STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **10**, June 2022, pp 949–967.
Published online in International Academic Press (www.IAPress.org)

New Algorithms and Software for Significance Controlled Variable Selection

Adriano Z. Zambom [1*], Jongwook Kim [2]

[1] *Department of Mathematics, California State University Northridge, USA*
[2] *Department of Statistics, Indiana University Bloomington, USA*

Abstract Stepwise regression algorithms have been widely used for a variety of applications and continue to be a fundamental tool in variable selection. Most functions available in statistical software packages deliver models that may contain insignificant predictors because of the criterion of the optimization at each step. Here we introduce an R package that provides the user with several measures of the prospective model at each step of the algorithm. These prospective models are checked with multiple testing p-value corrections such as Bonferroni and False Discovery Rate and hence the algorithm's final model includes only predictors that have their significance controlled by the choice of correction type and alpha level. Moreover, the steps forward or backward can have an entry or drop criterion that is a combination of the p-values of prospective models. We illustrate the functionality of the package with examples and simulations.

Keywords multiple testing, forward selection, backward elimination, stepwise selection, p-value correction

AMS 2010 subject classifications 62J05, 62J12

DOI: 10.19139/soic-2310-5070-1520

1. Introduction

Enhancing predictive power and accuracy through parsimonious models is one of the primary concerns in modern statistical model building. Stepwise regression ([4], [3]) is one of the most popular solutions in diverse applications. It has been widely used for decades and continues to be an important tool in recent research, see for example its application in [14], [8], [10], and [9]. The idea of the methods implemented in the R ([13]) package *SignifReg*, which is available in the Comprehensive R Archive Network (CRAN) at *https://CRAN.R-project.org/package=SignifReg*, is to perform forward selection, backward elimination, and stepwise regression using a novel criterium that is based on a combination of the p-values of the predictors already selected and the p-value of the predictor to be added/dropped at each step. Additionally, the package allows the use of AIC, BIC, adjusted R-square, or PRESS residuals as the criterium for a predictor to enter/leave the model while controlling, at every step, the significance of the retained predictors based on their p-values. The package options for controlling the significance of predictors include those in the *p.adjust*() function in the *stats* package ("holm" [6], "hochberg" [5], "hommel" [7], "BH" or its alias "fdr" [1], and "BY" [2]), or no correction "none" where all p-values must be below the alpha level, which is an argument of the function. Note that if the choice of criterium is AIC for example, and the alpha level is set to 1 (no p-value corrections), the algorithm is equivalent to the *step* function in the *stats* package in R. Overall, with the combination of stepwise directions, criteria, and corrections, the *SignifReg* package offers a wide range of novel options for variable selection.

*Correspondence to: Adriano Zanin Zambom (Email: adriano.zambom@csun.edu). Department of Mathematics, California State University Northridge, California, USA.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

Given the high demand for easy to use software for practical applications, various software packages have been developed to implement stepwise algorithms for variable selection. For instance, the SAS (*https://www.sas.com*) software offers the *PROC GLMSELECT*, which can be specified with the "*SELECTION* = " options for forward, backward, and stepwise directions. This procedure provides several criteria choices such as SBC (Schwarz's Bayesian Criterion), F-statistics, AIC (Akaike Information Criterion), and the average square error on validation data. Likewise, Matlab possesses various functions for stepwise regression. For instance, *stepwiselm* offers stepwise regression with sum of squares of errors (SSE), AIC, BIC, r-squared, and adjusted r-squared. Minitab performs stepwise regression by adding or removing variables until all variables not in the model have p-values greater than the specified significance level, and all variables in the model have p-values less than or equal to the specified significance level. In R, the *step* and the *stepAIC* are the most commonly used procedures. These functions use the AIC as a criterion for the model selection at each step by adding or removing variables as long as the model decreases the AIC value. Additionally, there are other packages in R that perform stepwise variable selection, as for example the package *olsrr* and its several functions, the *regsubsets* function in the *leaps* package, the *train* function in the *caret* package, among others. However, to the best of our knowledge, no software performs stepwise variable selection while controlling for false discovery rate or using Bonferroni p-value corrections at each step, nor do they make use of the combination of the p-values of already selected and prospective predictors as criteria to add/drop predictors.

The broad availability of software that contain stepwise variable selection algorithms makes the use of statistical techniques accessible to all researchers, however it can lead to serious misjudgment or misinterpretation of results. It is important to understand the underlying variable selection algorithm, as it is so widely used by practitioners of all backgrounds. For instance in epidemiological analysis, a total 59 (20%) among 300 articles used stepwise regression for selection of covariates in the four major epidemiological journals only in the year of 2008 (American Journal of Epidemiology, Epidemiology, European Journal of Epidemiology and the International Journal of Epidemiology) ([15]). Overall, it is important that the selection method achieves the desired outcome with theoretical foundation, however it is crucial that the final model output from the software they choose yields low prediction error. With the huge amount of available data in all fields of research, the risk of overfitting becomes higher as technology allows for the collection of more predictors. Thus, it is desirable that the selected predictors have high evidence against the null hypothesis that their corresponding parameters are equal to 0. This is important because the most used classical stepwise variable selectors do not require that all selected predictors are significant, that is, the final selected model may contain predictors whose corresponding p-values are large. In forward, backward and stepwise algorithms this happens because in many cases the AIC or the BIC are used as the criterium for inclusion/exclusion of predictors. More specifically for the forward selection, each addition of a new variable may render one or more of the already included variables non-significant when the criteria for addition is lowering the AIC or even when the criteria is a significant p-value of the prospective predictor.

In this paper we introduce new algorithms for backward elimination and stepwise selection whose criteria to include or exclude predictors is based on the maxium p-value corresponding to hypothesis tests in prospective model. Moreover, we compare the general algorithms developed to benchmark procedures in the *R* system and relevant implementations in other software systems. The general goal of the algorithms in the *SignifReg* package is to select the predictors that compose the true underlying data mechanism rather than the set of predictors with smallest prediction error. When building a regression model, the set of predictors, among all available covariates, that yield the smallest prediction error may not include the true underlying model. This can be due to spurious correlations or simply the randomness of the data. However, if the algorithm manages to select the significant predictors only, in the sense of the true underlying model, as the sample size increases one expects the prediction error to decrease. The methods in this paper are specially recommended for situations where the sample size is moderate to large, which is commonly found in modern datasets, because of the large number of hypothesis tests performed and the multiple testing corrections.

The remainder of the paper is structured as follows. In Section 2 we summarize the forward selection, backward elimination and a stepwise algorithm using selection/elimination strategy based on the p-values of prospective

models. Section 3 presents the package *SignifReg* with its main function *SignifReg*() and its numerous criteria and correction choices with examples. In addition, the functions *add1SignifReg* and *drop1SignifReg* (similar to the *add1* and *drop1* from the *stats* package in R) are introduced, which perform one step of the main algorithms while providing the user with a table with the information of prospective models. In the simulations in Section 4, the performance of the proposed method is compared to that of other functions in widely used statistical software for different signal to noise ratios determined by an increasing variance of the model error. We observe that the proposed methods perform better in selecting the correct predictors for small and moderate values of the noise while underfitting when the noise is high.

2. Consistent Significance Controlled Variable Selection

Assume there are d available covariates $X_1, \ldots, X_d$, however only d_0 of them, denoted as $X_{\ell_1}, \ldots, X_{\ell_{d_0}}$, compose the true data generating mechanism for the regression model. Let $I_0 = \{\ell_1, \ldots, \ell_{d_0}\} \subseteq \{1, \ldots, d\}$ denote the subset of indices corresponding to the d_0 predictors in the true model. To be specific, assume that the data is generated by the linear regression model

$$\mathbf{Y} = \mathbf{X}_0 \boldsymbol{\beta}_0 + \boldsymbol{\epsilon},$$

where $\mathbf{Y} = (Y_1, \ldots, Y_n)'$, $\mathbf{X}_0$ is the $n \times (d_0 + 1)$ design matrix of observed values of the covariates $X_{\ell_1}, \ldots, X_{\ell_{d_0}}$, $\boldsymbol{\beta}_0 = (\gamma, \beta_{\ell_1}, \ldots, \beta_{\ell_{d_0}})'$ is the vector of unknown parameters in the true model and $\boldsymbol{\epsilon} = (\boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_n)$ is a vector of i.i.d. errors with $E(\epsilon_i) = 0$, $E(\epsilon_i^2) = \sigma^2$, and $E|\epsilon_i|^{4+\delta} < \infty$ for some $\delta > 0$. Using this framework, the objective is to select with high accuracy the true predictors from the d available, that is, estimate a set of indices $\hat{I}$ which identifies the subset I_0 with high accuracy.

In this section we will describe in detail the forward selection algorithm ([16]) which uses a novel criterion based on the combination of the p-values while applying a Bonferroni correction at every step. With the sample size increasing to infinity, it can be theoretically shown that forward selecting predictors while controlling for Bonferroni (or other p-value correction methods) is consistent in the sense that $\hat{I}$ is equal to I_0 with probability increasing to 1 (see [16] for details). The use of AIC, BIC and other criteria as well as other correction methods (FDR or fixed value) follows with straightforward changes. For ease of notation, let $\hat{I}_k$ be the set of indices selected after step k of the algorithm. If k = 1, then $\hat{I}_k$ is the empty set. Let $\{\hat{I}_k, r\}$ be the set of predictors in $\hat{I}_k$ with a new added predictor r. Let $\pi^j_{\{\hat{I}_k, r\}}$ be the p-value of the hypothesis test of the j-th coefficient, that is $H_0 : \beta^j_{\{\hat{I}_k, r\}} = 0, j \in \{\hat{I}_k, r\}$, computed from the linear regression model with predictors corresponding to the set of indices $\{\hat{I}_k, r\}$. Let α be a specified Type I error rate that we are willing to commit, which will be used in the Bonferroni correction (or FDR, or fixed for all p-values). The algorithm is as follows.

Forward Variable Selection Algorithm

==========

Step 1. Starting from the NULL model, let $\hat{I}_1$ be the index ℓ of the covariate with lowest marginal p-value, i.e., $\ell = \arg\min_\ell \{\pi^1, \ldots, \pi^d\}$, where $\pi^j, j = 1, \ldots, d$, is the p-value obtained from the simple linear regression of Y on each X_j; as long as $\pi^\ell \leq \alpha$ and set $k = 2$. If no $\pi^j \leq \alpha$, then stop and retain no predictor.

Step 2. Given the previously selected lags $\hat{I}_k$, include in $\hat{I}_k$ the index

$$\ell = \arg \min_{r \in I_d \setminus \hat{I}_k} \max_{j \in \{\hat{I}_k, r\}} \pi^j_{\{\hat{I}_k, r\}},$$

as long as all selected coefficients are significant after Bonferroni correction, that is, $\pi^j_{\{\hat{I}_k, \ell\}} \leq \frac{\alpha}{k+1}$ for all $j \in \{\hat{I}_k, \ell\}$.

Step 3. Repeat Step 2 until no more indices can be added to $\hat{I}_k$.

==========

Denote the final set of selected indices after no more index can be added by $\hat{I}$. This set will only contain indices corresponding to predictors that are considered significant in the regression model according to the Bonferroni correction. As in any stepwise variable selection procedure, this algorithm follows a path for inclusion and hence does not evaluate all possible combinations of predictors in the regression model. Such an attempt, called best subset regression, demands high computational cost and is prohibitive when the number of predictors is moderate to large. The concept behind the proposed algorithm is to include a new predictor to the model based on what we call the $\min\max$ criterion of p-values: choose the prospective predictor that when included in the model yields the set of p-values that are the smallest.

Next we extend the forward selection procedure to a backward elimination method with a similar strategy. At each step, given the current predictors in the model, the idea is to compute the sub-model with a predictor removed, and all p-values of the remaining predictors. If all these p-values are low, then the model without that predictor is competitive. After computing this over all predictors in the model, we will chose to drop the predictor whose generating sub-model has the smallest p-values, in fact, the minimum of the maximum p-values. In order to describe the algorithm more precisely, let $\pi^j_{\{\hat{I}_k,-r\}}$ be the p-value from the hypothesis test of the j-th coefficient $H_0 : \beta^j_{\{\hat{I}_k,-r\}} = 0, j \in \{\hat{I}_k, -r\}$, computed from the linear regression model with predictors corresponding to the set of indices $\hat{I}_k \setminus r$, that is, the set $\hat{I}_k$ minus the r index, $r \in \hat{I}_k$. The backward algorithm is as follows.

Backward Elimination Algorithm

==========

Step 1. With the previously selected lags $\hat{I}_k$ (in first iteration $\hat{I}_k$ is composed of the indices of the FULL model), if $\pi^j_{\{\hat{I}_k\}} \leq \frac{\alpha}{k}$ for all $j = 1, \ldots, k$, then stop and retain $\hat{I}_k$. Otherwise go to Step 2.

Step 2. Given the previously selected lags $\hat{I}_k$, remove from $\hat{I}_k$ the index

$$\ell = \arg\min_{r\in\hat{I}_k} \max_{j\in\{\hat{I}_k,-r\}} \pi^j_{\{\hat{I}_k,-r\}}.$$

Step 3. Repeat Steps 1 and 2 until all selected coefficients are significant after the Bonferroni correction, that is, $\pi^j_{\{\hat{I}_k\}} \leq \frac{\alpha}{k}$ for all $j \in \hat{I}_k$.

==========

It is important to notice that when Step 2 is reached (from Step 1), this means that the p-values for the coefficients in the current model do not pass the Bonferroni correction. This means that he current model is overfit in the sense that not all predictors are significant. The algorithm removes a predictor until all predictors left in the model are significant, which means that it seeks the largest model with significant predictors. A combination of both directions, forward and backward, composes the stepwise algorithm, which is described below.

Stepwise Selection Algorithm

==========

Let $\hat{I}_k$ denote the lags selected in the current model.

Step 1. Compute the maximum p-value of the one-step forward model that adds a single predictor from the current model

$$\ell_f = \arg\min_{r\in I_d\setminus\hat{I}_k} \max_{j\in\{\hat{I}_k,r\}} \pi^j_{\{\hat{I}_k,r\}},$$

Step 2. Compute the maximum p-value of the one-step backward model that eliminates a single predictor from the current model.

$$\ell_b = \arg\min_{r\in\hat{I}_k} \max_{j\in\{\hat{I}_k,-r\}} \pi^j_{\{\hat{I}_k,-r\}}.$$

Step 3. Let $\max_{j\in\hat{I}_k} \pi^j_{\hat{I}_k}$ be the maximum p-value of the current model.

If

$$\max_{j\in\{\hat{I}_k,\ell_f\}} \pi^j_{\{\hat{I}_k,\ell_f\}} < \min\left\{\max_{j\in\hat{I}_k} \pi^j_{\hat{I}_k}, \max_{j\in\{\hat{I}_k,-\ell_b\}} \pi^j_{\{\hat{I}_k,-\ell_b\}}\right\}$$

include in $\hat{I}_k$ the index ℓ_f as long as all selected coefficients are significant after Bonferroni correction, that is, $\pi^j_{\{\hat{I}_k,\ell_f\}} \leq \frac{\alpha}{k+1}$ for all $j \in \{\hat{I}_k, \ell_f\}$. If

$$\max_{j\in\{\hat{I}_k,-\ell_b\}} \pi^j_{\{\hat{I}_k,-\ell_b\}} < \min\left\{\max_{j\in\hat{I}_k} \pi^j_{\hat{I}_k}, \max_{j\in\{\hat{I}_k,\ell_f\}} \pi^j_{\{\hat{I}_k,\ell_f\}}\right\}$$

remove from $\hat{I}_k$ the index ℓ_b if the coefficients in the current model are not significant after Bonferroni correction, that is, $\pi^j_{\{\hat{I}_k\}} \leq \frac{\alpha}{k}$ for all $j \in \hat{I}_k$

Step 4. Repeat Steps 1, 2, and 3 until no prospective model can be selected.

==========

Section 3 describes the functions in the package *SignifReg* and their usage for these algorithms, including functions that allow the user to perform one only addition or removal of a predictor in a similar way to that of *add1* or *drop1* in the *stats* library in R, however using the steps of the proposed algorithm.

3. Significance Controlled Variable Selection with package *SignifReg*

The *SignifReg*() function can be used with the input of a few arguments which define a) the scope or predictors, b) the direction of the algorithm, c) the criterion to compare models, and d) the correction on the p-values. The call of the function, as described in the package manual, is

```
SignifReg(fit, scope, alpha = 0.05, direction = "forward",
  criterion = "p-value", adjust.method = "fdr", trace=FALSE)
```

The function *SignifReg*() returns an object inheriting from the class *lm* or *glm* (depending on the input), which can be used for regression analysis with the additional component *steps.info*. The argument 'fit' is an *lm* or *glm* object, which represents the initial model for the variable selection procedure. The user can specify the desired scope, as a *formula*, of predictors to be considered in the same way one would provide it to the the *step* function in the *stats* package. If scope is not provided the function will automatically consider those in the fitted model argument *fit*. When the argument *trace* is equal to TRUE, *SignifReg* will print a table with the information on the candidate models when a predictor is included/excluded and the chosen model for each step. This information table is similar to that of the function *step* in R, however it contains the value of all criteria yielded with the candidate models, including RSS, AIC, BIC, R-adj, PRESS residuals and maximum p-value of the predictors in it, as well as the maximum Variance Inflation Factor of the predictors in the model, and finally a boolean indicating whether the candidate model passed the correction levels indicated in *adjust.method*. The additional component *steps.info* returned by the function is a dataframe with each step taken by the algorithm, which can be used for example to build graphs of the progress or compare with different methods. This component, as well as the output table printed when *trace = TRUE*, differently from other available software, contains a large amount of information to help the user understand and ultimately choose (by changing the parameters of the function) the model and its desired properties.

To illustrate the function *Signifreg*(), first we will consider the following scenario. Suppose we have 100 observations of a set of predictors $\mathbf{X} = (X_1, \ldots, X_{10})$ and a response variable Y. Assume that the true but unknown data generating mechanism is

$$Y = 5X_1 - 3X_3 - X_8 + \epsilon,$$

where ϵ is the independent and identically distributed error with a standard Gaussian distribution, and $X_1, \ldots, X_{10}$ are independent with Uniform distribution on (0,1). We use the following code to generate the dataset.

```
R> n = 100
R> d = 10 #number of variables
R> set.seed(1324)
R> X <- matrix(nrow=n, ncol=d)
R> for(i in 1:d){
+   X[,i] <- runif(n)
+ }
R> Y <- 5*X[,1]  - 3*X[,3]   - X[,8] + rnorm(nrow(X))
R> dat <- data.frame(X,Y)
```

The *Signifreg*() with its default arguments will run the forward algorithm described in Section 2, and will not display each step of the algorithm but only the resulting model.

```
R> install.packages("SignifReg")
R> library(SignifReg)
R>  model<-lm(Y~1, data = dat)
R>  fit<-SignifReg(model, scope = formula(lm(Y ~ ., data = dat)))
R>  summary(fit)

Call:
lm(formula = Y ~ X1 + X3 + X8, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-2.4660 -0.7512  0.0049  0.7419  2.6343

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.3635     0.3393   1.071 0.286674
X1            4.7840     0.3543  13.502  < 2e-16 ***
X3           -3.1744     0.3538  -8.973 2.39e-14 ***
X8           -1.2011     0.3530  -3.402 0.000976 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.045 on 96 degrees of freedom
Multiple R-squared:  0.7745,Adjusted R-squared:  0.7674
F-statistic: 109.9 on 3 and 96 DF,  p-value: < 2.2e-16
```

The selected predictors in this simulation correspond to the true data generating model and each p-value is significant after correcting for FDR. It is possible to examine the steps taken by the algorithm to reach the final model with the *steps.info* component returned within the object returned by the *SignifReg* function. To assess it one can use the following code.

```
R> fit$steps.info

  Step Df  Deviance Resid.Df Resid.Dev      AIC      BIC adj.rsq    PRESS
1      NA                99  464.7142 441.4129 446.6233 0.00000 474.1498
2 + X1  1 -252.2719       98  212.4423 365.1377 372.9532 0.53819 221.2592
3 + X3  1 -95.00518       97  117.4371 307.8610 318.2817 0.74208 125.2498
4 + X8  1 -12.63562       96  104.8015 298.4775 311.5033 0.76743 114.2418
  max_pvalue max.VIF pass fdr correction
1         NA      NA                 NA
2    0.00000      NA               TRUE
3    0.00000 1.01356               TRUE
4    0.00098 1.02879               TRUE
```

Next we examine the steps taken by the algorithm by using the "*trace = TRUE*" argument. As can be seen in the code output below, the function displays a table for each step of the algorithm, containing the criteria and correction results. For instance, each row of the first table shows the AIC, BIC, adj.rsq, PRESS residuals, p-value of the model only with the corresponding predictor displayed in that row. VIF is not displayed in the first table because it is only computable with 2 or more predictors. For the default criterion, which is the smallest p-value, the table is ordered in ascending order of the p-values and the algorithm chooses to include X_1 in the model (note that the p-value 0 is rounded by the software). This model passes the "fdr" correction (default) as we see *TRUE* in the corresponding column. Note that the model only with X_3, and the model only with X_8 would also pass the correction checks. The second table shows the results when X_1 is already included in the model. For example, the row "<none>" is the same row as the one corresponding to X_1 in the first table, which shows the results when only X_1 is in the model. The ninth row of the second table (corresponding to X_2) shows the results for the model with X_1 and X_2. The *max_pvalue* 0.82706 in that row is the largest of π_1 and π_2, which are the p-values of the tests $H_0^1 : \beta_1 = 0$ and $H_0^2 : \beta_2 = 0$, respectively, when only X_1 and X_2 are in the model. Now, on the third row of the *max_pvalue* column we find 0.00215, which is the largest of π_1 and π_8, the p-values of the tests $H_0^1 : \beta_1 = 0$ and $H_0^8 : \beta_8 = 0$, respectively, when only X_1 and X_8 are in the model. From this second table, the algorithm chooses the smallest of these *max_pvalues*, and includes in the model the predictor X_3, since it passes the "fdr" correction. Similarly, the third table shows the inclusion of X_8 in the model and the last table suggests that no other predictor should be included since none passes the correction cut-offs.

```
R> fit <- SignifReg(model, scope = formula(lm(Y ~ ., data = dat)), trace = TRUE)

Call:
lm(formula = Y ~ 1, data = dat)

Coefficients:
(Intercept)
     0.4832

       Resid.Dev      AIC      BIC  adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
+ X1    212.4423 365.1377 372.9532  0.53819 221.2592    0.00000      NA                TRUE
+ X3    332.0272 409.7924 417.6079  0.27823 346.0595    0.00000      NA                TRUE
+ X8    422.3509 433.8543 441.6698  0.08189 440.6285    0.00227      NA                TRUE
+ X2    454.4266 441.1743 448.9898  0.01216 474.0537    0.13957      NA               FALSE
+ X7    458.1016 441.9798 449.7953  0.00417 479.0101    0.23717      NA               FALSE
+ X9    459.1888 442.2168 450.0323  0.00181 479.9711    0.28018      NA               FALSE
+ X10   459.9271 442.3775 450.1930  0.00020 481.0354    0.31500      NA               FALSE
+ X6    461.2542 442.6656 450.4811 -0.00268 479.4119    0.39332      NA               FALSE
+ X5    462.9987 443.0431 450.8586 -0.00648 480.4387    0.54818      NA               FALSE
+ X4    464.6790 443.4054 451.2209 -0.01013 485.8172    0.93154      NA               FALSE
<none>  464.7142 441.4129 446.6233  0.00000 474.1498         NA      NA                  NA

Call:
lm(formula = Y ~ X1, data = dat)

Coefficients:
(Intercept)           X1
     -2.123        5.310

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
<none>  212.4423 365.1377 372.9532 0.53819 221.2592    0.00000      NA                TRUE
+ X3    117.4371 307.8610 318.2817 0.74208 125.2498    0.00000 1.01356                TRUE
+ X8    192.6983 357.3833 367.8039 0.57679 206.1638    0.00215 1.01787                TRUE
+ X7    195.5304 358.8423 369.2629 0.57057 207.9424    0.00466 1.00927                TRUE
+ X5    209.2887 365.6421 376.0628 0.54035 222.2828    0.22961 1.00086               FALSE
+ X10   211.4831 366.6852 377.1059 0.53553 226.0905    0.50872 1.00585               FALSE
+ X6    212.1464 366.9984 377.4191 0.53408 224.8924    0.71383 1.00693               FALSE
+ X9    212.2894 367.0657 377.4864 0.53376 226.1957    0.79211 1.01551               FALSE
```

```
+ X2    212.3372 367.0883 377.5089 0.53366 225.9079    0.82706 1.03421               FALSE
+ X4    212.3934 367.1147 377.5354 0.53354 225.1238    0.88155 1.00000               FALSE

Call:
lm(formula = Y ~ X1 + X3, data = dat)

Coefficients:
(Intercept)           X1           X3
    -0.2802       4.9307      -3.2860

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
<none>  117.4371 307.8610 318.2817 0.74208 125.2498    0.00000 1.01356                TRUE
+ X8    104.8015 298.4775 311.5033 0.76743 114.2418    0.00098 1.02879                TRUE
+ X7    114.1170 306.9931 320.0190 0.74676 124.0867    0.09793 1.07576               FALSE
+ X10   116.5063 309.0652 322.0911 0.74146 126.8795    0.38334 1.01939               FALSE
+ X2    116.6091 309.1535 322.1793 0.74123 127.0635    0.41107 1.05054               FALSE
+ X4    117.0591 309.5385 322.5644 0.74023 127.7700    0.57896 1.02103               FALSE
+ X6    117.2783 309.7257 322.7515 0.73975 127.7522    0.71924 1.02021               FALSE
+ X9    117.2996 309.7438 322.7696 0.73970 128.5259    0.73797 1.02901               FALSE
+ X5    117.4242 309.8500 322.8759 0.73942 127.7722    0.91841 1.04399               FALSE

Call:
lm(formula = Y ~ X1 + X3 + X8, data = dat)

Coefficients:
(Intercept)           X1           X3           X8
     0.3635       4.7840      -3.1744      -1.2011

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
<none>  104.8015 298.4775 311.5033 0.76743 114.2418    0.00098 1.02879                TRUE
+ X7    102.6042 298.3586 313.9896 0.76991 114.0549    0.15705 1.08150               FALSE
+ X4    103.6409 299.3638 314.9949 0.76759 115.4774    0.30495 1.04370               FALSE
+ X2    103.7733 299.4916 315.1226 0.76729 115.7464    0.33443 1.06719               FALSE
+ X10   103.7847 299.5026 315.1336 0.76727 115.5637    0.33713 1.03485               FALSE
+ X6    104.7340 300.4131 316.0441 0.76514 116.6094    0.80511 1.03469               FALSE
+ X9    104.7577 300.4357 316.0667 0.76508 117.2160    0.84251 1.04290               FALSE
+ X5    104.7649 300.4426 316.0736 0.76507 116.6753    0.85592 1.05352               FALSE
```

Next we will use the backward algorithm to perform variable selection with a different dataset. In this case we consider the same 10 predictors but with a different model

$$Y = 1.5X_1 + 1.5X_2 + 2.3X_3 - 1.2X_5 - 3.2X_6 + 1.9X_8 + 1.8X_9 - 4.2X_{10} + \epsilon,$$

where ϵ and $\mathbf{X}$ are generated similarly to the previous model. We run the selection with the Bonferroni correction as follows

```
R> n = 100
R> d = 10 #number of variables
R> set.seed(1324)
R> X <- matrix(nrow=n, ncol=d)
R> for(i in 1:d){
+      X[,i] <- runif(n)
+  }
R> Y = 1.5*X[,1] + 1.5*X[,2] + 2.3*X[,3] - 1.2*X[,5] - 3.2*X[,6]
+           + 1.9*X[,8] +  1.8*X[,9] - 4.2*X[,10] + rnorm(nrow(X))
R> dat <- data.frame(X,Y)
R> model<-lm(Y~.,dat)
R> fit <- SignifReg(model, direction = "backward",
       adjust.method = "bonferroni", trace = TRUE)
```

```
Call:
lm(formula = Y ~ ., data = dat)

Coefficients:
(Intercept)           X1           X2           X3           X4           X5
     0.5080       1.2150       1.0693       2.2203       0.4080      -0.9794
         X6           X7           X8           X9          X10
    -3.2039      -0.4757       1.6853       2.0452      -4.4909

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
- X4    100.3357 306.1228 334.7797 0.77685 125.0811    0.19084 1.22518               FALSE
- X7    100.8037 306.5882 335.2451 0.77581 125.9849    0.25521 1.11424               FALSE
<none>   99.1637 306.9479 338.2099 0.77698 126.3626    0.30786 1.23670               FALSE
- X2    108.8823 314.2974 342.9543 0.75784 135.5839    0.31719 1.23276               FALSE
- X5    106.0102 311.6243 340.2811 0.76423 132.8155    0.39209 1.17421               FALSE
- X10   253.3652 398.7539 427.4108 0.43650 315.1725    0.41972 1.18667               FALSE
- X1    110.8525 316.0907 344.7476 0.75346 138.4006    0.50833 1.20431               FALSE
- X8    123.1441 326.6062 355.2631 0.72612 152.8000    0.55620 1.21846               FALSE
- X3    137.0207 337.2839 365.9407 0.69526 169.0151    0.68329 1.15557               FALSE
- X9    128.1771 330.6119 359.2688 0.71493 158.1410    0.87184 1.18591               FALSE
- X6    177.3869 363.1040 391.7609 0.60548 219.0695    0.89303 1.23006               FALSE

Call:
lm(formula = Y ~ X1 + X2 + X3 + X5 + X6 + X7 + X8 + X9 + X10,
    data = dat)

Coefficients:
(Intercept)           X1           X2           X3           X5           X6
     0.6534       1.2451       1.1232       2.2651      -0.9560      -3.1643
         X7           X8           X9          X10
    -0.5145       1.7379       1.9892      -4.4996

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
- X7    102.2723 306.0346 332.0863 0.77504 125.0929    0.00677 1.09575               FALSE
- X5    106.8814 310.4426 336.4943 0.76490 130.8349    0.06748 1.16554               FALSE
- X10   255.2151 397.4814 423.5331 0.43862 310.0643    0.11083 1.17404               FALSE
<none>  100.3357 306.1228 334.7797 0.77685 125.0811    0.19084 1.22518               FALSE
- X2    111.2880 314.4829 340.5346 0.75521 135.2176    0.26128 1.22290               FALSE
- X1    112.6862 315.7314 341.7831 0.75213 137.4083    0.44396 1.19555               FALSE
- X8    126.3512 327.1773 353.2289 0.72207 153.3868    0.47857 1.21050               FALSE
- X6    177.4227 361.1242 387.1759 0.60974 214.9185    0.63876 1.21656               FALSE
- X3    140.2609 337.6211 363.6728 0.69148 168.8406    0.75689 1.13088               FALSE
- X9    128.3008 328.7084 354.7601 0.71779 155.1020    0.84023 1.16649               FALSE

Call:
lm(formula = Y ~ X1 + X2 + X3 + X5 + X6 + X8 + X9 + X10, data = dat)

Coefficients:
(Intercept)           X1           X2           X3           X5           X6
     0.6719       1.1684       1.1028       2.1260      -1.0707      -3.1222
         X8           X9          X10
     1.6860       1.8746      -4.6024

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
- X9    128.3584 326.7533 350.1999 0.72073 152.3350    0.00534 1.08511                TRUE
- X5    110.9035 312.1367 335.5833 0.75871 133.0201    0.00660 1.09565                TRUE
<none>  102.2723 306.0346 332.0863 0.77504 125.0929    0.00677 1.09575               FALSE
- X2    112.8507 313.8773 337.3238 0.75447 134.5197    0.01615 1.08611               FALSE
- X1    113.4183 314.3789 337.8255 0.75323 135.4409    0.01950 1.09149               FALSE
- X8    127.0542 325.7321 349.1786 0.72357 151.0423    0.02035 1.09122               FALSE
- X3    140.4095 335.7270 359.1735 0.69451 166.0706    0.03309 1.08462               FALSE
```

```
- X10   271.3677 401.6182 425.0647 0.40958 323.6463    0.03340 1.08781               FALSE
- X6    177.8553 359.3677 382.8142 0.61304 211.0353    0.04831 1.09118               FALSE

Call:
lm(formula = Y ~ X1 + X2 + X3 + X5 + X6 + X8 + X10, data = dat)

Coefficients:
(Intercept)           X1           X2           X3           X5           X6
      1.631        1.345        1.143        2.069       -1.465       -2.873
         X8          X10
      1.622       -4.677

       Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF pass fdr correction
<none>  128.3584 326.7533 350.1999 0.72073 152.3350    0.00534 1.08511                TRUE
- X10   303.3075 410.7454 431.5868 0.34718 353.4149    0.00678 1.07578                TRUE
- X2    139.7209 333.2353 354.0767 0.69928 161.5338    0.00927 1.05370               FALSE
- X8    151.3399 341.2235 362.0648 0.67427 176.4146    0.01322 1.06911               FALSE
- X5    145.2680 337.1287 357.9701 0.68734 168.7815    0.01492 1.08405               FALSE
- X3    164.5260 349.5775 370.4189 0.64589 188.8564    0.02166 1.07095               FALSE
- X6    193.5558 365.8273 386.6686 0.58341 223.6356    0.03205 1.07832               FALSE
- X1    143.2625 335.7385 356.5799 0.69165 166.4775    0.03480 1.04200               FALSE
```

The first model displayed by the *SignifReg* function is the initial model, which for this backward case is the full model with all 10 predictors. It is important to clearly interpret the tables presented in the backward algorithm, which differ from the ones shown in the forward algorithm. The row corresponding to "<none>", shows the criteria for the full model. Additionally, the following rows show the criteria when the corresponding variable is excluded from the model. For example, when X_1 is excluded from the model, that is the model containing $X_2, \ldots, X_{10}$ we obtain RSS equal to 110.8525 and adjusted R-squared equal to 0.75346. Recalling that we are using the Bonferroni correction, the model without X_1 does not have all 9 remaining p-values less than the Bonferroni correction, and thus "FALSE" is seen on the row corresponding to X_1 of the Bonferroni column in this first table. The first predictor eliminated by the backward algorithm is X_4, since the model without it has the smallest "max_pvalue", that is, the maximum of the p-values of the model with $X_1, \ldots, X_3, X_5, \ldots, X_{10}$ is 0.19084, the smallest compared to other models (with 9 covariates) with other predictors removed. The last table displayed by the algorithm shows that two models are candidates (they pass the Bonferroni correction): the first with X_5 removed and the second with X_9 removed. This means that the model without X_5 for example, which would be composed of predictors $X_j, j = 1, 2, 3, 6, 8, 9, 10$, would have all predictors with p-values below the Bonferroni cut-offs. The model without X_9 is chosen since it produces the smallest "max_pvalue", so the final model is chosen to be the one with the predictors $X_j, j = 1, 2, 3, 5, 6, 8, 10$.

If one is interested in exploring the addition or removal of a predictor for a given regression model while keeping the significance of all added predictors, the functions *add1SignifReg*() and *drop1SignifReg*() will show how each of the criteria and correction methods perform according an individual step of the proposed algorithms. This is similar to the *add* and *drop* functions in the *stats* package in R.

To demonstrate the use of these functions, suppose we are interested in adding or removing a predictor from the generalized linear regression model with binary response (logistic regression) for predicting death using the heart failure dataset available at
https://archive.ics.uci.edu/ml/datasets/Heart+failure+clinical+records. This dataset contains the clinical features of 299 patients who had heart failure and the event recorded was whether the patient died or not. Features include age, sex, if the patient had diabetes, number of platelets in the blood, among others. We consider an initial logistic regression model only with *age*. The following code uses the function *add1SignifReg*() to assess the possible predictors that can be added, together with all the information of the model containing *age* and each prospective predictor when *print.step = TRUE*

```
R> data_heart = read.csv("https://archive.ics.uci.edu/ml/machine-learning-databases/
       00519/heart_failure_clinical_records_dataset.csv", header = TRUE)
```

```
R> fit = glm(DEATH_EVENT ~ age, data = data_heart, family = "binomial")
R> add1SignifReg(fit, scope = formula(lm(DEATH_EVENT~., data = data_heart)),
    print.step = TRUE)

                          Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue
+ ejection_fraction         327.4015 333.4015 344.5029      NA 334.4432    0.00000
<none>                      355.9928 359.9928 367.3937      NA 360.7789    0.00007
+ serum_creatinine          334.1227 340.1227 351.2240      NA 341.9026    0.00074
+ serum_sodium              345.5911 351.5911 362.6924      NA 353.2427    0.00307
+ time                      271.4643 277.4643 288.5656      NA 277.2341    0.00426
+ creatinine_phosphokinase  353.8454 359.8454 370.9467      NA 361.7142    0.18118
+ high_blood_pressure       354.9659 360.9659 372.0672      NA 362.2801    0.35553
+ anaemia                   355.3216 361.3216 372.4229      NA 362.6095    0.45519
+ platelets                 355.4946 361.4946 372.5959      NA 362.9402    0.52004
+ diabetes                  355.7601 361.7601 372.8614      NA 363.0023    0.66026
+ sex                       355.8342 361.8342 372.9356      NA 363.1491    0.71675
+ smoking                   355.9059 361.9059 373.0073      NA 363.1327    0.78829
                          max.VIF pass fdr correction
+ ejection_fraction        1.05168               TRUE
<none>                         NA               TRUE
+ serum_creatinine         1.01102               TRUE
+ serum_sodium             1.00053               TRUE
+ time                     1.00002               TRUE
+ creatinine_phosphokinase 1.01864              FALSE
+ high_blood_pressure      1.00296              FALSE
+ anaemia                  1.00183              FALSE
+ platelets                1.00006              FALSE
+ diabetes                 1.01892              FALSE
+ sex                      1.00744              FALSE
+ smoking                  1.00056              FALSE

Call:  glm(formula = DEATH_EVENT ~ age + ejection_fraction, family = "binomial",
    data = data_heart)

Coefficients:
      (Intercept)               age  ejection_fraction
         -1.79490           0.05582           -0.06586

Degrees of Freedom: 298 Total (i.e. Null);  296 Residual
Null Deviance:     375.3
Residual Deviance: 327.4  AIC: 333.4
```

Note that in this case, there are a few predictors that are candidates for entering the model, namely *ejection_fraction*, *serum_creatinine*, *serum_sodium*, and *time* (models with each of these predictors included pass the FDR correction). The output of the function *add1SignifReg*() is the *glm* object with *age* and *ejection_fraction*, with the additional component *step.info* for the information about why it was selected. The predictor *ejection_fraction* is chosen to be added because the p-values of the tests $H_0^1 : \beta_1 = 0$ and $H_0^2 : \beta_2 = 0$, respectively, when only *age* and *ejection_fraction* are in the model are both equal to or smaller than 0.00001, that is, the max of these two p-values is smaller than 0.00001. This is the smallest of all *max_pvalues*, computed from models with *age* and the other prospective predictors. It is important to notice that the default of the *SignifReg*() function is to use the FDR correction, so that the model with *age* and *ejection_fraction* is only chosen because both p-values are below the cutoff values of the FDR.

Now suppose we are interested in dropping a predictor and checking which model, after dropping the predictor, has the best performance in terms of *max_p-value* while all remaining predictors are significant according to FDR or Bonferroni corrections. The function *drop1SignifReg*() will display all the information of the prospective models when each predictor is removed, will perform one step of the backward elimination algorithm in the *SignifReg* function and will return the model with a predictor removed (if the current model does not pass the correction). To

demonstrate its use, assume we have the logistic regression model with response variable death and predictors *age, ejection_fraction, serum_creatinine*, and *platelets*. The following code uses *drop1SignifReg* to assess the possible predictors that can be removed and displays all the information of the prospective models. Removing *platelets* yields a model with the lowest maximum p-value, so the function returns the *glm* object with *age*, *ejection_fraction*, and *serum_creatinine* only.

```
R> fit = glm(DEATH_EVENT ~ age+ ejection_fraction + serum_creatinine+ platelets,
   data = data_heart, family = "binomial")
R> drop1SignifReg(fit, print.step = TRUE)
                    Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF
- platelets          305.2827 313.2827 328.0844      NA 314.8040    0.00002 1.07064
- ejection_fraction  333.8877 341.8877 356.6895      NA 344.2673    0.64901 1.01276
- serum_creatinine   327.2865 335.2865 350.0883      NA 336.8010    0.74770 1.05494
- age                324.2484 332.2484 347.0502      NA 333.8983    0.79726 1.00367
<none>               305.2628 315.2628 333.7650      NA 317.1645    0.89000 1.07395
                    pass fdr correction
- platelets                       TRUE
- ejection_fraction              FALSE
- serum_creatinine               FALSE
- age                            FALSE
<none>                           FALSE

Call:  glm(formula = DEATH_EVENT ~ age + ejection_fraction + serum_creatinine,
    family = "binomial", data = data_heart)

Coefficients:
      (Intercept)               age  ejection_fraction   serum_creatinine
         -2.35306           0.05173           -0.07000            0.66592

Degrees of Freedom: 298 Total (i.e. Null);  295 Residual
Null Deviance:     375.3
Residual Deviance: 305.3  AIC: 313.3
```

Finally, to see the prospective models when adding or dropping a predictor one can use the functions *add1summary* and *drop1summary* from the package *SignifReg*. These functions are informative only and do not return an *lm (glm)* object. As an example we check the prospective models when adding *anaemia*, *high_blood_pressure* or *cyl* to the model with *serum_sodium* with the following code.

```
R> fit = glm(DEATH_EVENT ~ age, data = data_heart, family = "binomial")
R> add1summary(fit, scope = ~.+ anaemia + high_blood_pressure+ serum_sodium)
                     Resid.Dev      AIC      BIC adj.rsq    PRESS max_pvalue max.VIF
<none>                355.9928 359.9928 367.3937      NA 360.7789    0.00007      NA
+ serum_sodium         345.5911 351.5911 362.6924      NA 353.2427    0.00307 1.00053
+ high_blood_pressure  354.9659 360.9659 372.0672      NA 362.2801    0.35553 1.00296
+ anaemia              355.3216 361.3216 372.4229      NA 362.6095    0.45519 1.00183
                     pass fdr correction
<none>                             TRUE
+ serum_sodium                     TRUE
+ high_blood_pressure              FALSE
+ anaemia                         FALSE
```

Note that the call of the *add1summary* function includes the model, the scope, the significance level, the p-value correction method (default is “fdr”) and finally the column by which to sort the table (which needs to be one of the criteria available - default is “p-value”). Similar code and output can be see when using *drop1summary*.

4. Simulations

In this section we assess the finite sample performance of the variable selection methods available in the package *SignifReg* and, for comparison purposes, the results of the classical algorithms from other available statistical software are also analyzed. Each simulation was run 1000 times with sample size of $n = 500$. The results presented here with moderately large sample sizes aim at modern applications, where available data storage and recording technologies usually allow for large samples. Nevertheless, simulations with $n = 100$, which are not reported here for the sake of brevity and space, suggest that the results are comparatively similar to the ones presented in this section for the methods considered.

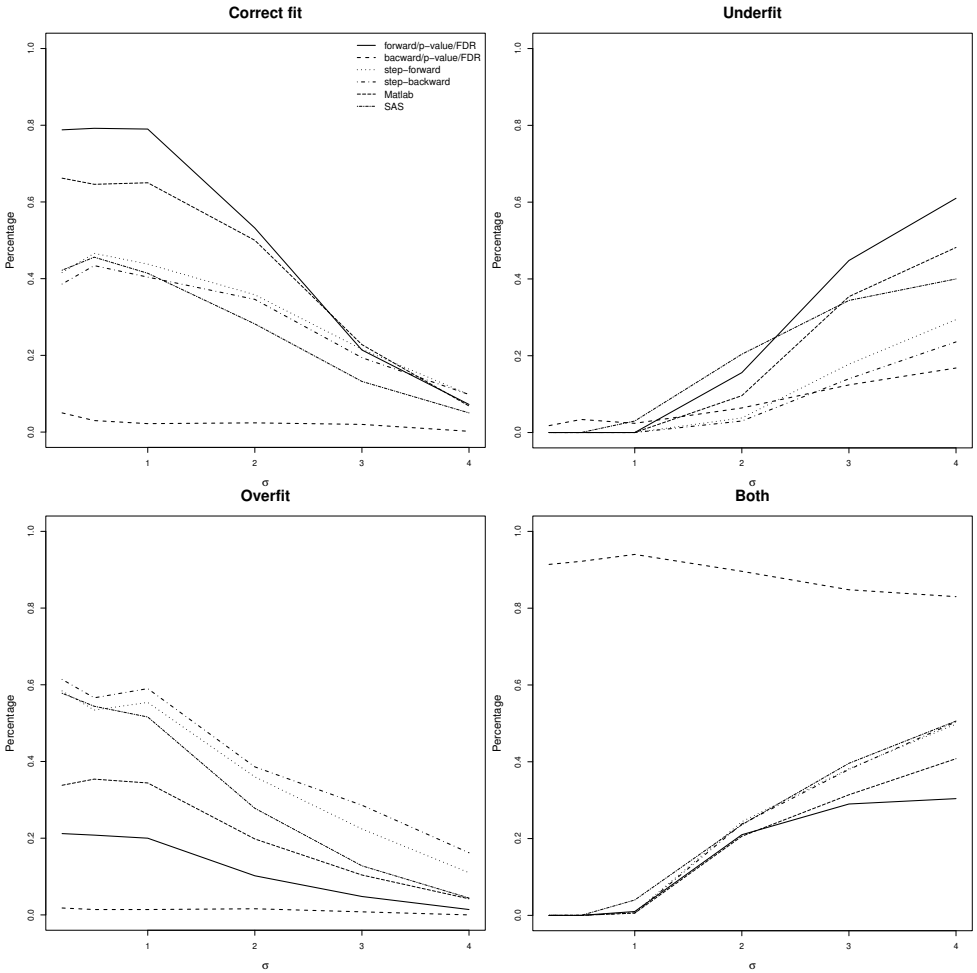

Figure 1. Proportion of correct fit (top left), underfit (top right), overfit (bottom left), and under/over (bottom right) out of 1000 simulation runs for 10 predictors with multivariate Normal distribution.

In this simulation study we generate the data from the linear model $Y = \mathbf{X}\boldsymbol{\beta} + \epsilon$, where $\epsilon \sim N(0, \sigma^2)$ are i.i.d., for a range of values of the variance σ^2, $\mathbf{X} = (X_1, \ldots, X_d)$ for $d = 10$ or 30, and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)$. For the scenarios where $d = 10$, the coefficients are set to $\beta_1 = 4, \beta_4 = 1, \beta_5 = -2, \beta_8 = -0.5, \beta_{10} = 1.3$ and $\beta_j = 0$ for $j \notin \{1, 4, 5, 8, 10\}$. For the scenarios where $d = 30$, the coefficients are set to $\beta_1 = 3, \beta_{10} = -1, \beta_{11} = -1, \beta_{15} = -4, \beta_{25} = 0.4$ and $\beta_j = 0$ for $j \notin \{1, 10, 11, 15, 25\}$. We consider independent and dependent predictors: for the independent case $X_j, j = 1, \ldots, d$ are generated independently from $U(0, 5)$, while for the dependent

case the predictors are generated from a multivariate Normal distribution with zero mean and covariance matrix $\Sigma = (\sigma_{k\ell})_{d\times d}$ with $\sigma_{k\ell} = 0.9^{|k-\ell|}$ for all $k, \ell \in 1, \ldots, d$. We assess the performance of the variable selection methods by computing the percentage of the simulation runs where a) *Correct fit*: all five active predictors were selected, b) *Underfit*: only a few of the active predictors were selected and no non-active predictor was selected, c) *Overfit*: all five active predictors and some non-active predictors were selected, and d) *Under/Over*: some active predictors were not selected and some non-active predictors were selected. These measures are computed for difference values of the variance σ, that is, with increasing levels of difficulty given the increasing noise.

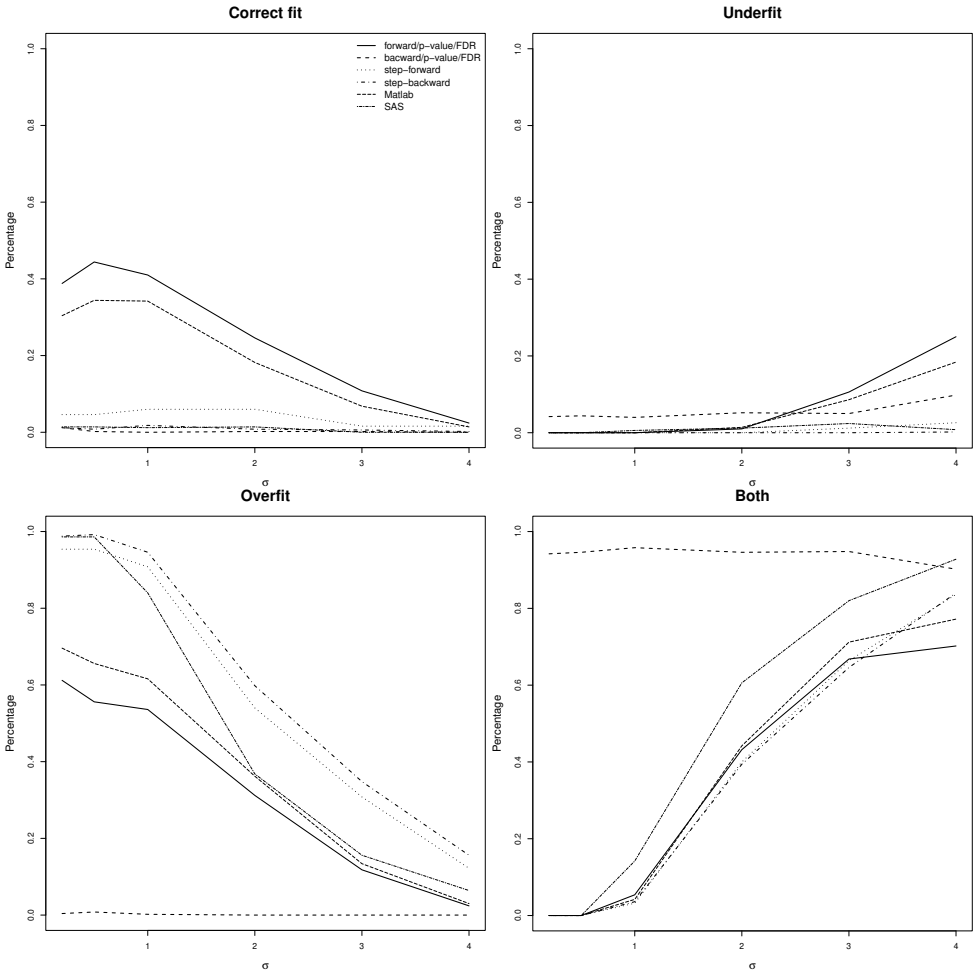

Figure 2. Proportion of correct fit (top left), underfit (top right), overfit (bottom left), and under/over (bottom right) out of 1000 simulation runs for 30 predictors with multivariate Normal distribution.

We compare the results of the proposed method with those from the *step* function in R, *stepwisefit* in Matlab (*https://www.mathworks.com/help/stats/stepwisefit.html*), and PROC REG in SAS with SELECT option. The *step* function in R chooses the model based on AIC comparisons and was set to start with the null model for forward and full model for backward. The *stepwisefit* in Matlab was used with steps in both directions and the criterion to add/drop predictors was the sse. For SAS, we used the *stepwise* option in the *PROC GLMSELECT* procedure with the significance level (F statistic) of each predictor as the criteria to add or drop variables. The procedure terminates when adding any effect to the model increases the predicted residual sum of squares. Results of other

software such as Minitab and SPSS are not included because of their graphic user interface (GUI), which bring challenges to program Monte Carlo simulations such as the ones in this section.

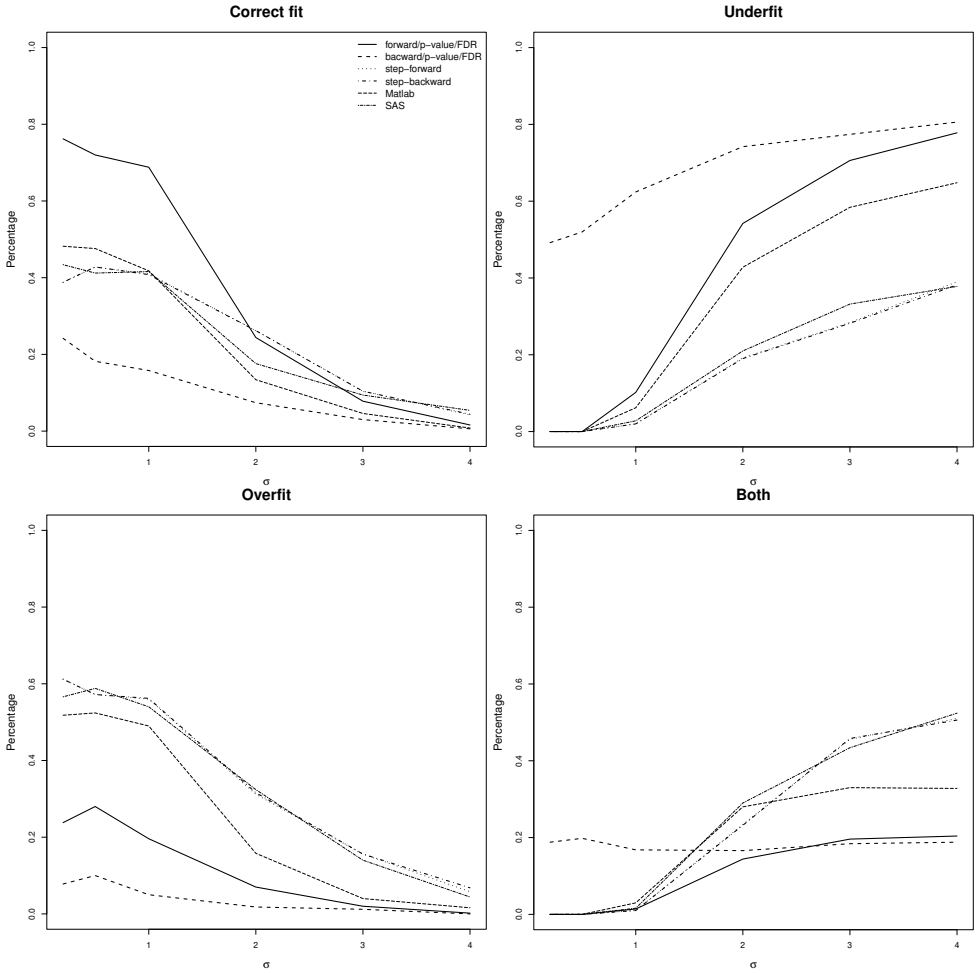

Figure 3. Proportion of correct fit (top left), underfit (top right), overfit (bottom left), and under/over (bottom right) out of 1000 simulation runs for 10 independent predictors with Uniform distribution.

Figures 1 and 2 show the percentage of correct, underfit, overfit, and under/overfit out of the 1000 Monte Carlo simulation runs for the multivariate Normal covariates with $d = 10$ and $d = 30$ respectively. For small to moderate values of σ, the proposed forward procedure has the highest percentage of correct fit followed by Matlab's algorithm. In the case of $d = 30$ virtually all other competitors achieve almost 0 correct fit rates. For low values of σ and $d = 10$, the proposed forward algorithm has low rates of overfit, underfit and under/overfit, while more overfit is seen when $d = 30$ as expected. Also expected is the fact that the underfit rates for the proposed method tend to be higher when σ^2 is large, since requiring small p-values (under FDR or Bonferroni thresholds) is restrictive in selecting predictors, however, overfit and under/overfit rates are generally the lowest for all values of σ. The proposed backward procedure does not seem to perform well in this scenario with a high percentage of under/overfit, that is, some active predictors were dropped from the model while some non-active predictors were kept. The *step* function in R, *stepwisefit* in Matlab, and *PROC GLMSELECT* in SAS had reasonable percentages of correct fit for $d = 10$ but performed poorly for $d = 30$, mostly overfiting the model for small values of σ and under/overfiting the model for larger values of σ. The results in Figures 3 and 4, which correspond to the percentage

of correct, underfit, overfit, and under/overfit for the independent Uniform covariates with $d = 10$ and $d = 30$ respectively, show rates with patterns similar to those in the multivariate Normal case. Overall, the results suggest that the proposed forward selection algorithm with p-value corrections is preferable for finding the correct set of predictors especially if the variability of the error is relatively small to moderate, with the expected rate of underfit climbing faster than other methods given its restriction (Bonferroni or FDR) in allowing new variables to be added. The very high rates of overfit in the methods applied from other software, especially for small values of σ, would yield models with unnecessary predictors and thus consume unnecessary degrees of freedom, which in consequence can cause inaccurate inference.

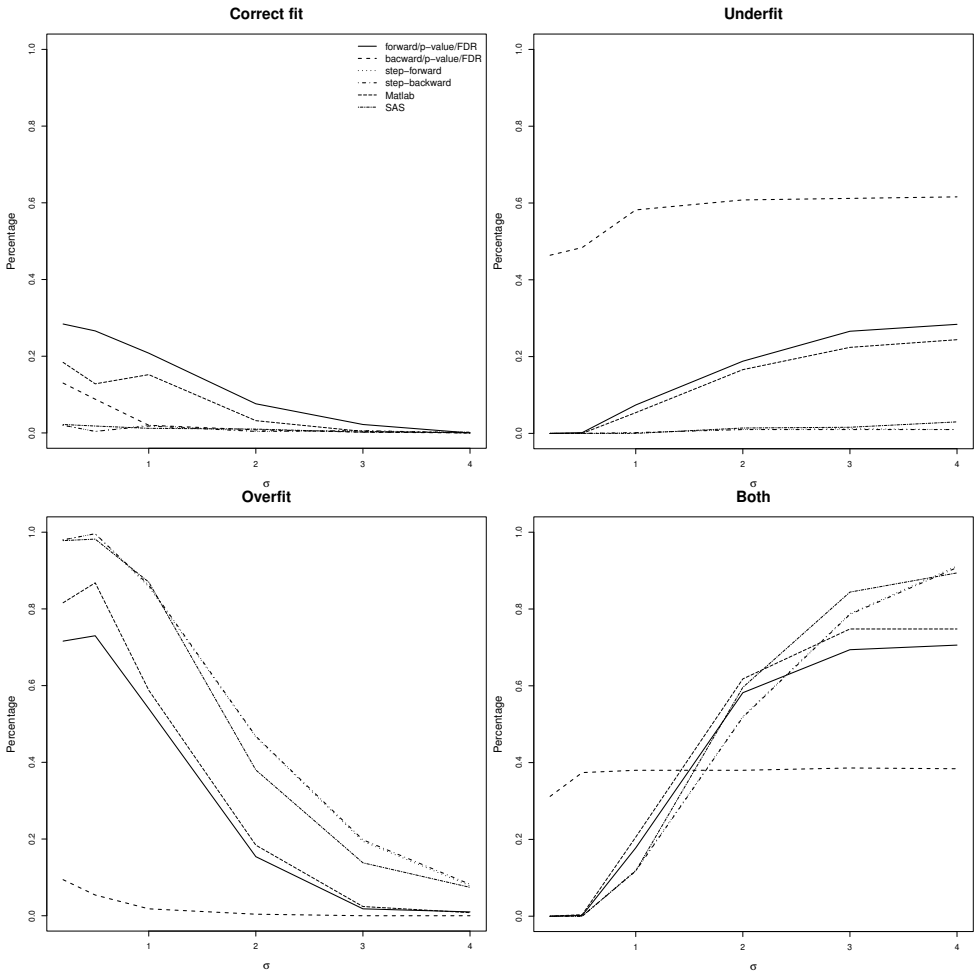

Figure 4. Proportion of correct fit (top left), underfit (top right), overfit (bottom left), and under/over (bottom right) out of 1000 simulation runs for 30 independent predictors with Uniform distribution.

5. Real Data

In this section we assess the performance of the proposed method in comparison with competing statistical software on two benchmark real datasets for regression: *medical costs* ([11]) and *US macroeconomic data* ([12]). The dataset *medical costs* was originally introduced in Brett Lantz' book *Machine Learning with R* and is available on

Medical Costs						
	forward/p-value/FDR	backward/p-value/FDR	step-forward	step-backward	Matlab	SAS
Age	X	X	X	X	X	X
Sex						
bmi	X	X	X	X	X	X
Children	X	X	X	X	X	X
Smoker	X	X	X	X	X	X
Region			X	X		X

Table 1. Results of variable selection based on each algorithm for the *Insurance Cost* dataset.

US Macroeconomic Data						
	forward/p-value/FDR	backward/p-value/FDR	step-forward	step-backward	Matlab	SAS
GNP Deflator						
GNP	X		X	X	X	X
Uneployment	X	X	X	X	X	X
Armed Forces		X	X	X		X
Population						
Year		X	X	X		

Table 2. Results of variable selection based on each algorithm for the *US Macroeconomic Data* dataset.

Medical Costs						
	forward/p-value/FDR	backward/p-value/FDR	step-forward	step-backward	Matlab	SAS
AIC	27114.04	27114.04	27113.66	27113.66	27114.04	27113.66
Adj R^2	0.7489434	0.7489434	0.7495727	0.7495727	0.7489434	0.7495727

Table 3. Selected model characteristics based on each algorithm for the *Insurance Cost* dataset.

US Macroeconomic Data						
	forward/p-value/FDR	backward/p-value/FDR	step-forward	step-backward	Matlab	SAS
AIC	250.4944	236.5756	231.655	231.655	250.4944	248.3174
Adj R^2	0.9776784	0.9910588	0.993671	0.993671	0.9776784	0.9813745

Table 4. Selected model characteristics based on each algorithm for the *US Macroeconomic Data* dataset.

Kaggle (https://www.kaggle.com/datasets/mirichoi0218/insurance). It contains medical cost information of 1,338 individuals and 7 variables: age, sex, bmi, number of children, whether the person is a smoker or not, region where the person lives, and premium insurance charges. The goal with this dataset is to apply variable selection methods to a linear regression model with charges as a response variable and the other six variables as predictors. The second dataset, *US macroeconomic*, was provided by Longley (1967) [12] and is available in the *datasets* library in the software *R*. It is a relatively small dataset containing 16 observations and 7 highly correlated variables: total employment, GNP deflator, GNP, number of unemployed, size of armed forces, population, and year. For this dataset we consider total employment as the response variable.

Table 1 and Table 2 summarize the selected models based on each algorithm considered for the medical costs dataset and the US macroeconimic dataset respectively, while Table 3 and Table 4 report the AIC and adjusted R^2. As expected, in general *SignifReg* selects the most parsimonious models given its strict nature of controlling p-values. For the medical costs dataset, both forward and backward algorithms of *SignifReg* select the same model containing age, bmi, children, and smoker status, which is also the result of Matlab's algorithm. SAS and R's selected models, besides these predictors, also include region. The parsimonious model selected by the proposed algorithm yields AIC and Adjusted R^2 virtually identical to the larger model selected by SAS and R step function, differing at most of the order of 0.01%. The results for the US macroeconomic data follow the same pattern, however the parsimony of the model selected by the proposed algorithm is more pronounced. Only 2 variables, GNP and unemployment, are selected by the proposed forward procedure and Matlab, while SAS and R's step

function select 4 predictors. Similarly to the medical costs data results, AIC and adjusted R^2 of the models selected by the proposed algorithm are close to the ones obtained by SAS and R's step function. However, the models chosen by the competing algorithms include too many predictors, which are in this case known to be multi-colinear, and yield variance inflation factors (VIF) as high as 638.13, while the parsimonious model selected by the proposed algorithm yields a model with variance inflation factor no more than 1.57. Such a parsimonious model is preferable to the larger model, especially due to the inflation of the standard deviation caused by the high VIFs in the larger model.

6. Conclusion

Implementations of variable selection algorithms are available in a variety of statistical software. However, the standard approach, which selects variables according to a criterion such AIC or BIC, may yield a final model with insignificant predictors. In this paper, we proposed new algorithms that control for the significance of the included predictors at each step of the selection process, either backward or forward. The control can be performed using the commonly available p-value correction cut-offs, including for example the well known "hochberg" [5], "BH" or "fdr" [1], and "BY" [2]). The algorithms proposed in here are implemented in the R software *SignifReg*() and is available for download at https://cran.r-project.org/web/packages/SignifReg/index.html. In extensive simulation scenarios and two real benchmark datasets for regression we evaluated the performance of the proposed algorithms compared to other variable selection functions in R, Matlab and SAS. Overall, the results suggest that the proposed forward selection algorithm with p-value corrections is performs better when selecting variables especially if the variability of the error is relatively small to moderate.

REFERENCES

1. Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.
2. Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.
3. N. Draper and H. Smith. *Applied regression analysis*. John Wiley & Sons, New York, 1966.
4. M. Efroymson. Stepwise regression: a backward and forward look. *Eastern Regional Meetings of the Institute of Mathematical Statistics*, 1966.
5. Y. Hochberg. A sharper bonferroni procedure for multiple tests of significance. *Biometrika*, 75:800–803, 1988.
6. Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. ISSN 03036898, 14679469.
7. G. Hommel. A stagewise rejective multiple test procedure based on a modified Bonferroni test. *Biometrika*, 75(2):383–386, 06 1988. ISSN 0006-3444.
8. Cho-Ying Huang, Hsin-Lin Wei, Jiann-Yeou Rau, and Jyun-Ping Jhan. Use of principal components of uav-acquired narrow-band multispectral imagery to map the diverse low stature vegetation fapar. *GIScience & Remote Sensing*, 56(4):605–623, 2019.
9. A. B. Imran, K. Khan, N. Ali, N. Ahmad, A. Ali, and K. Shah. Narrow band based and broadband derived vegetation indices using sentinel-2 imagery to estimate vegetation biomass. *Global Journal of Environmental Science and Management*, 6:97–108, 2020.
10. Josely Correa Koury, Maria Almeida Ribeiro, Fabia Albernaz Massarani, Filomena Vieira, and Elisabetta Marini. Fat-free mass in adolescent athletes: Accuracy of bioimpedance equations and identification of new predictive equations. *Nutrition*, 60:59 – 65, 2019. ISSN 0899-9007.
11. Brett Lantz. Machine Learning with R. *Packt Publishing*, Birmingham, Mumbai, 2013.
12. James W. Longley. An appraisal of least-squares programs from the point of view of the user. *Journal of the American Statistical Association*, 62: 819 – 841, 1967.
13. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL https://www.R-project.org/.
14. S. Sarvepalli, C.A. Burke, M. Monachese, R. Lopez, B.H. Leach, L. Laguardia, M. O'Malley, M.F. Kalady, and J.M. Church. Web-based model for predicting time to surgery in young patients with familial adenomatous polyposis: An internally validated study. *American Journal of Gastroenterology*, 113:1881 – 1890, 2018.
15. S. Walter and H. Tiemeier. Variable selection: current practice in epidemiological studies. *European Journal of Epidemioloy*, 24: 733–736, 2009.
16. Adriano Zambom and Jongwook Kim. Consistent significance controlled variable selection in high-dimensional regression. *STAT*, 7, 2018.