [p]

L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown. Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka. *Journal of Machine Learning Research*, 18 (1):826–830, 2017. [p]

R. Krasser. *Explore: Simplifies Exploratory Data Analysis*, 2019. URL https://CRAN.R-project.org/package=explore. R package version 0.4.3. [p]

M. Kuhn and D. Vaughan. *parsnip: A Common API to Modeling and Analysis Functions*, 2019. URL https://CRAN.R-project.org/package=parsnip. R package version 0.0.2. [p]

J. Lemon and P. Grosjean. *prettyR: Pretty Descriptive Stats*, 2018. URL https://CRAN.R-project.org/package=prettyR. R package version 2.2-2. [p]

C. Molnar, B. Bischl, and G. Casalicchio. Iml: An r package for interpretable machine learning. *JOSS*, 3 (26):786, 2018. URL https://doi.org/10.21105/joss.00786. [p]

M. Młynarczyk and P. Biecek. *Cr17: Testing Differences Between Competing Risks Models and Their Visualisations*, 2017. URL https://CRAN.R-project.org/package=cr17. R package version 0.1.0. [p]

A. Nair. *RtutoR: Shiny Apps for Plotting and Exploratory Analysis*, 2018a. URL https://CRAN.R-project.org/package=RtutoR. R package version 1.2. [p]

A. Nair. Automating basic eda. https://www.r-bloggers.com/automating-basic-eda/, 2018b. Retrieved on 25 March 2019. [p]

V. Nijs. *Radiant: Business Analytics Using R and Shiny*, 2019. URL https://CRAN.R-project.org/package=radiant. R package version 0.9.9.1. [p]

R. S. Olson, R. J. Urbanowicz, P. C. Andrews, N. A. Lavender, L. C. Kidd, and J. H. Moore. *Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016, Porto, Portugal, March 30 – April 1, 2016, Proceedings, Part I*, chapter Automating Biomedical Data Science Through Tree-Based Pipeline Optimization, pages 123–137. Springer-Verlag, 2016. URL https://doi.org/10.1007/978-3-319-31204-0_9. [p]

A. H. Petersen and C. T. Ekstrom. *dataMaid: A Suite of Checks for Identification of Potential Errors in a Data Frame as Part of the Data Screening Process*, 2018. URL https://CRAN.R-project.org/package=dataMaid. R package version 1.2.0. [p]

S. Putatunda, K. Rama, D. Ubrangala, and R. Kondapalli. SmartEDA: An R Package for Automated Exploratory Data Analysis. *arXiv e-prints*, art. arXiv:1903.04754, 2019. [p]

M. Quinn, A. McNamara, E. Arino de la Rubia, H. Zhu, and S. Ellis. *Skimr: Compact and Flexible Summaries of Data*, 2019. URL https://CRAN.R-project.org/package=skimr. R package version 1.0.7. [p]

A. Rushworth. *Inspectdf: Inspection, Comparison and Visualisation of Data Frames*, 2019. URL https://CRAN.R-project.org/package=inspectdf. R package version 0.0.3. [p]

C. Ryu. *Dlookr: Tools for Data Diagnosis, Exploration, Transformation*, 2019. URL https://CRAN.R-project.org/package=dlookr. R package version 0.3.8. [p]

B. Schloerke, J. Crowley, D. Cook, F. Briatte, M. Marbach, E. Thoen, A. Elberg, and J. Larmarange. *GGally: Extension to 'ggplot2'*, 2018. URL https://CRAN.R-project.org/package=GGally. R package version 1.4.0. [p]

P. Seibelt. *Xray: X Ray Vision on Your Datasets*, 2017. URL https://CRAN.R-project.org/package=xray. R package version 0.2. [p]

Y. Tang. Autoplotly: An r package for automatic generation of interactive visualizations for statistical results. *Journal of Open Source Software*, 3, 2018. URL https://doi.org/10.21105/joss.00657. [p]

Y. Tang, M. Horikoshi, and W. Li. ggfortify: Unified Interface to Visualize Statistical Results of Popular R Packages. *The R Journal*, 8(2):474–485, 2016. URL https://doi.org/10.32614/rj-2016-060. [p]

N. Tierney. Visdat: Visualising whole data frames. *JOSS*, 2(16):355, 2017. URL https://doi.org/10.21105/joss.00355. [p]

D. Ubrangala, K. Rama, and R. Kondapalli. *SmartEDA: Summarize and Explore the Data*, 2018. URL https://CRAN.R-project.org/package=SmartEDA. R package version 0.3.0. [p]

H. Wickham. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag, 2016. ISBN 978-3-319-24277-4. URL https://doi.org/10.1007/978-0-387-98141-3. [p]

R. Wirth. Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, pages 29–39, 2000. [p]

Y. Xie. *Dynamic Documents with R and Knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition, 2015. URL https://doi.org/10.18637/jss.v056.b02. [p]

K. Yoshida and J. Bohn. *Tableone: Create 'Table 1' to Describe Baseline Characteristics*, 2018. URL https://github.com/kaz-yos/tableone. R package version 0.9.3. [p]

V. Zabalza and F. Engineers. lens, 2018. URL https://doi.org/10.5281/zenodo.2593337. Python library version 0.4.5. [p]

J. J. Zhang and K. B. Storey. RBioplot: An Easy-to-Use R Pipeline for Automated Statistical Analysis and Data Visualization in Molecular Biology and Biochemistry. *PeerJ*, 2016(9), 2016. URL https://doi.org/10.7717/peerj.2436. [p]

Mateusz Staniak
Faculty of Mathematics and Information Science
Warsaw University of Technology
Poland
mtst@mstaniak.pl

Przemysław Biecek
Faculty of Mathematics, Informatics and Mechanics
University of Warsaw
Poland
Samsung R&D Institute Poland (SRPOL)
ORCiD: 0000-0001-8423-1823
przemyslaw.biecek@gmail.com

Task type	Task	a	aE	DE	dM	d	EPD	e	eR	fM	i	R	SE	s	v	x
Dataset	Variable types		x	x	x	x		x		x	x		x	x	x	
	Dimensions		x	x	x	x	x			x	x		x		x	
	Other info			x							x				x	
	Compare datasets	x								x	x				x	
Validity	Missing values		x	x	x	x	x	x		x	x		x	x	x	x
	Redundant col.		x		x	x		x		x			x	x	x	
	Outliers		x		x	x	x			x						
	Atypical values				x					x						x
	Level encoding				x											
Univar.	Descriptive stat.		x		x	x	x	x		x	x	x	x	x		x
	Histograms		x	x	x	x	x	x		x	x	x	x	x		
	Other dist. plots			x			x									
	Bar plots		x	x	x	x	x	x		x	x	x	x	x		
	QQ plots			x		x							x			
Bivar.	Descriptive stat.	x				x		x				x	x	x		
	Correlation matrix			x		x	x				x				x	
	1 vs each corr.		x						x	x			x			
	Time-dependency	x					x									x
	Bar plots by target		x	x		x	x	x		x		x	x			
	Num. plots by target		x			x	x	x		x			x			
	Scatter plots			x			x		x			x	x			
	Contigency tables	x				x							x	x		
	Other stats. (factor)									x	x		x			
Multivar.	PCA			x												
	Stat. models	x					x	x								
	PCP												x			
Transform.	Imputation			x		x		x								
	Scaling					x			x	x						
	Skewness					x										
	Outlier treatment					x	x	x		x						
	Binning			x		x				x						
	Merging levels			x						x						
Reporting	Reports		x	x	x	x		x				x	x			
	Saving outputs	x							x	x				x		x

Table 4: Overview of functionalities of all described packages. Package names were shortened to make the table as compact as possible. **a** denotes **arsenal**, **aE - autoEDA**, **DE - DataExplorer**, **dM - dataMaid**, **d - dlookr**, **EPD - ExPanDaR**, **e - explore**, **eR - exploreR**, **fM - funModeling**, **i - inspectdf**, **R - RtutoR**, **SE - SmartEDA**, **s - summarytools**, **v - visdat**, **x** denotes **xray**. *Num. plots by target* refers to either histogram, density, violin or box plot.

HCmodelSets: An R Package for Specifying Sets of Well-fitting Models in High Dimensions

by Henrique Hoeltgebaum and Heather Battey

Abstract In the context of regression with a large number of explanatory variables, Cox and Battey (2017) emphasize that if there are alternative reasonable explanations of the data that are statistically indistinguishable, one should aim to specify as many of these explanations as is feasible. The standard practice, by contrast, is to report a single effective model for prediction. This paper illustrates the R implementation of the new ideas in the package **HCmodelSets**, using simple reproducible examples and real data. Results of some simulation experiments are also reported.

Introduction

In a recent paper Cox and Battey (2017) outline a procedure for regression analysis when there are more explanatory features than study individuals, a situation that arises particularly in genomics. Their emphasis is on understanding the true data-generating mechanism rather than prediction. The distinction is important. For prediction there may be several models that are essentially equally effective and any choice between them is rather arbitrary. On the other hand, since different well-fitting models typically have different subject-matter implications, it is insufficient, and often misleading, to report an arbitrary one. Even if the immediate goal is prediction, a causal explanation is likely to produce more stable and more generalizable predictions. A key message of Cox and Battey (2017) is that if there are several models that fit the data essentially equally well, one should aim to specify as many as is feasible. This view is in contraposition to that implicit in the use of the lasso (Tibshirani, 1996) and other variable selection methods, which produce a single model effective for prediction.

The methods of Cox and Battey (2017) are summarized in Section Methodology. Software implementing these ideas in R has been written by Hoeltgebaum (2018) and is available in the **HCmodelSets** package. The software supports most widely used models of dependency including the linear model, the linear logistic model for binary data (Cox, 1958), and the proportional hazards model fitted by partial likelihood (Cox, 1972, 1975b). The present article aims to provide a detailed guide to usage based on simple examples.

Methodology

Suppose that data are available on n units, for each of which an outcome y is observed along with a vector x of d potential explanatory variables, where d is much larger than n. For progress an assumption of sparsity is needed, and the most explicit and interpretable such assumption is that relatively few of the potential explanatory features have a real effect, an assumption central to the formulation of the lasso and similar penalized regression procedures.

Cox and Battey (2017) suggest a different approach whose aim is essentially a confidence set of models. There are three stages to a discussion, and conditionally on the first two, the resulting set of models can be made to have the formal statistical properties associated with confidence sets.

In the first stage, an initial reduction is made in which a large number of variables are discarded on the basis that they have no explanatory power, or that any explanatory power that they appear to have is explained away by other variables. The assessment is made by fitting a suitable low-dimensional regression model several times to each variable, each time alongside a different set of k companion variables. A variable is retained for further study if it satisfies a particular criterion in at least half of the analyses in which it appears. The sets of variables to be considered together are specified by a partially balanced incomplete block design (Yates, 1936) in which variable indices are arranged in a hypercube of appropriate dimension. This initial dimension is determined by d and a constraint on k to mitigate the effect of dependence between p-values, or the associated test statistics, in any single analysis. Ideally k will be between 10 and 15; see §7 of Battey and Cox (2018) for a discussion of this choice. Successive reductions are made using arrangements in successively lower-dimensional hypercubes, where the criterion for retaining variables in each stage is guided by the theoretical discussion of Battey and Cox (2018), the need to keep the number of rows, columns, etc., of successive hypercubes ideally ≤ 15, and the requirement for a degree of stability over rerandomization of the variable indices in the hypercube. Thus, judgement is required at various stages.

On the resulting set of variables, of which there will be roughly 10-20 by construction, an exploratory analysis is performed, of the kind that is standard in much statistical work. For instance, inspection of interaction plots or probability plots of t statistics. The objective is to detect possible nonlinearities or outliers.

All variables retained through the reduction phase and any squared or interaction terms suggested at the exploratory phase comprise the so called comprehensive model. All low dimensional subsets of the comprehensive model are then tested for their compatibility with the data using a likelihood ratio test, and all models that pass this test are reported. If, among this set, there are models that contain interaction terms without the corresponding main effects, the main effects are added.

For the resulting sets of models to have the formal statistical properties associated with confidence sets, conditional on the first two phases, it is necessary to either split the sample, see Cox (1975a) for a discussion, or to adjust standard tests of model adequacy in account of the alternative hypothesis being selected in the light of the data.

Illustration of usage: a simple reproducible example

Some simple data generating processes

We illustrate the functions available in **HCmodelSets**, and their appropriate usage, using simple examples. These functions include DGP, which can be used to reproduce the simulation study of Battey and Cox (2018) and to explore further sensitivities.

```
library(HCmodelSets)
dgp = DGP(s=5, a=3, sigStrength=1, rho=0.9, n=100,
                intercept=5, noise=1, var=1, d=1000, DGP.seed = 2018)
```

This generates normally distributed responses as $Y_i = \mu + x_i^T\beta + \varepsilon_i$ $(i = 1, \ldots, n)$ where, in the present example `n = 100`, μ `= intercept = 5`, the ε_i are independently standard normally distributed and the x_i are realizations of a `d = 1000` dimensional normal random vector of mean zero and covariance matrix Σ. The vector of regression coefficients β is sparse in the sense that only `s = 5` entries are non-zero, equal to `sigStrength = 1`, and Σ is such that a correlation `rho = 0.9` is induced between the corresponding entries of x_i, the so called signal variables, and among `a = 3` of the remaining variables. All potential explanatory variables have variance `var = 1`. The results of the subsequent analysis can be reproduced by setting `DGP.seed = 2018` as above, but this argument is not needed.

With the appropriate modification to its arguments, the DGP function also generates survival times from a proportional hazards model with Weibull baseline hazard. The hazard function for the ith individual is

$$h_i(t;\beta) = \kappa\tau(\tau t)^{\kappa-1}\exp\{x_i^T\beta\},$$

where $h(t) = \kappa\tau(\tau t)^{\kappa-1}$ is the Weibull hazard function. From this, the density and distribution functions of survival times conditional on x_i are obtained as

$$\begin{aligned} f_i(t;\beta) &= \kappa\tau(\tau t)^{\kappa-1}\exp\{x_i^T\beta\}\exp\{-e^{x_i^T\beta}(\tau t)^{\kappa}\}, \\ F_i(t;\beta) &= \exp\{-e^{x_i^T\beta}(\tau t)^{\kappa}\}. \end{aligned}$$

Thus, given covariates x_i, uncensored survival times from the above proportional hazards model are generated as $T_i = \{-\log U/(\tau^{\kappa}e^{x_i^T\beta})\}^{1/\kappa}$, where U is a uniform random variable on $(0,1)$.

In the section entitled Illustration of performance in some idealized settings a minor modification to the previous code is given to generate (potentially censored) survival time data from this model. Simulation results for the procedure fitted to both types of data are also reported in the same section.

Reduction phase

Based on the previous output, typical usage of the function Reduction.Phase is:

```
out = Reduction.Phase(X = dgp$X,Y = dgp$Y,
                family = gaussian, seed.HC = 1012)
```

In particular, this arranges the indices of the columns of dgp$X in a hypercube of appropriate dimension, and fits a linear regression model to each set of variables indexed by the rows, columns, etc., of the hypercube. Other choices of the argument family are illustrated in section entitled Real example. The arrangement of the variable indices in the hypercube is at random. However, `seed.HC = 1012` allows the results of the analyses reported here to be reproduced. If the argument dmHC is left unspecified (the recommendation), as in this example, the dimension of the initial hypercube is set to be the smallest dimension such that the number of rows, columns etc., is no greater than 15. Thus, the present example initially has the 1000 variable indices arranged in a $10 \times 10 \times 10$ cube.

Because the comprehensive model obtained from the full data achieves better fit than an arbitrary model embedding the one to be tested, a test of adequacy of the smaller model rejects too often in hypothetical repeated application. It is therefore usually sensible to split the sample in two and use, say 70%, for the reduction and exploratory phases, and the remaining 30% for construction of the conditional confidence sets of models. The appropriate modification to the previous code, so that only the first 70 observations are used for the reduction phase, is:

```
outSplit1 = Reduction.Phase(X = dgp$X[1:70,],
                Y = dgp$Y[1:70], family = gaussian, seed.HC = 1012)
```

If the initial sample size is rather small and the model to be fitted is non-Gaussian, the sample size available for the final phase of the procedure is likely to be too small for the distribution of the maximum likelihood estimator to be well-approximated by its asymptotic distribution. Correspondingly, the coverage probability of the confidence sets of models conditional on the reduction phase is likely to differ from the nominal value. This could be mitigated through a Bartlett correction to the likelihood ratio statistic, but this has not been implemented in the current version of the package. See Bartlett (1937); Barndorff-Nielsen and Cox (1984) and Barndorff-Nielsen and Cox (1994) (p133, p152–53) for a discussion of the theory of Bartlett correction.

A strong reassurance over the security of one's conclusions is given if the set of retained variables does not alter much upon rerandomization of the arrangement of the variable indices in the (hyper)cube, and this is a suitably cautious check in practice. Indeed, if the answers so obtained differ appreciably, the suggestion is that too severe a reduction has been performed. Thus we consider also the outcome outSplit2, obtained when no argument seed.HC is provided, so that variable indices are arranged in their original order. Some variables will appear in all or almost all analyses.

```
outSplit2 = Reduction.Phase(X = dgp$X[1:70,],
                Y = dgp$Y[1:70], family = gaussian)
```

The outcomes outSplit1 and outSplit2 of the previous two analyses are two lists of variable indices from each successive reduction. Only the latter reductions are of ultimate interest, but the intermediate reductions should be inspected to ensure that the number of variables retained is not so large as to be detrimental to the subsequent stage of the reduction. In the present example, the final lists of variables are arrived at by an implementation of the default decision rules, to some extent guided by the analysis of Battey and Cox (2018). These are to retain variables if they are among the two most significant in at least half the analyses in which they appear in the first stage reduction, and if they are significant at the 1% level in at least half the analysis in which they appear in subsequent reductions. The 1% threshold is arbitrary and judgement should be exercised if the output of such an analysis is unreasonable, for instance if too many variables are retained in any stage of the reduction. This is particularly important when the initial number of variables is very large, so that variables are initially arranged in a four or five dimensional hypercube. The Real example illustrates appropriate use of judgement through the optional argument vector.signif of the Reduction.Phase function. The objective of the reduction phase is to reduce the number of candidate signal variables to ideally not more than 20, to be subjected to more detailed joint analysis.

The sorted lists of retained variables using the default decision rules and the two initial arrangements of variables indices in the cube are:

```
v1=sort(outSplit1$List.Selection$`Hypercube with dim 2`$numSelected1)
v2=sort(outSplit2$List.Selection$`Hypercube with dim 2`$numSelected1)
v1
#> [1] 46 51 66 156 229 263 272 319 423 496 531 559 735 804 827 897 929 984 1000
v2
#> [1] 46 156 272 291 319 397 531 559 642 827 897 929 984
```

Of these variables, ten are in common, an appreciable overlap. The indices of the five true signal variables are contained in v1 and v2 (and their intersection). These are

```
dgp$TRUE.idx
#> [1] 46 531 559 897 929
```

Usage of the other functions in the package is illustrated using the output of the second analysis, i.e., the variables in v2.

An alternative to the reduction phase is to use a deliberately undertuned lasso fit. The lasso is typically fitted by the coordinate descent algorithm in general regression settings, or by the least angle regression algorithm in the linear model. Thus, the practical implementation of the lasso is essentially forward selection. By contrast, the reduction phase of Cox and Battey (2017) is a version of backward elimination. Both forward selection and backward elimination are likely to be effective in many cases, although a theoretical elucidation of the conditions on the design matrix to ensure this has not been attempted. If the objective is to obtain a superset of the comprehensive model, as here, backward elimination has advantages in simpler settings. See Illustration of performance in some idealized settings for an empirical comparison in idealized examples.

The lasso, fitted by coordinate descent as implemented in the R package **glmnet**, and undertuned to produce at least the same number of variables as in v2, is obtained by

```
library(glmnet)
lasso.fit = glmnet(x = dgp$X[1:70,],y = dgp$Y[1:70])
n.coefs = apply(coef(lasso.fit), 2, function(x) length(which(x!=0)))
idx.coefs = which(n.coefs == length(v2))
if(length(idx.coefs)==0){
        idx.coefs = min(which(n.coefs >= length(v2)))}
lasso.var = which(coef(lasso.fit)[,idx.coefs[1]]!=0)
```

In the present example, the associated variables excluding the intercept, are

```
lasso.var[-1]-1
#> [1] 40 46  161 341 384 511 531  559  827 897  929  984
```

Seven of these are in common with v1 and v2, including the five signal variables.

Exploratory phase

The analysis discussed by Cox and Battey (2017) is intended to be largely exploratory, and a key aspect of the procedure is that it allows informal checks, standard in much statistical work. The function Exploratory.Phase automates some, but by no means all, of what would typically take place in an exploratory data analysis, and is provided as a rough guide. Usage of the silent argument is illustrated in the Real example section, in which silent = FALSE forces a certain degree of judgement to be exercised.

The following code detects potentially important squared or interaction terms among the variables whose indices are stored in v2.

```
out.exp.phase = Exploratory.Phase(X = dgp$X[1:70,],
                Y = dgp$Y[1:70], list.reduction = v2,
                family = gaussian, signif = 0.01)
```

Neither squared terms nor interaction terms are suggested as potentially important.

Model selection phase

The final stage of the procedure is to test all low-dimensional subsets of the comprehensive model for compatibility with the data. The comprehensive model is that containing all variables from the reduction phase and any squared or interaction terms suggested at the exploratory phase, of which there are none in the present example. Usage is:

```
out.MS = ModelSelection.Phase(X = dgp$X[71:100,],
                Y = dgp$Y[71:100], list.reduction = v2, signif = 0.01)
```

The appropriate modification to the arguments of this function when squared or interaction terms are to be considered is illustrated in the Real example section.

The above finds all models of dimension 5 or smaller whose likelihood ratio test against the comprehensive is not rejected at the signif = 0.01 significance level. The optional argument modelSize specifies the maximum size of the models to be searched over. The true model appears in the set of all well-fitting models identified, i.e., in the list of models displayed by out.MS$goodModels$`Model Size 5`. All models that are found to be compatible with the data should be reported. Specifically, the output of the function ModelSelection.Phase should be used to produce (sometimes large) tables like those appearing in the supplementary file of Cox and Battey (2017), available at:
https://www.pnas.org/content/pnas/suppl/2017/07/20/1703764114.DCSupplemental

Provided that the sample is split as in the example above, such tables constitute a conditional confidence set of models. Conditional on the reduction phase, these have, in principle, exact nominal coverage in the linear regression model and asymptotically nominal coverage in more general regression models fitted by maximum likelihood.

Illustration of performance in some idealized settings

The present section explores empirical sensitivities of the procedure to modifications to the data generating mechanism. Several aspects are of interest: sensitivity of the reduction phase as described by Cox and Battey (2017) (a version of backward elimination) and of the undertuned lasso (a version of forward selection) in terms of retaining the true model in its entirety; efficacy of the model confidence sets in terms of their coverage probabilities and size. Full sample and split sample properties of both approaches are considered.

It is an open problem to elucidate the conditions on the design matrix and signal strength in order for the procedure based on traversal of successively lower dimensional hypercubes to retain a reasonably sized superset of the true set of signal variables with quantifiable high probability. Some related discussion for the undertuned lasso is given by Bühlmann and Van De Geer (2011) chapter 7 and Belloni and Chernozhukov (2013).

In the tables below, $\mathcal{S}$ is the true set of signal variables, $\widehat{\mathcal{S}}$ is the set of variables surviving the reduction phase, $\mathcal{M}$ is the set of low-dimensional models whose likelihood ratio test against the comprehensive model is not rejected at the 1% level. In all the simulation experiments considered, the first stage of the reduction phase arranges the 1000 variables in a $10 \times 10 \times 10$ cube and retains variables if they are among the two most significant in at least two of the three analyses in which they appear. The second stage reduction is tuned so that approximately 10-20 variables are retained through the reduction phase, however the associated threshold for the significance tests is fixed across Monte Carlo replications so that the number of retained variables is random. Results for the linear model with a sample of size $n = 100$ are reported in Table 1, where 'CB' is the procedure of Cox and Battey (2017) implemented using **HCmodelSets** package. The threshold of the second-stage significance test is 0.1%.

				$\mathrm{pr}(\mathcal{S} \subseteq \widehat{\mathcal{S}})$				$\mathrm{pr}(\mathcal{S} \in \mathcal{M})$		$\mathbb{E}\lvert\mathcal{M}\backslash\mathcal{S}\rvert$	
v_{S0}	v_{C0}	ρ	$\frac{\text{signal}}{\text{noise}}$	undertuned lasso (full)	undertuned lasso (split)	CB (full)	CB (split)	CB (full)	CB (split)	CB (full)	CB (split)
1	1	0.9	1	1.00 (0.04)	0.99 (0.08)	1.00 (0.00)	1.00 (0.00)	0.57 (0.49)	0.98 (0.14)	6.16 (7.25)	16.3 (28.6)
1	1	0.9	0.6	0.94 (0.24)	0.84 (0.37)	0.98 (0.15)	0.83 (0.37)	0.41 (0.49)	0.83 (0.38)	4.93 (4.82)	16.1 (31.5)
1	1	0.5	1	0.92 (0.27)	0.87 (0.34)	1.00 (0.00)	1.00 (0.00)	0.56 (0.50)	0.98 (0.13)	4.63 (5.89)	8.73 (18.7)
1	1	0.5	0.6	0.85 (0.36)	0.75 (0.43)	0.99 (0.11)	0.85 (0.35)	0.39 (0.49)	0.84 (0.36)	2.29 (2.73)	8.68 (19.2)
1	3	0.9	1	1.00 (0.04)	0.99 (0.08)	1.00 (0.04)	0.99 (0.08)	0.62 (0.49)	0.96 (0.19)	24.6 (24.8)	77.2 (126)
1	3	0.9	0.6	0.92 (0.27)	0.79 (0.41)	0.97 (0.16)	0.81 (0.39)	0.48 (0.50)	0.79 (0.41)	22.7 (18.4)	44.6 (78.1)
1	3	0.5	1	0.97 (0.16)	0.92 (0.27)	1.00 (0.00)	1.00 (0.00)	0.59 (0.49)	0.98 (0.13)	10.9 (14.0)	14.3 (36.1)
1	3	0.5	0.6	0.89 (0.31)	0.82 (0.39)	0.97 (0.17)	0.87 (0.34)	0.36 (0.48)	0.85 (0.35)	3.66 (5.01)	10.3 (25.0)
5	1	0.9	1	0.98 (0.15)	0.95 (0.21)	1.00 (0.00)	1.00 (0.00)	0.94 (0.23)	0.98 (0.13)	7.15 (7.19)	80.4 (85.7)
5	1	0.9	0.6	0.79 (0.40)	0.57 (0.50)	1.00 (0.04)	0.98 (0.15)	0.89 (0.31)	0.96 (0.19)	40.9 (35.7)	146 (149)
5	1	0.5	1	1.00 (0.04)	0.98 (0.15)	1.00 (0.00)	1.00 (0.00)	0.96 (0.20)	0.99 (0.11)	0.01 (0.10)	8.64 (13.0)
5	1	0.5	0.6	0.99 (0.10)	0.96 (0.21)	1.00 (0.00)	0.99 (0.10)	0.88 (0.32)	0.98 (0.15)	1.18 (2.04)	51.6 (64.2)
5	3	0.9	1	0.99 (0.11)	0.95 (0.22)	1.00 (0.00)	1.00 (0.00)	0.94 (0.24)	0.98 (0.15)	16.7 (18.4)	212 (202)
5	3	0.9	0.6	0.77 (0.42)	0.51 (0.50)	1.00 (0.06)	0.96 (0.20)	0.86 (0.35)	0.94 (0.24)	101 (88.2)	418 (351)
5	3	0.5	1	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	0.96 (0.20)	0.98 (0.14)	0.01 (0.10)	20.0 (35.0)
5	3	0.5	0.6	1.00 (0.06)	0.98 (0.13)	1.00 (0.00)	0.99 (0.12)	0.89 (0.32)	0.96 (0.20)	2.85 (4.03)	123 (137)

Table 1: Monte Carlo estimates and their estimated standard errors (in parentheses) from 500 Monte Carlo draws from the linear model with parameter combinations as displayed. In the split sample case, 70 observations are used for reduction and 30 for construction of the confidence sets of models.

The same experiment is performed on survival time data, generated according to a proportional hazards model with Weibull baseline hazard as described in the section entitled Some simple data generating processes. The survival times are censored, with the censoring times generated from an exponential distribution of rate 0.1. In particular, our previous code fragment is modified so that in each Monte Carlo replication, data are generated as:

```
dgp = DGP(s=Vs0, a=Vc0, sigStrength=sigNoise, rho=corr, n=150, intercept=0,
                DGP.seed = (n.rand + cont.error),
                var=1, d=1000, type.response = "S", scale=1, shape=1, rate=0.1)
```

adopting the values `c(1,5)` for `Vs0`, `c(1,3)` for `Vc0`, `c(0.9,0.5)` for `corr` and `c(1,0.6)` for `sigNoise` as previously done for the linear theory model in Table 1.

In the notation of Section Some simple data generating processes, the parameters of the Weibull distribution are set as τ = `scale = 1`, and κ = `shape = 1`. Knowledge of the baseline hazard is ignored and the data are fit by partial likelihood as implemented in the `coxph` function of the **survival** package. Summary statistics over 500 Monte Carlo replications are reported in Table 2. The threshold of the second-stage significance test is 0.25%.

				$\text{pr}(\mathcal{S} \subseteq \hat{\mathcal{S}})$				$\text{pr}(\mathcal{S} \in \mathcal{M})$		$\mathbb{E}\lvert\mathcal{M}\backslash\mathcal{S}\rvert$	
v_{S0}	v_{C0}	ρ	signal / noise	undertuned lasso (full)	undertuned lasso (split)	CB (full)	CB (split)	CB (full)	CB (split)	CB (full)	CB (split)
1	1	0.9	1	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	0.03 (0.17)	0.95 (0.23)	54.1 (92.2)	1273 (1490)
1	1	0.9	0.6	0.99 (0.12)	0.94 (0.24)	1.00 (0.04)	0.97 (0.17)	0.00 (0.04)	0.89 (0.31)	15.6 (57.3)	1863 (2264)
1	1	0.5	1	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	0.04 (0.21)	0.95 (0.21)	57.4 (97.4)	962 (1085)
1	1	0.5	0.6	1.00 (0.00)	0.98 (0.13)	0.99 (0.08)	0.96 (0.20)	0.00 (0.00)	0.90 (0.31)	13.0 (31.6)	1734 (2374)
1	3	0.9	1	1.00 (0.00)	0.99 (0.09)	1.00 (0.00)	1.00 (0.04)	0.07 (0.25)	0.95 (0.22)	102 (209)	2468 (2738)
1	3	0.9	0.6	0.97 (0.18)	0.90 (0.30)	0.98 (0.13)	0.95 (0.21)	0.01 (0.09)	0.91 (0.29)	45.0 (98.5)	3182 (3700)
1	3	0.5	1	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	0.07 (0.25)	0.95 (0.22)	105 (158)	1094 (1090)
1	3	0.5	0.6	1.00 (0.00)	1.00 (0.06)	1.00 (0.00)	0.97 (0.17)	0.00 (0.04)	0.91 (0.28)	18.6 (51.1)	1955 (2859)
5	1	0.9	1	0.98 (0.15)	0.90 (0.30)	1.00 (0.00)	1.00 (0.04)	0.78 (0.41)	0.91 (0.29)	30.9 (46.5)	916 (1165)
5	1	0.9	0.6	0.79 (0.41)	0.52 (0.50)	1.00 (0.00)	0.99 (0.08)	0.59 (0.49)	0.94 (0.24)	136 (180)	2216 (2390)
5	1	0.5	1	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	0.80 (0.40)	0.91 (0.28)	0.00 (0.09)	59.0 (118)
5	1	0.5	0.6	1.00 (0.00)	0.99 (0.12)	1.00 (0.00)	1.00 (0.04)	0.54 (0.50)	0.90 (0.31)	1.46 (4.22)	382 (572)
5	3	0.9	1	0.98 (0.13)	0.86 (0.35)	1.00 (0.00)	1.00 (0.04)	0.80 (0.40)	0.86 (0.35)	46.4 (66.2)	1383 (1682)
5	3	0.9	0.6	0.71 (0.45)	0.48 (0.50)	1.00 (0.00)	0.99 (0.11)	0.63 (0.48)	0.90 (0.30)	242 (310)	2846 (2603)
5	3	0.5	1	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	1.00 (0.00)	0.83 (0.38)	0.87 (0.34)	0.09 (1.03)	73.4 (175)
5	3	0.5	0.6	1.00 (0.00)	0.99 (0.11)	1.00 (0.00)	1.00 (0.04)	0.59 (0.49)	0.90 (0.30)	2.35 (5.25)	575 (925)

Table 2: Monte Carlo estimates and their estimated standard errors (in parentheses) from 500 Monte Carlo draws from the Weibull proportional hazards model with parameter combinations as displayed.

The results are qualitatively similar to those for the linear model. The main difference is that the coverage probability of the confidence sets of models, conditional on all variables being retained through the first stage reduction, is lower than the 0.99 nominal level. The reason is that the distribution theory underpinning the associated likelihood ratio tests is, in principle, exact for the linear model and is at best asymptotically valid for most other types of regression model. This could be mitigated through a Bartlett correction to the likelihood ratio statistic, but this has not been implemented in the current version of the package. The results of Table 2 are for $n = 150$ with 100 observations used for reduction and 50 for construction of confidence sets of models in the split sample case. As mentioned previously, adjustments to the likelihood ratio statistic to improve the χ^2 approximation to its distribution are possible, but these have not been implemented in **HCmodelSets**.

Real example

We now illustrate the use of **HCmodelSets** to construct conditional confidence sets of models for the survival times of lymphoma patients. The data[1] are from the study of Alizadeh et al. (2000) and also used by Simon et al. (2011). There are measurements on $d = 7399$ genetic variants for $n = 240$ patients. The indices of these variables are arranged in a 4 dimensional hypercube, which is the default starting dimension. As before, the data are divided into those to be used in the reduction and exploratory phases and those to be used in the model selection phases.

```
data(LymphomaData)
x = t(patient.data$x)
y = patient.data$time
status = patient.data$status
# Data Splitting
X.in = x[1:168,]
Y.in = y[1:168]
status.in = status[1:168]
Y = cbind(Y.in,status.in)
X.out = x[169:240,]
Y.out = y[169:240]
status.out = status[169:240]
```

The first stage decision rule is to retain all variables that are among the two most significant in at least two of the three analyses in which they appear. The decision rules for the remaining reduction stages are specified by the argument `vector.signif` in the `Reduction.Phase` function:

[1]Avaliable at https://www.jstatsoft.org/article/view/v039i05

```
library(HCmodelSets)
out.1 = Reduction.Phase(X = X.in,Y = Y,Cox.Hazard = TRUE,
                vector.signif = c(2,0.0025,0.001), seed.HC = 2)
```

The choice `vector.signif = c(2,0.0025,0.001)` means that the second stage decision rule retains variables if they are significant at the 0.25% level in at least two of the three analyses in which they appear and the third stage decision rule retains variables if they are significant at the 0.1% level in at least one of the two analyses in which they appear. This choice was determined by checking that the numbers of variables retained through each stage of the reduction is sensible, that the number of variables ultimately retained is within the target range, and that the outcome is not too sensitive to changes to the original arrangement of the variable indices in the hypercube. The set of variables ultimately retained is

```
v1 = out.1$List.Selection$`Hypercube with dim 2`$numSelected1
sort(v1)
#> [1] 1188 1660 1825 2437 2879 2902 3172 3177 3800 3814 3822 3824 5027 6134 6706 6896 7357
```

Rerandomizing the variable indices in the hypercube produces the set of variables

```
out.2 = Reduction.Phase(X = X.in,Y = Y,Cox.Hazard = TRUE,
                vector.signif = c(2,0.0025,0.001), seed.HC = 11)
v2 = out.2$List.Selection$`Hypercube with dim 2`$numSelected1
sort(v2)
#> [1] 1188 1675 1714 1825 1984 2437 2900 3172 3800 3811 3818 3819 3822 3833  4126
        5027  6134  6706  7069  7357
```

Ten of the variables in the original list of 17 are also in the second list. The lasso, undertuned to select at least the same number of variables as in `v1` produces an overlap of 8 variables with `v1`, namely,

```
library(glmnet)
lasso.fit = glmnet(X.in, Surv(Y.in,status.in),
                family = "cox", alpha = 1)
n.coefs = apply(coef(lasso.fit), 2, function(x) length(which(x!=0)))
idx.coefs = which(n.coefs == length(v1))
if(length(idx.coefs)==0){
        idx.coefs = min(which(n.coefs >= v1))}
lasso.var = which(coef(lasso.fit)[,idx.coefs[1]]!=0)
lasso.var
#> [1] 394 1072 1188 1456 1662 1681 1825 2902 3172 3180 3801
        3822 4882 5027 6134 6896 7357
```

The variable 3801, found by the lasso, has empirical correlation greater than 0.9 with variable 3800 in `v1` and `v2`.

The exploratory phase now uses significance tests as an informal guide to suggesting potential squared or interaction terms. For each of the variables in `v1`, a regression is fitted by partial likelihood with its squared term added. Extreme t statistics on squared terms suggest a potentially important effect. The linear interactions of pairs of variables are checked in a similar way, with `silent = FALSE` in `Exploratory.Phase` producing plots of the response variable as a function of pairs of variables for any interaction suggested as potentially important. Example usage is

```
out.exp.phase =  Exploratory.Phase(X=X.in,Y=Y,
                list.reduction = v1, silent = FALSE,
                Cox.Hazard = TRUE, signif=0.01)
```

which produces a sequence of plots and questions of the form

```
Discard interaction term? [Y/N].
```

For illustrative purposes, we answer `N` (no) to the questions for which the plots are displayed in Figure 1, although the suggestion from an interaction plot ought to be much stronger to justify an interaction's inclusion. See Cox and Battey (2017) for an example.

Thus we have 20 variables in all, the seventeen variables contained in `v1`, one squared term contained in `out.Exploratory.Phase$mat.select.SQ` and two interaction terms given by the rows of `out.Exploratory.Phase$mat.select.INTER`. The analysis proceeds as follows:

```
sq.terms = out.exp.phase$mat.select.SQ
in.terms = out.exp.phase$mat.select.INTER
```