lmSubsets: Efficient Computation of Variable-Subsets Linear Regression in R

Marc Hofmann
`TODO: affiliation`

Cristian Gatu
`TODO: affiliation`

Erricos J. Kontoghiorghes
Birbeck College

Achim Zeileis
Universität Innsbruck

Abstract

`TODO: write abstract`

Keywords: linear regression, model selection, variable selection, best subset regression, R.

1. Introduction

An important problem in statistical modeling is that of subset selection regression or, equivalently, of finding the best regression equation (Hastie, Tibshirani, and Friedman 2001). Given a set of possible variables to be included in the regression, the problem is to select a subset which optimizes some statistical criterion. The latter originates in the estimation of the corresponding submodel (Miller 2002).

`TODO: Some comments and references to applications might be useful here.`

Consider the standard linear regression model

$$y = X\beta + \epsilon, \tag{1}$$

where $y \in \mathbb{R}^M$ is the output variable vector, $X \in \mathbb{R}^{M\times N}$ is the regressor matrix of full column rank, $\beta \in \mathbb{R}^N$ is the coefficient vector and $\epsilon \in \mathbb{R}^N$ is the noise vector. The columns of X correspond to the set of regressor variables $V = \{v_1, \ldots, v_N\}$. A submodel S of (1) comprises some of the variables in V. The best-subset selection problem is to determine

$$S^* = \operatorname*{argmin}_{S\subseteq V} f(S), \tag{2}$$

where f is some criterion function. Standard selection criteria include the AIC family (Miller 2002). For constant number of parameters these criteria are monotonic functions of the residual sum of squares (RSS), and attain their optimal value when the RSS is minimal. Thus (2) can be re-written as a two stage problem. In a first stage find

$$S_n^* = \operatorname*{argmin}_{S\subseteq V,\ |S|=n} \mathrm{RSS}(S) \quad \text{for} \quad n = 1, \ldots, N, \tag{3}$$

and in the second stage select

$$S^* = \underset{S_n^*, n=1,\ldots,N}{\operatorname{argmin}} f(S_n^*). \tag{4}$$

Solving (3) means to compute all-subset models corresponding to each submodel size. Since the S_n^* $(n = 1, \ldots, N)$ do not depend on a model size penalty, the solution of (3) can rely on the RSS only. In the second stage the best submodel S^* can be selected with respect to any standard criterion over the N submodels $S_1^*, \ldots, S_N^*$. Therefore, solving the problem (2) can be accomplished by solving the problems (3) and (4). Note that solving directly the best-subset problem (2) can allow a computational strategy to focus on finding the best submodel at lower computational cost than problem (3). On the other hand, solving (3) has the advantage that all N submodels are available and the computationally cheap problem (4) can be performed for different criterion functions (e.g., AIC and BIC).

`TODO: The subsequent paragraph should connect better to the previous paragraph. It would be good to have some general discussion about pros and cons of exact vs. approximate algorithms. And then we can discuss the implementation of the different algorithms in R.`

Algorithms for subset regression that do exact search but avoid exhaustive fitting of all models can be found on the R package **leaps** (Lumley and Miller 2009) based on Miller (2002). Exhaustive search has been considered also within the context of generalized linear models and summarized on the **bestglm** package (McLeod and Xu 2014). Alternative approximate solutions based on simulated annealing were introduced in the **subselect** package (Orestes Cerdeira, Duarte Silva, Cadima, and Minhoto 2015) based on Duarte Silva (2001). Furthermore approximate solutions via genetic algorithms for generalized linear models and beyond have been implemented in **glmulti** (Calcagno and de Mazancourt 2010) and **kofnGA** (Wolters 2015). Regularized estimation of parametric models with automatic variable selection is performed by lasso or elastic net estimation for generalized linear models (Friedman, Hastie, and Tibshirani 2010).

Here, the **lmSubsets** package for exact, best variable-subset regression is presented. It implements the algorithms presented by Gatu and Kontoghiorghes (2006) and Hofmann, Gatu, and Kontoghiorghes (2007). The package embeds functions that solve both the best-subset problem (2) and the all-subsets problem (3). A branch-and-bound device is employed to prune non-optimal sub-spaces when computing the solution. Exact and approximate strategies are deployed in order to improve the computational performance of the branch-and-bound algorithm. The numerical tool employed for estimation is the QR decomposition and its modification. Computationally intensive core code is written in C++. It is available as a package (Hofmann, Gatu, Kontoghiorghes, and Zeileis 2016) for the R system for statistical computing (R Core Team 2016) from the Comprehensive R Archive Network at `https://CRAN.R-project.org/package=lmSubsets`.

`TODO: The remainder of this manuscript is organized ...`

`TODO: Introduce BBA abbreviation somewhere.`

2. Computation strategies

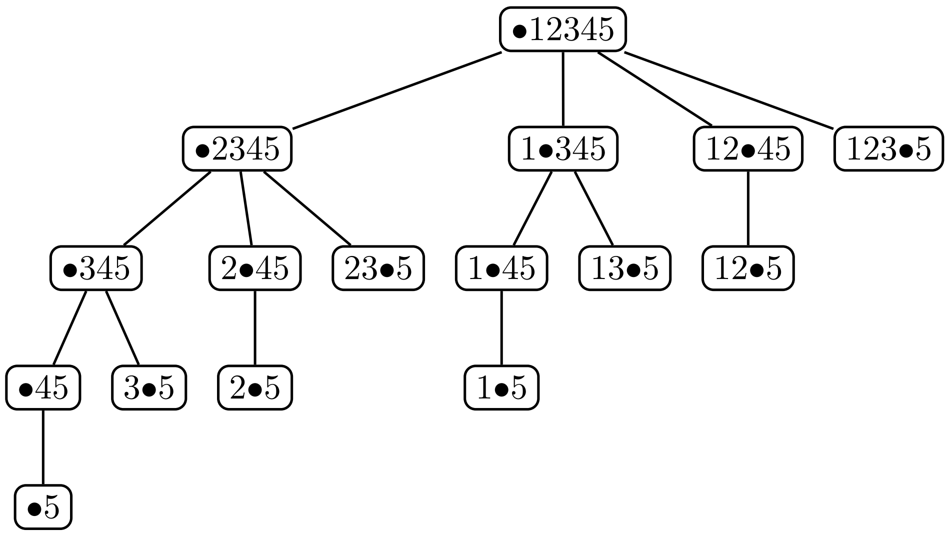

Figure 1: All-subsets regression tree for $N = 5$ variables.

Before discussing the implementation in **lmSubsets**, the underlying methods and algorithms are briefly reviewed. Emphasis is given to the general idea and strategy whereas for all theory and algorithmic details we refer to Gatu and Kontoghiorghes (2006) and Hofmann *et al.* (2007), respectively.

In the linear regression model from Equation 1 there are $2^N - 1$ possible subset models and an exhaustive computation of all of them is only feasible for small values of N. However, regression trees can be employed to traverse the search space in a systematic fashion, avoiding the need to explicitly compute every subset combination. Figure 1 illustrates a regression tree for $N = 5$ variables. A node in the regression tree is a pair (S, k), where $S = (s_1, \ldots, s_n)$ is a certain subset of n variables and k is an integer, symbolized by a $\bullet$.

Each node corresponds to a unique subset of variables, although not every possible subset gives rise to a node. Thus, the number of nodes is 2^{N-1}. The root node corresponds to the full set of variables, that is $(V, 1)$. When the algorithm visits node (S, k), it reports the RSS of the models corresponding to the leading variable subsets of size $k + 1, \ldots, n$, i.e., the subleading models $(s_1, \ldots, s_{k+1}), \ldots, (s_1, \ldots, s_n)$. It then generates and in turn visits the nodes $(S - \{s_j\}, j - 1)$ for $j = n - 1, \ldots, k + 1$. The reported RSS is stored in a subset table r along with its corresponding subset of variables. The entry r_n corresponds to the RSS of the best subset model with n variables found so far (Gatu and Kontoghiorghes 2006).

The algorithm employs a branch-and-bound strategy to reduce the number of generated nodes by cutting subtrees which do not contribute to the best solution. A cutting test is employed to determine which parts of the tree are redundant. That is, a new node $(S - \{s_j\}, j - 1)$ is generated only if $\mathrm{RSS}(S) < r_j$ $(j = k + 1, \ldots, n - 1)$. The quantity $\mathrm{RSS}(S)$ is said to be the *bound* of the subtree rooted in (S, k); that is, no subset model extracted from the subtree can have a lower RSS (Gatu and Kontoghiorghes 2006).

The search can be restricted to find only the submodels of size within a specified range. This yields in spanning only a part of the regression tree in Figure 1, and therefore it requires a reduced computational cost.

In order to encourage the cutting of large subtrees, the regression tree is generated such that large subtrees have greater bounds. The algorithm achieves this by preordering the variables. Computing the bounds of the variables implies a computational overhead. Thus, it is not advisable to preorder the variables in all the nodes. A parameter – the preordering radius p $(0 \leq p \leq N)$ – defines the extent to which variables are preordered (Gatu and Kontoghiorghes 2006; Hofmann *et al.* 2007).

The efficiency of the branch-and-bound strategy is improved by allowing the algorithm to prune non-redundant portions of the regression tree. Thus, the cutting test is relaxed by employing a tolerance parameter $\tau_n \geq 0$ $(n = 1, \ldots, N)$. A node $(S - \{s_j\}, j - 1)$ is generated only if there exists at least one i such that $(1 + \tau_i) \cdot \text{RSS}(S) < r_i$ $(i = j, \ldots, n - 1)$. The algorithm is non-exhaustive if $\tau_n > 0$ for any n, meaning that the computed solution is not guaranteed to be optimal. The algorithm cuts subtrees more aggressively the greater the value of τ_n; the relative error of the solution is bounded by the employed tolerance (Gatu and Kontoghiorghes 2006; Hofmann *et al.* 2007).

The algorithm reports the N subset models with the lowest RSS, one for each subset size. The user can then analyze the list of returned subsets to determine the "best" subset, e.g., by evaluating some criterion function. This approach is practical but not necessarily efficient. The algorithm may be optimized for a particular criterion f under the condition that the latter may be expressed as a function of the subset size n and the RSS ρ, i.e., $f(n, \rho)$, and that f is monotonic with respect to both n and ρ. It takes a single tolerance value and returns a single solution, that is the overall (i.e., over all subset sizes) best subset model according to criterion function f.

3. Implementation in R

The R package **lmSubsets** provides infrastructure for solving both the all-subsets regression (3) and the best-subset selection (2) problem in least-squares regression models. The corresponding high-level S3 generics are `lmSubsets()` and `lmSelect()`, respectively.

The interfaces of both methods are closely modeled after the `lm()` function and implement R's standard `formula` interface: they can be called with any entity that can be coerced to a `formula` object (Chambers and Hastie 1992). After the `formula` description has been processed, the raw data as well as problem-specific parameters are forwarded to the appropriate core functions. The workhorse functions `lmSubsets_fit()` and `lmSelect_fit()` implement the core computational logic without any `formula`-related overhead. An S3 object of class "`lmSubsets`" or "`lmSelect`" is returned, for which several standard extractor methods have been defined. An overview of the various functions is given in Table 1.

3.1. Specifying the regression and subset selection problem

The user first specifies a linear model and provides a dataset from which numerical values for the variables are taken. When *best*-subset regression is performed, a *penalty* per parameter must be indicated as well.

The initial model is typically given in the form of a symbolic description. A `formula` object – or any other object that can be coerced to a `formula` – specifies the dependent and independent variables, which are taken from a `data.frame` object. For example, the call

Problem	Function	Description
All-subsets regression	`lmSubsets.default()`	Standard formula interface
	`lmSubsets_fit()`	Workhorse function
Best-subsets regression	`lmSelect.default()`	Standard formula interface
	`lmSelect.lmSubsets()`	Coercion method
	`lmSelect_fit()`	Workhorse function
	`lmSubsets_select()`	Explicit conversion function

Table 1: Generator methods and functions.

```
lmSubsets(mortality ~ precipitation + temperature1 + temperature7 +
  age + household + education + housing + population + noncauc +
  whitecollar + income + hydrocarbon + nox + so2 + humidity,
  data = AirPollution)
```

specifies a response variable (`mortality`) and fifteen predictor variables with all variables being taken from the `AirPollution` dataset (Miller 2002). In this case, the call can be abbreviated to

```
lmSubsets(mortality ~ ., data = AirPollution)
```

where the dot (`.`), as usual, stands for "all variables not mentioned in the left-hand side of the formula". By default, an intercept term is included in the model; that is, the call in the previous example is equivalent to

```
lmSubsets(mortality ~ . + 1, data = AirPollution)
```

To discard the intercept, the call may be rewritten as follows:

```
lmSubsets(mortality ~ . - 1, data = AirPollution)
```

During execution, candidate models can be rejected based on the presence or absence of certain independent variables. The parameter `include` is employed to accept only submodels that comprise one or more specified variables. In the following example, only submodels containing the variable `noncauc` will be retained:

```
lmSubsets(mortality ~ ., include = "noncauc", data = AirPollution)
```

Conversely, the `exclude` parameter can be employed to discard a specific set of variables, as in the following example:

```
lmSubsets(mortality ~ ., exclude = "whitecollar", data = AirPollution)
```

The same effect can be achieved by rewriting the formula as follows:

```
lmSubsets(mortality ~ . - whitecollar, data = AirPollution)
```

The `include` and `exclude` parameters may be used in combination, and both may specify more than one variable (e.g., `include = c("noncauc", "whitecollar")`).

The criterion used for best-subsets selection is evaluated following the expression

$$-2 \cdot \texttt{logLik} + \texttt{penalty} \cdot \texttt{npar},$$

where `penalty` is the *penalty per model-parameter*, `logLik` the log-likelihood of the fitted model, and `npar` the number of model-parameters (including the error variance). The `penalty` value indicates how strongly model parameters are penalized with higher values favoring more parsimonious solutions. When `penalty` $= 2$, the criterion corresponds to Akaike's information criterion (AIC, Akaike 1974); when `penalty` $= \log(\texttt{nobs})$, to Schwarz's Bayesian information criterion (BIC, Schwarz 1978), `nobs` being the number of observations. For example, either one of

```
lmSelect(mortality ~ ., data = AirPollution, penalty = 2)
lmSelect(mortality ~ ., data = AirPollution, penalty = "AIC")
```

will select the best submodels according to the usual AIC. By default, the `lmSelect()` function employs the BIC.

3.2. Core functions

The high-level interfaces first process the formula specification, set up the model data, and pass these to dedicated core functions – along with further optional problem-specific arguments. The core functions act as wrappers for the C++ library which implements the problem logic. The full list of arguments (see also Table 2) is given by:

```
lmSubsets_fit(x, y, weights = NULL, offset = NULL,
  include = NULL, exclude = NULL, nmin = NULL,
  nmax = NULL, tolerance = 0, pradius = NULL, nbest = 1, ...,
  .algo = "phbba")

lmSelect_fit(x, y, weights = NULL, offset = NULL,
  include = NULL, exclude = NULL, penalty = "BIC",
  tolerance = 0, pradius = NULL, nbest = 1, ...,
  .algo = "phbba")
```

The model data is given by means of a numeric matrix `x` and vector `y`. The `weights` and `offset` parameters correspond to the homonomic parameters of the `lm` function.

The `include` and `exclude` parameters allow the user to specify variables that are to be included in or excluded from all candidate models. They can be either logical vectors – with each entry corresponding to one variable – or are automatically expanded to such if given in the form of an integer vector (i.e., set of variable indices) or character vector (i.e., set of variable names).

For a large numbers of variables (see Section 5), execution times may become intractable. In order to speed up the execution, either the search space can be reduced, or one may settle for a non-exhaustive solution. In the first appraoch, the user can specify values for the `nmin` and

Parameter	Description	Representation
`x`	Data matrix	`numeric[nobs,nvar]`
`y`	Response variable	`numeric[nobs]`
`weights`	Model weights	`numeric[nobs]`
`offset`	Model offset	`numeric[nvar]`
`include`	Regressors to force in	`logical[nvar]`
`exclude`	Regressors to force out	`logical[nvar]`
`nmin`	Min. number of regressors (`lmSubsets()` only)	`integer[1]`
`nmax`	Max. number of regressors (`lmSubsets()` only)	`integer[1]`
`tolerance`	BBA tolerance parameters	`numeric[nvar]` (`lmSubsets()`)
		`numeric[1]` (`lmSelect()`)
`pradius`	Preordering radius	`integer[1]`
`nbest`	Number of best subsets	`integer[1]`
`.algo`	Algorithm to execute	`character[1]`

Table 2: Core parameters.

`nmax` parameters, in which case submodels with less than `nmin` or more than `nmax` variables are discarded, effectively removing entire branches of the regression tree from the search space.

In the second approach, expectations with respect to the solution quality are lowered, i.e., non-optimal solutions are *tolerated*. This is indicated by passing a numeric value – typically between 0 and 1 – to the `tolerance` parameter, which will be used by the BBA cutting test to prune the search tree. The solution produced by the algorithm satisfies the following relationship:

$$f(S) \leq (1 + \texttt{tolerance}) \cdot f(S^*),$$

where S is the returned solution, S^* the optimal (theoretical) solution, and f the value of a submodel (i.e., deviance, AIC). The `lmSubsets_fit()` function accepts a vector of tolerances, with one entry for each subset size.

The `nbest` parameter controls how many submodels (per subset size) are retained. In the case of `lmSubsets_fit()`, a two-dimensional result set is constructed with `nbest` submodels for each subset size, while in the case of `lmSelect_fit()`, a one-dimensional sequence of `nbest` submodels is handed back to the user.

The `pradius` parameter serves to specify the desired preordering radius. The algorithm employs a default value of $\lfloor \texttt{nvar}/3 \rfloor$. The need to set this parameter directly should rarely arise; please refer to Hofmann *et al.* (2007) for further information. The `.algo` parameter serves to specify the computational algorithm to be employed. This parameter is used for testing purposes only and should never be set directly.

3.3. Extracting submodels

The user is handed back a result object that encapsulates the solution to an all-subsets (class "`lmSubsets`") or best-subsets (class "`lmSelect`") selection problem. An object of class "`lmSubsets`" represents a two-dimensional `nbest` $\times$ `nvar` set of submodels; an object of class "`lmSelect`", a linear sequence of `nbest` submodels. Problem-specific information is stored alongside the selected submodels. Table 3 summarizes the components of the result object.

Component	Description	Representation
`nobs`	Number of observations	`integer[1]`
`nvar`	Number of regressors	`integer[1]`
`weights`	Weights used	`numeric[nobs]`
`offset`	Offset used	`numeric[nobs]`
`intercept`	Intercept flag	`logical[1]`
`include`	Regressors forced in	`logical[nvar]`
`exclude`	Regressors forced out	`logical[nvar]`
`penalty`	Penalty used ("`lmSelect`" only)	`numeric[nvar]`
`tolerance`	Tolerances used	`numeric[nvar]`
`nbest`	Number of best subsets	`integer[1]`
`df`	Degrees of freedom	`integer[nbest,nvar]` (for "`lmSubsets`")
		`integer[nbest]` (for "`lmSelect`")
`rss`	Residual sum of squares	`numeric[nbest,nvar]` (for "`lmSubsets`")
		`numeric[nbest]` (for "`lmSelect`")
`which`	Selected regressors	`logical[nvar,nbest,nvar]` (for "`lmSubsets`")
		`logical[nvar,nbest]` (for "`lmSelect`")

Table 3: Components of "`lmSubsets`" and "`lmSelect`" objects.

A wide range of standard methods to visualize, summarize, and extract information is provided (see Table 4). The `print()`, `plot()`, and `summary()` methods present a compact overview – either textual or graphical – of information gathered on the selected submodels, which the user can find useful to identify "good" models. The remaining extractor functions behave in the usual way, and can be used to extract variable names, coefficients, covariances matrices, fitted values, etc.

In order to identify the submodel for which information is to be extracted, "`lmSubsets`" methods provide two parameters, namely `size` and `best`, which specify the number of regressors in and the ranking of the desired model, respectively. The `size` argument is mandatory, while by default `best = 1`. For "`lmSelect`" methods, the `size` parameter has no meaning and is not defined. Furthermore, methods that return scalar values — i.e., `deviance()`, `logLik()`, `AIC()`, `BIC()` – can process more than one submodel at a time, by passing a numeric vector as an argument to either `size` (e.g., `size = 5:10`) or `best` (e.g., `best = 1:3`).

4. Case study: Variable selection for weather forecasting

Over the last decades the field of weather forecasting made steady and substantial improvements especially through improvements in numerical weather prediction (NWP) models (Bauer, Thorpe, and Brunet 2015). Starting from Glahn and Lowry (1972) the outputs from these physically-based large-scale (typically global) NWP models is statistically post-processed to correct small-scale biases and obtain predictions for specific locations. Below we use such model output statistics (MOS) to predict temperature at a specific station (Innsbruck Airport, Austria) based on a wide range of NWP quantities from the nearest NWP grid point. Variable subset selection is relevant here because it is not obvious which quantities beyond the temperature NWP forecasts should enter the MOS regression.

More specifically, we model 00UTC temperature observations (in degree Celsius) based on

Method	Description
`print()`	Print object
`plot()`	Plot RSS (and information criteria)
`image()`	Heatmap of selected regressors (“`lmSubsets`” only)
`summary()`	Summary statistics
`variable.names()`	Extract variables names
`formula()`	Extract formula object
`model.frame()`	Extract (full) model frame
`model.matrix()`	Extract model matrix
`model.response()`	Extract model response
`refit()`	Fit subset “`lm`” model
`coef()`	Extract regression coefficients
`vcov()`	Extract covariance matrix
`fitted()`	Extract fitted values
`residuals()`	Extract residual values
`deviance()`	Extract deviance (RSS)
`logLik()`	Extract log-likelihood
`AIC()`	Extract AIC values
`BIC()`	Extract BIC values

Table 4: S3 methods for “`lmSubsets`” and “`lmSelect`” objects.

the corresponding 24-hour GEFS reforecast Hamill *et al.* (2013) ensemble means for SYNOP station Innsbruck Airport (11120; 47.260, 11.357) from 2011-01-01 to 2015-12-31. The data frame `IbkTemperature` contains 1824 daily observations/forecasts for the observed response, 36 NWP outputs, and five deterministic time trend/season patterns that are available as potential regressors. The NWP variables include several temperature quantities (in degree Kelvin, e.g., 2-meter, minimum, maximum, soil) as well as several quantities capturing precipitation, wind, and fluxes among others. See `?IbkTemperature` for more details. The data from the NOAA (United States National Oceanic and Atmospheric Administration) are obtained from `http://www.esrl.noaa.gov/psd/forecasts/reforecast2/` (reforecasts) and `http://www.ogimet.com/synops.phtml.en` (observations), respectively.

To start the analysis the data from the **lmSubsets** package can be loaded and for simplicity a couple of days with some missing values are omitted.

```
R> data("IbkTemperature", package = "lmSubsets")
R> IbkTemperature <- na.omit(IbkTemperature)
```

First, a simple climatological model for the temperature (`temp`) with a linear trend (`time`) and a harmonic seasonal pattern (`sin`/`cos` for the annual and `sin2`/`cos2` for the bi-annual frequencies).

```
R> m0 <- lm(temp ~ time + sin + cos + sin2 + cos2, data = IbkTemperature)
```

This model does not make use of any NWP outputs and is used as a basic reference model against which the subsequent MOS models can be compared. Second, the model is updated to a simple MOS by including the most obvious direct model output – 2-meter temperature (`t2m`) – in addition to the season/trend regressors.

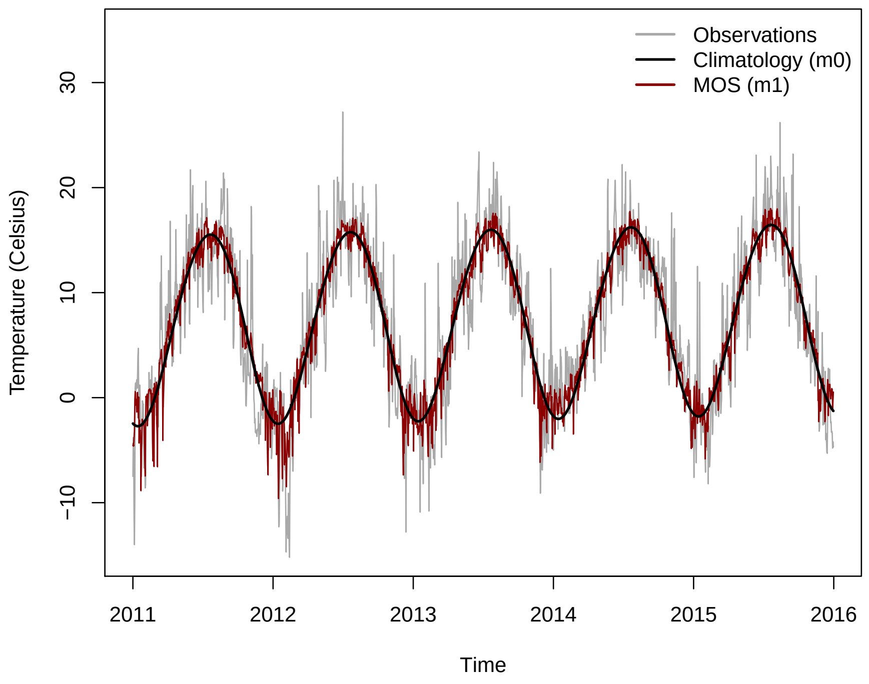

Figure 2: Observed temperature at Innsbruck Airport (gray) and fitted values from the climatological model (`m0`, black) and the simple MOS (`m1`, red).

```
R> m1 <- update(m0, . ~ . + t2m)
```

A graphical comparison of the raw data and the fitted values from both models is provided in Figure 2. The corresponding estimated coefficients (and standard errors) are shown in Table 5 (produced with **memisc**, Elff 2016). This shows that, not surprisingly, inclusion of the 2-meter temperature leads to substantial (and highly significant) improvements. The season/trend coefficients are dampened but remain significant which means that not all seasonal temperature pattern at Innsbruck Airport are resolved in the coarse NWP grid.

Subsequently, we now try to improve the MOS by not only including the direct model output for 2-meter temperature but also further NWP model outputs. As a starting point we force the regressors from `m1` into the model and use all-subsets regression to select the relevant regressors from the remaining 35 NWP variables.

```
R> ms2 <- lmSubsets(temp ~ ., data = IbkTemperature,
+    include = c("t2m", "time", "sin", "cos", "sin2", "cos2"))
R> m2 <- refit(lmSelect(ms2, penalty = "BIC"))
```

After obtaining the all-subsets regression `ms2` with `lmSubsets()` the best BIC solution is extracted with `lmSelect()` and turned into a "`lm`" model with `refit()`. The more costly all-subsets regression is solved here to gain more insights into the selected variables not only for the best-BIC solution but also other models.

To assess whether our initial MOS strategy in `m1` (and forced into `m2`) is really the most suitable we also carry out another all-subsets regression without restricting the search space.

	m0	m1	m2	m3
(Intercept)	-458.732^{***}	-345.252^{**}	-666.584^{***}	-661.700^{***}
	(119.936)	(109.212)	(95.349)	(95.225)
time	0.231^{***}	0.132^{*}	0.149^{**}	0.147^{**}
	(0.060)	(0.054)	(0.047)	(0.047)
sin	-2.433^{***}	-1.234^{***}	0.522^{***}	0.811^{***}
	(0.121)	(0.126)	(0.147)	(0.120)
cos	-8.716^{***}	-6.329^{***}	-0.812^{**}	
	(0.120)	(0.164)	(0.273)	
sin2	0.051	0.240^{*}	-0.794^{***}	-0.870^{***}
	(0.120)	(0.110)	(0.119)	(0.118)
cos2	-0.380^{**}	-0.332^{**}	-1.067^{***}	-1.128^{***}
	(0.120)	(0.109)	(0.101)	(0.097)
t2m		0.318^{***}	0.055	
		(0.016)	(0.029)	
sshnf			0.016^{***}	0.018^{***}
			(0.004)	(0.004)
vsmc			20.200^{***}	20.181^{***}
			(3.115)	(3.106)
tmax2m			0.145^{***}	0.181^{***}
			(0.037)	(0.023)
st			1.077^{***}	1.142^{***}
			(0.051)	(0.043)
wr			0.450^{***}	0.505^{***}
			(0.109)	(0.103)
t2pvu			0.064^{***}	0.149^{***}
			(0.011)	(0.028)
mslp				-0.000^{***}
				(0.000)
p2pvu				-0.000^{**}
				(0.000)
AIC	9838.5	9493.6	8954.9	8948.2
BIC	9877.1	9537.7	9032.0	9025.3
Deviance	23605.3	19506.5	14411.1	14357.9
sigma	3.6	3.3	2.8	2.8
R-squared	0.8	0.8	0.9	0.9

Table 5: Estimated regression coefficients (and standard errors) along with further summary statistics for the climatological model `m0` and the three MOS models (`m1`–`m3`).

```
R> ms3 <- lmSubsets(temp ~ ., data = IbkTemperature)
R> m3 <- refit(lmSelect(ms3, penalty = "BIC"))
```

Obtaining `ms3` is computationally somewhat more costly than `ms2` but still very fast taking only a couple of seconds on standard PCs.

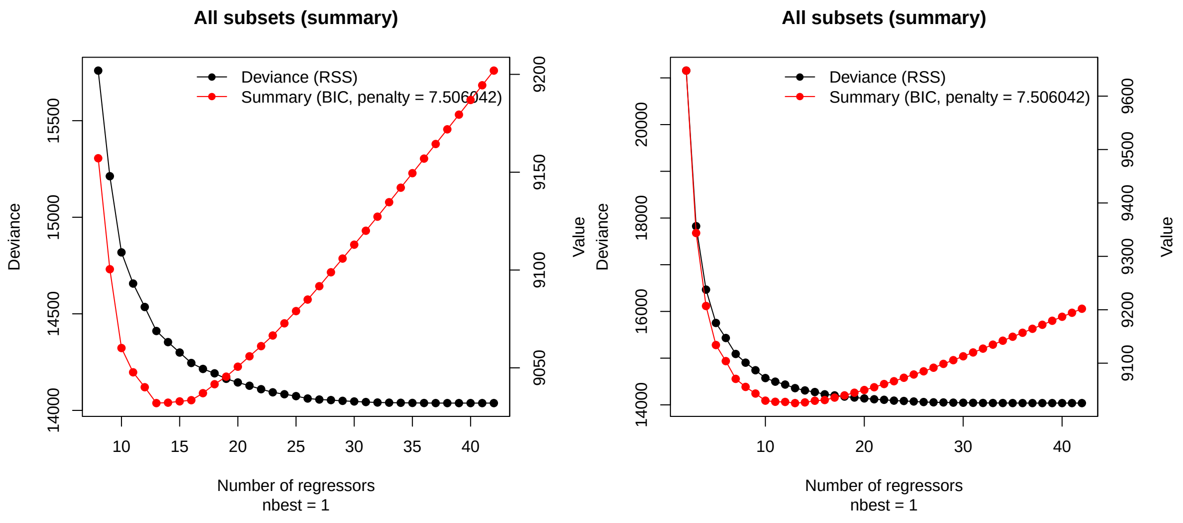

Figure 3: Best RSS and associated BIC for all subset sizes considered in `ms2` (left) and `ms3` (right).

To assess the subset selections, Figure 3 shows a graphical summary of best RSS and associated BIC for all subset sizes considered in `ms2` and `ms3`, respectively. The plots can be easily produced by `plot(summary(...))` and start with subset size 8 for `ms2` because seven variables are forced into the model while in `ms3` only the intercept is always included. For both models the RSS and BIC curves look rather similar and the best-BIC models both have 13 regressors.

The corresponding selected variables can be seen in Table 5 produced by `mtable(m1, m2, m3, m4)`. This shows that both `m2` and `m3` are rather similar with respect to the selected variables and corresponding coefficients. However, interestingly the direct model output `t2m` is not selected for `m3` and instead the soil temperature (`st`) as well as the maximal 2-meter temperature (`tmax2m`) and temperature on the so-called 2 PVU surface (`t2pvu`) are used which are selected in addition to `t2m` in `m2`. Additionally, various other meteorological quantities are selected that improve the forecasting model further, e.g., soil moisture `vsmc`) among others.

To gain further insight into the best-subset selection for various sizes the `image()` method is useful. Figure 4 and Figure 5 show the results for `ms2` and `ms3`, respectively, for subset sizes up to 20 variables. The dark cells in the heatmap show the variables that are selected while the best-BIC solution is highlighted with a red rectangle and by underlining the variable names in the x-axis labels. While there are many similarities between the patterns shown, it is striking that `t2m` is not considered for any of the subsets in `ms3` while we forced it into `ms2`. The decision to always include the deterministic season/trend regressors, however, is confirmed as some of these regressors are already selected for low subset sizes and all of them are selected from size 14 onwards.

Nevertheless, the differences between `m2` and `m3` in terms of model fit are fairly small compared to the reference models `m0` and `m1`. Comparing the BIC and the root mean squared error (RMSE) gives

```
R> BIC(m0, m1, m2, m3)
```

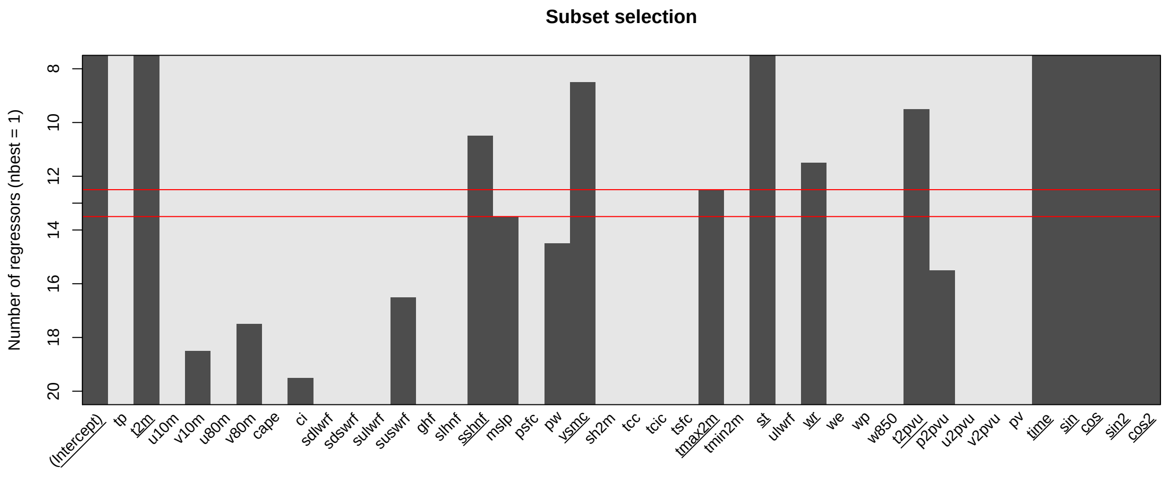

Figure 4: Subset selection for `ms2`.

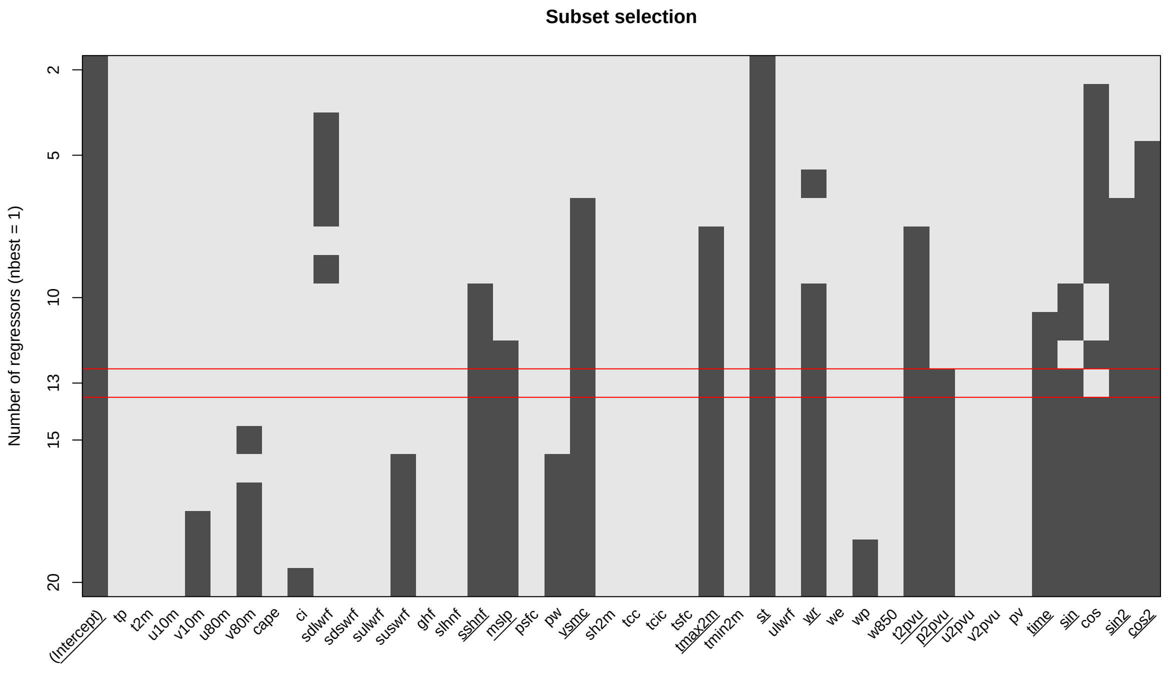

Figure 5: Subset selection for `ms3`.

```
   df      BIC
m0  7 9877.073
m1  8 9537.650
m2 14 9031.992
m3 14 9025.267
```

```
R> sqrt(sapply(list(m0, m1, m2, m3), deviance)/nrow(IbkTemperature))
```

```
[1] 3.602371 3.274711 2.814703 2.809505
```

showing that `m2` and `m3` improve the model substantially over `m0` and `m1`. By construction the BIC of `m3` is lower than that of `m2` but not by much.

In summary, this shows that best-subset selection can easily identify relevant variables beyond the direct model output for MOS regressions, yielding substantial improvements in forecasting quality. In a full meteorological application this should, of course, be further tested using cross-validation or other out-of-sample assessments. But as this is beyond the scope of this paper we confine ourselves to the in-sample assessment presented here.

To conclude, it is also worth pointing out that recently there has been increasing interest in MOS models beyond least-squares linear regression. Incorporating heteroscedasticity is important ensemble MOS models as well as censoring or truncation for quantities like precipitation or wind (see the **crch** package of Messner, Mayr, and Zeileis 2016a for some examples). All-subsets regression for those models would be much more burdensome and is not available in **lmSubsets**. An alternative solution in those situations is for example boosting as proposed by Messner, Mayr, and Zeileis (2016b).

5. Benchmark

`TODO: by X. Below are some rough ideas...Before going into detail about this one should check what other authors in this field have used for setting up their regressions. Following their templates might make the design easier and more convincing for the reviewers.`

Data-generating process:

- Allow increasing M and/or N.
- A certain fixed fraction of variables could be relevant (say 20%).
- All regressors could have the same distribution, e.g., standard normal or uniform on $[-1, 1]$.
- The relevant variables could also all have the same coefficient or follow a linearly decreasing pattern for example. All irrelevant variables have coefficient zero.
- Then one could consider two values of M (say, 200 and 1000) and a sequence of Ns (say, 20, 40, ...).

Competitors:

- lmSubsets and lmSelect, either exactly or with certain relaxations.
- Other exact solutions (at least: leaps).
- Approximate solutions (at least: subselect and/or glmulti).
- Penalized solutions (at least: glmnet).

Outcome measurements:

- Computation time of the R functions (especially for exact solutions).

- Proportion of correctly selected regressors.
- RSS, AIC, BIC.

Acknowledgments

This work was in part supported by the *Förderverein des wirtschaftswissenschaftlichen Zentrums der Universität Basel*, through research project B-123. The authors are grateful to Jakob Messner for sharing the GEFS forecast data in `IbkTemperature`.

References

Akaike H (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control*, **19**(6), 716–723. `doi:10.1109/tac.1974.1100705`.

Bauer P, Thorpe A, Brunet G (2015). "The Quiet Revolution of Numerical Weather Prediction." *Nature*, **525**(7567), 47–55. `doi:10.1038/nature14956`.

Calcagno V, de Mazancourt C (2010). "**glmulti**: An R Package for Easy Automated Model Selection with (Generalized) Linear Models." *Journal of Statistical Software*, **34**(12), 1–29. `doi:10.18637/jss.v034.i12`.

Chambers JM, Hastie TJ (eds.) (1992). *Statistical Models in S.* Chapman & Hall, London.

Duarte Silva AP (2001). "Efficient Variable Screening for Multivariate Analysis." *Journal of Multivariate Analysis*, **76**, 35–62. `doi:10.1006/jmva.2000.1920`.

Elff M (2016). ***memisc**: Tools for Management of Survey Data and the Presentation of Analysis Results.* R package version 0.99.7-1, URL `https://CRAN.R-project.org/package=memisc`.

Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. `doi:10.18637/jss.v033.i01`.

Gatu C, Kontoghiorghes EJ (2006). "Branch-and-Bound Algorithms for Computing the Best Subset Regression Models." *Journal of Computational and Graphical Statistics*, **15**, 139–156. `doi:10.1198/106186006x100290`.

Glahn HR, Lowry DA (1972). "The Use of Model Output Statistics (MOS) in Objective Weather Forecasting." *Journal of Applied Meteorology*, **11**(8), 1203–1211. `doi:10.1175/1520-0450(1972)011<1203:TUOMOS>2.0.CO;2`.

Hamill TM, Bates GT, Whitaker JS, Murray DR, Fiorino M, Galarneau Jr TJ, Zhu Y, Lapenta W (2013). "NOAA's Second-Generation Global Medium-Range Ensemble Reforecast Data Set." *Bulletin of the American Meteorological Society*, **94**(10), 1553–1565. `doi:10.1175/BAMS-D-12-00014.1`.

Hastie TJ, Tibshirani RJ, Friedman J (2001). *The Elements of Statistical Learning – Data Mining, Inference, and Prediction.* Springer-Verlag, New York.

Hofmann M, Gatu C, Kontoghiorghes EJ (2007). "Efficient Algorithms for Computing the Best Subset Regression Models for Large-Scale Problems." *Computational Statistics & Data Analysis*, **52**, 16–29. `doi:10.1016/j.csda.2007.03.017`.

Hofmann M, Gatu C, Kontoghiorghes EJ, Zeileis A (2016). ***lmSubsets**: Variable Subset Selection in Linear Regressions.* R package version 0.1-0, URL `https://CRAN.R-project.org/package=lmSubsets`.

Lumley T, Miller A (2009). ***leaps**: Regression Subset Selection.* R package version 2.9, URL `https://CRAN.R-project.org/package=leaps`.

McLeod AI, Xu C (2014). ***bestglm**: Best Subset GLM.* R package version 0.34, URL `https://CRAN.R-project.org/package=bestglm`.

Messner JW, Mayr GJ, Zeileis A (2016a). "Heteroscedastic Censored and Truncated Regression with **crch**." *The R Journal.* Forthcoming, URL `https://journal.R-project.org/archive/accepted/messner-mayr-zeileis.pdf`.

Messner JW, Mayr GJ, Zeileis A (2016b). "Non-Homogeneous Boosting for Predictor Selection in Ensemble Post-Processing." *Working Paper 2016-04*, Working Papers in Economics and Statistics, Research Platform Empirical and Experimental Economics, Universität Innsbruck. URL `http://EconPapers.RePEc.org/RePEc:inn:wpaper:2016-04`.

Miller AJ (2002). *Subset Selection in Regression*, volume 95. 2nd edition. Chapman and Hall. `doi:10.1201/9781420035933`.

Orestes Cerdeira J, Duarte Silva AP, Cadima J, Minhoto M (2015). ***subselect**: Selecting Variable Subsets.* R package version 0.12-5, URL `https://CRAN.R-project.org/package=subselect`.

R Core Team (2016). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Schwarz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**(2), 461–464. `doi:10.1214/aos/1176344136`.

Wolters MA (2015). "A Genetic Algorithm for Selection of Fixed-Size Subsets with Application to Design Problems." *Journal of Statistical Software, Code Snippets*, **68**(1), 1–18. `doi:10.18637/jss.v068.c01`.

Affiliation:

Marc Hofmann
`TODO: address`