Journal of Statistical Software

February 2018, Volume 83, Issue 9. *doi: 10.18637/jss.v083.i09*

mplot: An R Package for Graphical Model Stability and Variable Selection Procedures

Garth Tarr
University of Sydney

Samuel Müller
University of Sydney

Alan H. Welsh
Australian National University

Abstract

The **mplot** package provides an easy to use implementation of model stability and variable inclusion plots (Müller and Welsh 2010; Murray, Heritier, and Müller 2013) as well as the adaptive fence (Jiang, Rao, Gu, and Nguyen 2008; Jiang, Nguyen, and Rao 2009) for linear and generalized linear models. We provide a number of innovations on the standard procedures and address many practical implementation issues including the addition of redundant variables, interactive visualizations and the approximation of logistic models with linear models. An option is provided that combines our bootstrap approach with **glmnet** for higher dimensional models. The plots and graphical user interface leverage state of the art web technologies to facilitate interaction with the results. The speed of implementation comes from the **leaps** package and cross-platform multicore support.

Keywords: model selection, variable selection, linear models, mixed models, generalized linear models, fence, R.

1. Graphical tools for model selection

In this article we introduce the **mplot** package (Tarr, Müller, and Welsh 2018) for R (R Core Team 2017), which provides a suite of interactive visualizations and model summary statistics for researchers to use to better inform the variable selection process and is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=mplot`. The methods we provide rely heavily on various bootstrap techniques to give an indication of the stability of selecting a given model or variable and even though not done here, could be implemented with resampling methods other than the bootstrap, for example cross-validation. The 'm' in **mplot** stands for model selection/building and we anticipate that in future more graphs and methods will be added to the package to further aid better and more stable building of regression models. The intention is to encourage researchers to engage more closely with the model selection process, allowing them to pair their experience and domain specific knowledge with comprehensive summaries of the relative importance of various statistical models.

Two major challenges in model building are the vast number of models to choose from and the myriad of ways to do so. Standard approaches include stepwise variable selection techniques and more recently the lasso (least absolute shrinkage and selection operator). A common issue with these and other methods is their instability, that is, the tendency for small changes in the data to lead to the selection of different models.

An early and significant contribution to the use of bootstrap model selection is Shao (1996) who showed that carefully selecting m in an m-out-of-n bootstrap drives the theoretical properties of the model selector. Müller and Welsh (2005, 2009) modified and generalized Shao's m-out-of-n bootstrap model selection method to robust settings, first in linear regression and then in generalized linear models. The bootstrap is also used in regression models that are not yet covered by the **mplot** package, such as mixed models (e.g., Shang and Cavanaugh 2008) or partially linear models (e.g., Müller and Vial 2009) as well as for the selection of tuning parameters in regularization methods (e.g., Park, Sakaori, and Konishi 2014).

Assume that we have n independent observations $\mathbf{y} = (y_1, \ldots, y_n)^\top$ and an $n \times p$ full rank design matrix $\mathbf{X}$ whose columns are indexed by $1, \ldots, p$. Let α denote any subset of p_α distinct elements from $\{1, \ldots, p\}$. Let $\mathbf{X}_\alpha$ be the corresponding $n \times p_\alpha$ design matrix and $\mathbf{x}_{\alpha i}^\top$ denote the ith row of $\mathbf{X}_\alpha$.

The **mplot** package focuses specifically on linear and generalized linear models (GLM). In the context of GLMs, a model α for the relationship between the response $\mathbf{y}$ and the design matrix $\mathbf{X}_\alpha$ is specified by

$$\mathsf{E}(\mathbf{y}) = h(\mathbf{X}_\alpha^\top \boldsymbol{\beta}_\alpha), \text{ and } \mathsf{VAR}(\mathbf{y}) = \sigma^2 v(h(\mathbf{X}_\alpha^\top \boldsymbol{\beta}_\alpha)), \tag{1}$$

where $\boldsymbol{\beta}_\alpha$ is an unknown p_α-vector of regression parameters and σ is an unknown scale parameter. Here $\mathsf{E}(\cdot)$ and $\mathsf{VAR}(\cdot)$ denote the expected value and variance of a random variable, h is the inverse of the usual link function and both h and v are assumed known. When h is the identity and $v(\cdot) = 1$, we recover the standard linear model.

The purpose of model selection is to choose one or more models α from a set of candidate models, which may be the set of all models $\mathcal{A}$ or a reduced model set (obtained, for example, using any initial screening method). Many model selection procedures assess model fit using the generalized information criterion (GIC),

$$\text{GIC}(\alpha, \lambda) = \hat{Q}(\alpha) + \lambda p_\alpha. \tag{2}$$

The $\hat{Q}(\alpha)$ component is a measure of "description loss" or "lack of fit", a function that describes how well a model fits the data, for example, the residual sum of squares or -2 × log-likelihood. The number of independent regression model parameters, p_α, is a measure of "model complexity". The penalty multiplier, λ, determines the properties of the model selection criterion (Müller, Scealy, and Welsh 2013; Müller and Welsh 2010). Special cases, when $\hat{Q}(\alpha) = -2 \times \text{log-likelihood}(\alpha)$, include the AIC (Akaike infomation criterion) with $\lambda = 2$, BIC (Bayesian information criterion) with $\lambda = \log(n)$ and more generally the generalized information criterion (GIC) with $\lambda \in \mathbb{R}$ (Konishi and Kitagawa 1996).

The **mplot** package currently implements "variable inclusion plots", "model stability plots" and a model selection procedure inspired by the adaptive fence of Jiang *et al.* (2008). Variable inclusion plots were introduced independently by Müller and Welsh (2010) and Meinshausen and Bühlmann (2010). The idea is that the best model is selected over a range of values of the penalty multiplier λ and the results are visualized on a plot which shows how often each variable is included in the best model. These types of plots have previously been referred to as stability paths, model selection curves and most recently variable inclusion plots (VIPs) in Murray *et al.* (2013). An alternative to penalizing for the number of variables in a model is to assess the fit of models within each model size. This is the approach taken in our model stability plots where searches are performed over a number of bootstrap replications and the best models for each size are tallied. The rationale is that if there exists a "correct" model of a particular model size it will be selected overwhelmingly more often than other models of the same size. Finally, the adaptive fence was introduced by Jiang *et al.* (2008) to select mixed models. This is the first time code has been made available to implement the adaptive fence and the first time the adaptive fence has been applied to linear and generalized linear models.

This article introduces three data examples that each highlight different aspects of the graphical methods made available by package **mplot**. Sections 2–5 are based on a motivating example where the true data generating model is known. We use this example to highlight one of the classical failings of stepwise procedures before introducing variable inclusion plots and model stability plots through the `vis()` function in Section 3. Our implementation of the adaptive fence with the `af()` function is presented in Section 4.

For all methods, we provide publication quality classical plot methods using **ggplot2** graphics (Wickham 2016) as well as interactive plots using the **googleVis** package (Gesmann and de Castillo 2011). In Section 5, we show how to add further utility to these plot methods by packaging the results in a **shiny** web interface which facilitates a high degree of interactivity (Chang, Cheng, Allaire, Xie, and McPherson 2017).

In Section 6 we show computing times in a simulation study, varying the number of variables from 5 to 50; we further illustrate the advantage of using multiple core technology. We then show with two applied examples the practical merit of our graphical tools in Section 7.

To conclude, we highlight in Section 8 the key contributions of the three data examples and make some final brief remarks.

2. Illustrative example

We will present three examples to help illustrate the methods provided by the **mplot** package. Two real data sets are presented as case studies in Section 7. The first of these is a subset of the diabetes data set used in Efron, Hastie, Johnstone, and Tibshirani (2004) which has 10 explanatory variables and a continuous dependent variable, a measure of disease progression, suitable for use in a linear regression model. The second is a binomial regression example from Hosmer and Lemeshow (1989) concerning low birth weight.

The artificially generated data set was originally designed to emphasize statistical deficiencies in stepwise procedures, but here it will be used to highlight the utility of the various procedures and plots provided by package **mplot**. A scatterplot matrix of the data and the estimated pairwise correlations is given in Figure 1. All variables, while related, originate from a Gaussian distribution. The data set and details of how it was generated are provided with the **mplot** package.

```
R> install.packages("mplot")
R> data("artificialeg", package = "mplot")
R> help("artificialeg", package = "mplot")
```

Fitting the full model yields no individually significant variables.

```
R> library("mplot")
R> full.model <- lm(y ~ ., data = artificialeg)
R> round(coef(summary(full.model)), 2)

            Estimate Std. Error t value Pr(>|t|)
(Intercept)    -0.10       0.33   -0.31     0.76
x1              0.64       0.69    0.92     0.36
x2              0.26       0.62    0.42     0.68
x3             -0.51       1.24   -0.41     0.68
x4             -0.30       0.25   -1.18     0.24
x5              0.36       0.60    0.59     0.56
x6             -0.54       0.96   -0.56     0.58
x7             -0.43       0.63   -0.68     0.50
x8              0.15       0.62    0.24     0.81
x9              0.40       0.64    0.63     0.53
```

Performing default stepwise variable selection yields a model with all explanatory variables except x_8. As an aside, the dramatic changes in the p values indicate that there is substantial interdependence between the explanatory variables even though none of the pairwise correlations in Figure 1 are particularly extreme.

```
R> step.model <- step(full.model, trace = 0)
R> round(coef(summary(step.model)), 2)

            Estimate Std. Error t value Pr(>|t|)
(Intercept)    -0.11       0.32   -0.36     0.72
x1              0.80       0.19    4.13     0.00
x2              0.40       0.18    2.26     0.03
x3             -0.81       0.19   -4.22     0.00
x4             -0.35       0.12   -2.94     0.01
x5              0.49       0.19    2.55     0.01
x6             -0.77       0.15   -5.19     0.00
x7             -0.58       0.15   -3.94     0.00
x9              0.55       0.19    2.90     0.01
```

The true data generating process is, $y = 0.6\,x_8 + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 2^2)$. The bivariate regression of y on x_8 is the more desirable model, not just because it is the true model representing the data generating process, but it is also more parsimonious with essentially the same residual variance as the larger model chosen by the stepwise procedure. This example illustrates a key statistical failing of stepwise model selection procedures, in that they only

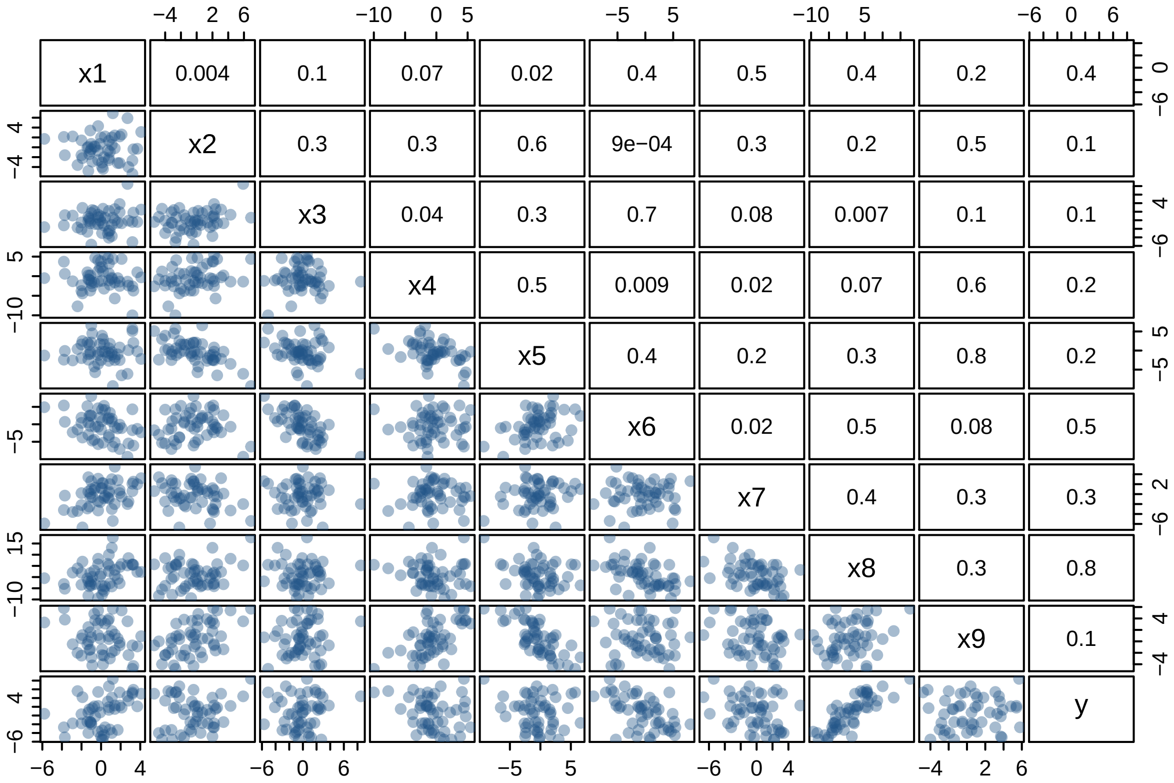


Figure 1: Scatterplot matrix of the artificially generated data set with estimated correlations in the upper right triangle. The true data generating process for the dependent variable is $y = 0.6\, x_8 + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, 2^2)$.

explore a subset of the model space so are inherently susceptible to local minima in the information criterion (Harrell 2001).

Perhaps the real problem of stepwise methods is that they allow researchers to transfer all responsibility for model selection to a computer and not put any real thought into the model selection process. This is an issue that is also shared, to a certain extent with more recent model selection procedures based on regularization such as the lasso and least angle regression (Tibshirani 1996; Tibshirani, Johnstone, Hastie, and Efron 2004), where attention focusses only on those models that are identified by the path taken through the model space. In the lasso, as the tuning parameter λ is varied from zero to ∞, different regression parameters remain non-zero, thus generating a path through the set of possible regression models, starting with the largest "optimal" model when $\lambda = 0$ to the smallest possible model when $\lambda = \infty$, typically the null model because the intercept is not penalized. The lasso selects that model on the lasso path at a single λ value, that minimizes one of the many possible criteria (such as 5-fold cross-validation, or the prediction error) or by determining the model on the lasso path that minimizes an information criterion (for example BIC).

An alternative to stepwise or regularization procedures is to perform exhaustive searches of the model space. While exhaustive searches avoid the issue of local minima, they are computationally expensive, growing exponentially in the number of variables p, with more than a thousand models when $p = 10$ and a million when $p = 20$. The methods provided in the

mplot package and described in the remainder of the article go beyond stepwise procedures by incorporating exhaustive searches where feasible and using resampling techniques to provide an indication of the stability of the selected model. The **mplot** package can feasibly handle up to 50 variables in linear regression models and a similar number for logistic regression models when an appropriate transformation (described in Section 7.2) is implemented.

3. Model stability and variable inclusion plots

The main contributions of the **mplot** package are model stability plots and variable inclusion plots, implemented through the `vis()` function, and the simplified adaptive fence for linear and generalized linear models via the `af()` function which is discussed in Section 4.

Our methods generate large amounts of raw data about the fitted models. While the print and summary output from both functions provide suggestions as to which models appear to be performing best, it is not our intention to have researchers simply read off the "best" model from the output. The primary purpose of these techniques is to help inform a researcher's model selection choice. As such, the real value in using these functions is in the extensive plot methods provided that help visualize the results and get new insights. This is reflected in the choice of the name `vis`, short for visualize, as this is the ultimate goal – to visualize the stability of the model selection process.

3.1. Model stability plots

In order to generate model stability and variable inclusion plots, the first step is to generate a '`vis`' object using the `vis()` function. To generate a '`vis`' object for the artificial data example the fitted full model object along with some optional arguments are passed to the `vis()` function.

```
R> lm.art <- lm(y ~ ., data = artificialeg)
R> vis.art <- vis(lm.art, B = 150, redundant = TRUE, nbest = "all",
+    seed = 2017)
```

The `B = 150` argument provided to the `vis()` function tells us that we want to perform 150 bootstrap replications. See Murray *et al.* (2013) for more detail on the use of exponential weights in bootstrap model selection. Specifying `redundant = TRUE` is unnecessary, as it is the default option; it ensures that an extra variable, randomly generated from a standard normal distribution and hence completely unrelated to the true data generating process, is added to the full model. This extra redundant variable can be used as a baseline comparison in the variable inclusion plots. The `nbest` argument controls how many models with the smallest $\hat{Q}(\alpha)$ for each model size $k = 1, \ldots, p$ are recorded. It can take an integer argument or specifying `nbest = "all"` ensures that all possible models are displayed when the plot method is called, as shown in the top left panel of Figure 2. Typically researchers do not need to visualize the entire model space and in problems with larger numbers of candidate variables it is impractical to store and plot results for all models. The default behavior of the `vis()` function is to set `nbest = 5`, essentially highlighting the maximum enveloping lower convex curve of Murray *et al.* (2013). Finally, the `seed` argument facilitates reproducibility in the parallelized bootstrap resampling.

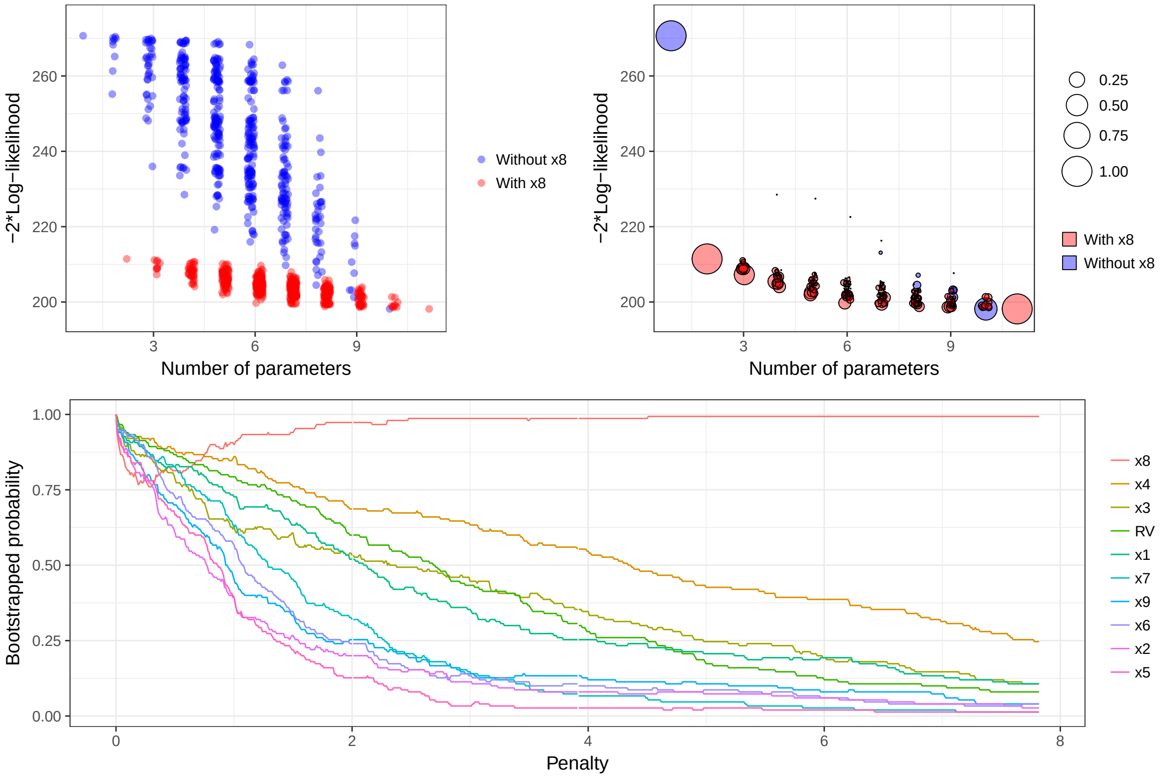


Figure 2: Results of calls to `plot(vis.art, interactive = FALSE)` with additional arguments `which = "lvk"` in the top left, `which = "boot"` in the top right and `which = "vip"` at the bottom.

The simplest visualization of the model space is to plot a measure of description loss against model complexity for all possible models, a special implementation is the Mallows C_p plot (Mallows 2000). This is done using the argument `which = "lvk"` to the plot function applied to a '`vis`' object. The string `"lvk"` is short for loss versus k, the dimension of the model.

```
R> plot(vis.art, interactive = FALSE, highlight = "x8", which = "lvk")
```

The result of this function can be found in the top left panel of Figure 2. The `highlight` argument is used to differentiate models that contain a particular variable from those that do not. This is an implementation of the "enriched scatter plot" of Murray *et al.* (2013). There is a clear separation between models that contain x_8 and those that do not, that is, all models containing x_8 (shown as red points) are clustered towards the bottom whereas the models without x_8 (blue points) are positioned above in a separate cluster. There is no similar separation for the other explanatory variables (not shown). These results strongly suggest that x_8 is the single most important variable. For clarity the points have been jittered slightly along the horizontal axis, though the model sizes remain clearly differentiated.

Rather than performing a single pass over the model space and plotting the description loss against model size, a more nuanced and discerning approach is to use a (exponential weighted) bootstrap to determine how often various models achieve the minimal loss for each model size. The advantage of the bootstrap approach is that it gives a measure of model stability for each model size as promoted by Meinshausen and Bühlmann (2010), Müller and Welsh (2010) and Murray *et al.* (2013).

The weighted bootstrap has two key benefits over the residual or nonparametric bootstrap: First, the weighted bootstrap always yields observable responses which is particularly relevant when these observable values are restricted to be integers (as in many generalized linear models), or, when y values are naturally bounded, say to be observed on the interval 0 to 1; Second, the weighted bootstrap does not suffer from separation issues that regularly occur in logistic and other models. The pairs bootstrap also yields observable responses and can be thought of as a special (boundary) case of the weighted bootstrap where some weights are allowed to be exactly zero, which can create a separation issue in logistic models. Furthermore, Shao and Tu (1995, Chapter 10) show how the weighted bootstrap is also closely related to the Bayesian bootstrap. Therefore, we have chosen to implement the weighted bootstrap because it is a simple, elegant method that appears to work well. Specifically, we utilize the exponential weighted bootstrap where the observations are reweighted with weights drawn from an exponential distribution with mean 1 (see Murray *et al.* 2013 for more detail).

To visualize the results of the exponential weighted bootstrap, the `which = "boot"` argument needs to be passed to the plot call on a '`vis`' object. The `highlight` argument can again be used to distinguish between models with and without a particular variable. Each circle represents a model with a non-zero bootstrap probability, that is, each model that was selected as the best model of a particular dimension in at least one bootstrap replication. Furthermore, the area of each circle is proportional to the corresponding model's bootstrapped selection probability.

Figure 2 is an example of a model stability plot for the artificial data set. The null model, the full model and the simple linear regression of y on x_8 all have bootstrap probabilities equal to one. While there are alternatives to the null and full model their inclusion in the plot serves two main purposes. Firstly, to gauge the potential range in description loss and secondly to provide a baseline against which to compare other circles to see if any approach a similar size, which would indicate that those are dominant models of a given model dimension. In Figure 2, there appears to be dominant models in models of size three and ten, as demonstrated by one of the circles being substantially larger than the other circles with models of the same size. However, in model dimensions of between four and nine, there are no clearly dominant models, that is, within each model size there are no models that are selected much more commonly than the alternatives.

A print method is available for '`vis`' objects which prints the model formula, log-likelihood and proportion of times that a given model was selected as the "best" model within each model size. The default minimum probability of a model being selected before it gets printed is 0.3, though this can be customized by passing a `min.prob` argument to the `print` function.

```
R> print(vis.art, min.prob = 0.25)

                           name prob logLikelihood
                            y~1 1.00       -135.33
                           y~x8 1.00       -105.72
                        y~x4+x8 0.44       -103.63
 y~x1+x2+x3+x4+x5+x6+x7+x9+RV 0.55        -99.09
```

The output above, reinforces what we know from the top right panel of Figure 2. The null model is always selected and in models of size two a regression of y on x_8 is always selected. In models of size three the most commonly selected model is `y~x4+x8`, selected 44% of the time. Interestingly, in models of size ten, the most commonly selected model does not contain x_8. We will see in the next section that this phenomenon is related to the failure of stepwise variable selection with this data set.

3.2. Variable inclusion plots

Rather than visualizing a loss measure against model size, it can be instructive to consider which variables are present in the overall "best" model over a set of bootstrap replications. To facilitate comparison between models of different sizes we use the GIC, Equation 2, which includes a penalty term for the number of variables in each model.

Using the same exponential weighted bootstrap replications as in the model selection plots, we have a set of B bootstrap replications and for each model size we know which model has the smallest description loss. This information is used to determine which model minimizes the GIC over a range of values of the penalty parameter, λ, in each bootstrap sample. For each value of λ, we extract the variables present in the "best" models over the B bootstrap replications and calculate the corresponding bootstrap probabilities that a given variable is present. These calculations are visualized in a variable inclusion plot (VIP) as introduced by Müller and Welsh (2010) and Murray *et al.* (2013). The VIP shows empirical inclusion probabilities as a function of the penalty multiplier λ. The probabilities are calculated by observing how often each variable is retained in B exponential weighted bootstrap replications. Specifically, for each bootstrap sample $b = 1, \ldots, B$ and each penalty multiplier λ, the chosen model, $\hat{\alpha}^b_\lambda \in \mathcal{A}$, is that which achieves the smallest $\mathrm{GIC}(\alpha, \lambda; \mathbf{w}_b) = \hat{Q}^b(\alpha) + \lambda p_\alpha$, where $\mathbf{w}_b$ is the n-vector of independent and identically distributed exponential weights (we refer to Section 2.5 in Murray *et al.* 2013 for more information on the weighted bootstrap). The inclusion probability for variable x_j is estimated by $B^{-1} \sum_{i=1}^{B} \mathbb{I}\{j \in \hat{\alpha}^b_\lambda\}$, where $\mathbb{I}\{j \in \hat{\alpha}^b_\lambda\}$ is one if x_j is in the final model and zero otherwise. Following Murray *et al.* (2013), the default range of λ values is $\lambda \in [0, 2\log(n)]$ as this includes most standard values used for the penalty parameter.

The example shown in the bottom panel of Figure 2 is obtained using the `which = "vip"` argument to the plot function. As expected, when the penalty parameter is equal to zero, all variables are included in the model; the full model achieves the lowest description loss, and hence minimizes the GIC when there is no penalization. As the penalty parameter increases, the inclusion probabilities for individual variables typically decrease as more parsimonious models are preferred. In the present example, the inclusion probabilities for the x_8 variable exhibit a sharp decrease at low levels of the penalty parameter, but then increase steadily as a more parsimonious model is sought. This pattern helps to explain why stepwise model selection chose the larger model with all the variables except x_8 – there exists a local minimum. Hence, for large models the inclusion of x_8 adds no additional value over having all the other explanatory variables in the model.

It is often instructive to visualize how the inclusion probabilities change over the range of penalty parameters. The ordering of the variables in the legend corresponds to their average inclusion probability over the whole range of penalty values. We have also added an independent standard Gaussian random variable to the model matrix as a redundant variable (`RV`). This provides a baseline to help determine which inclusion probabilities are "significant" in the sense that they exhibit a different behavior to the `RV` curve. Variables with inclusion probabilities near or below the `RV` curve can be considered to have been included by chance.

To summarize, VIPs continue the model stability theme. Rather than simply using a single penalty parameter associated with a particular information criterion, for example the AIC with $\lambda = 2$, our implementation of VIPs adds considerable value by allowing us to learn from a range of penalty parameters. Furthermore, we are able to see which variables are most often included over a number of bootstrap samples. An alternative approach to assessing model stability, the simplified adaptive fence, is introduced in the next section.

4. The simplified adaptive fence

The fence, first introduced by Jiang *et al.* (2008), is built around the inequality

$$\hat{Q}(\alpha) - \hat{Q}(\alpha_f) \leq c,$$

where $\hat{Q}$ is an empirical measure of description loss, α is a candidate model and α_f is the baseline, "full" model. The procedure attempts to isolate a set of "correct models" that satisfy the inequality. A model α^*, is described as "within the fence" if $\hat{Q}(\alpha^*) - \hat{Q}(\alpha_f) \leq c$. From the set of models within the fence, the one with minimum dimension is considered optimal. If there are multiple models within the fence at the minimum dimension, then the model with the smallest $\hat{Q}(\alpha)$ is selected. For a recent review of the fence and related methods, see Jiang (2014).

The implementation we provide in the **mplot** package is inspired by the simplified adaptive fence proposed by Jiang *et al.* (2009), which represents a significant advance over the original fence method proposed by Jiang *et al.* (2008). The key difference is that the parameter c is not fixed at a certain value, but is instead adaptively chosen. Simulation results have shown that the adaptive method improves the finite sample performance of the fence, see Jiang *et al.* (2008, 2009).

The adaptive fence procedure entails bootstrapping over a range of values of the parameter c. For each value of c a parametric bootstrap is performed under α_f. For each bootstrap sample we identify the smallest model inside the fence, $\hat{\alpha}(c)$. Jiang *et al.* (2009) suggest that if there is more than one model, choose the one with the smallest $\hat{Q}(\alpha)$. Define the empirical probability of selecting model α for a given value of c as $p^*(c, \alpha) = \mathsf{P}^*\{\hat{\alpha}(c) = \alpha\}$. Hence, if B bootstrap replications are performed, $p^*(c, \alpha)$ is the proportion of times that model α is selected. Finally, define an overall selection probability, $p^*(c) = \max_{\alpha \in \mathcal{A}} p^*(c, \alpha)$ and plot $p^*(c)$ against c to find the first peak. The value of c at the first peak, c^*, is then used with the standard fence procedure on the original data.

Our implementation is provided through the `af()` function and associated plot methods. An example with the artificial data set is given in Figure 3 which is generated using the following code.

```
R> af.art <- af(lm.art, B = 150, n.c = 50, seed = 2017)
R> plot(af.art, best.only = TRUE, legend.position = "right", model.wrap = 4)
R> summary(af.art)

Call:
af(mf = lm.art, B = 150, n.c = 50, seed = 2017)
```

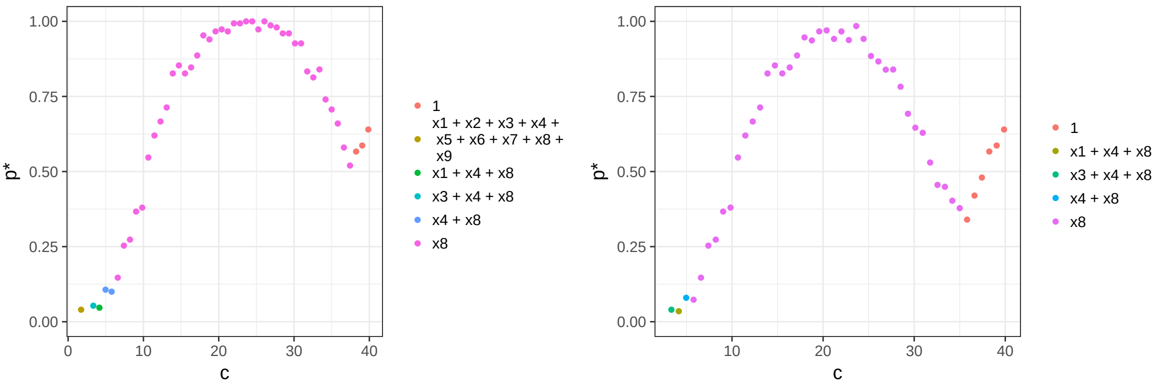


Figure 3: Result of a call to `plot(af.art)` with additional arguments `best.only = TRUE` on the left and `best.only = FALSE` on the right. The more rapid decay after the x_8 model is typical of using `best.only = FALSE` where the troughs between candidate/dominant models are more pronounced.

```
Adaptive fence model (c*=23.6):
y ~ x8

Model sizes considered: 1 to 11 (including intercept).
```

The arguments indicate that we perform $B = 150$ bootstrap resamples, over a grid of 50 values of the parameter c. In this example, there is only one peak, with $c^* = 23.6$.

One might expect that there should be a peak corresponding to the full model at $c = 0$, but this is avoided by the inclusion of at least one redundant variable. Any model that includes the redundant variable is known to not be a "true" model and hence is not included in the calculation of $p^*(c)$. This issue was first identified and addressed by Jiang *et al.* (2009).

There are a number of key differences between our implementation and the method proposed by Jiang *et al.* (2009). Perhaps the most fundamental difference is in the philosophy underlying our implementation. Our approach is more closely aligned with the concept of model stability than with trying to pick a single "best" model. This can be seen through the plot methods we provide. Instead of simply using the plots to identify the first peak, we add a legend that highlights which models were the most frequently selected for each parameter value, that is, for each c value we identify which model gave rise to the $p^*(c)$ value. In this way, researchers can ascertain if there are regions of stability for various models. In the example given in Figure 3, there is no need to even define a c^* value, it is obvious from the plot that there is only one viable candidate model, a regression of y on x_8.

Our approach considers not just the best model of a given model size, but also allows users to view a plot that takes into account the possibility that more than one model of a given model size is within the fence. The `best.only = FALSE` option when plotting the results of the adaptive fence is a modification of the adaptive fence procedure which considers all models of a particular size that are within the fence when calculating the $p^*(c)$ values. In particular, for each value of c and for each bootstrap replication, if a candidate model is found inside the fence, then we look to see if there are any other models of the same size that are also within the fence. If no other models of the same size are inside the fence, then that model is allocated a weight of 1. If there are two models inside the fence, then the best model is allocated a weight of 1/2. If three models are inside the fence, the best model gets a weight of 1/3, and so on. After B bootstrap replications, we aggregate the weights by summing over the various models. The $p^*(c)$ value is the maximum aggregated weight divided by the number of bootstrap replications. This correction penalizes the probability associated with the best model if there were other models of the same size inside the fence. The rationale is that if a model has no redundant variables then it will be the only model of that size inside the fence over a range of values of c. This results in more pronounced peaks which can help to determine the location of the correct peak and identify the optimal c^* value or more clearly differentiate regions of model stability. This can be seen in the right hand panel of Figure 3.

Another key difference is that our implementation is designed for linear and generalized linear models, rather than mixed models. As far as we are aware, this is the first time fence methods have been applied to such models. There is potential to add mixed model capabilities to future versions of the **mplot** package, but computational speed is a major hurdle that needs to be overcome. The current implementation is made computationally feasible through the use of the **leaps** (Lumley and Miller 2017) and **bestglm** (McLeod and Xu 2017) packages and the use of parallel processing, as discussed in Section 6.

We have also provided an optional initial stepwise screening method that can help limit the range of c values over which to perform the adaptive fence procedure. The initial stepwise procedure performs forward and backward stepwise model selection using both the AIC and BIC. From the four candidate models, we extract the size of smallest and largest models, k_L and k_U respectively. To obtain a sensible range of c values we consider the set of models with dimension between $k_L - 2$ and $k_U + 2$. Due to the inherent limitations of stepwise procedures, outlined in Section 2, it can be useful to check `initial.stepwise = FALSE` with a small number of bootstrap replications over a sparse grid of c values to ensure that the `initial.stepwise = TRUE` has produced a reasonable region.

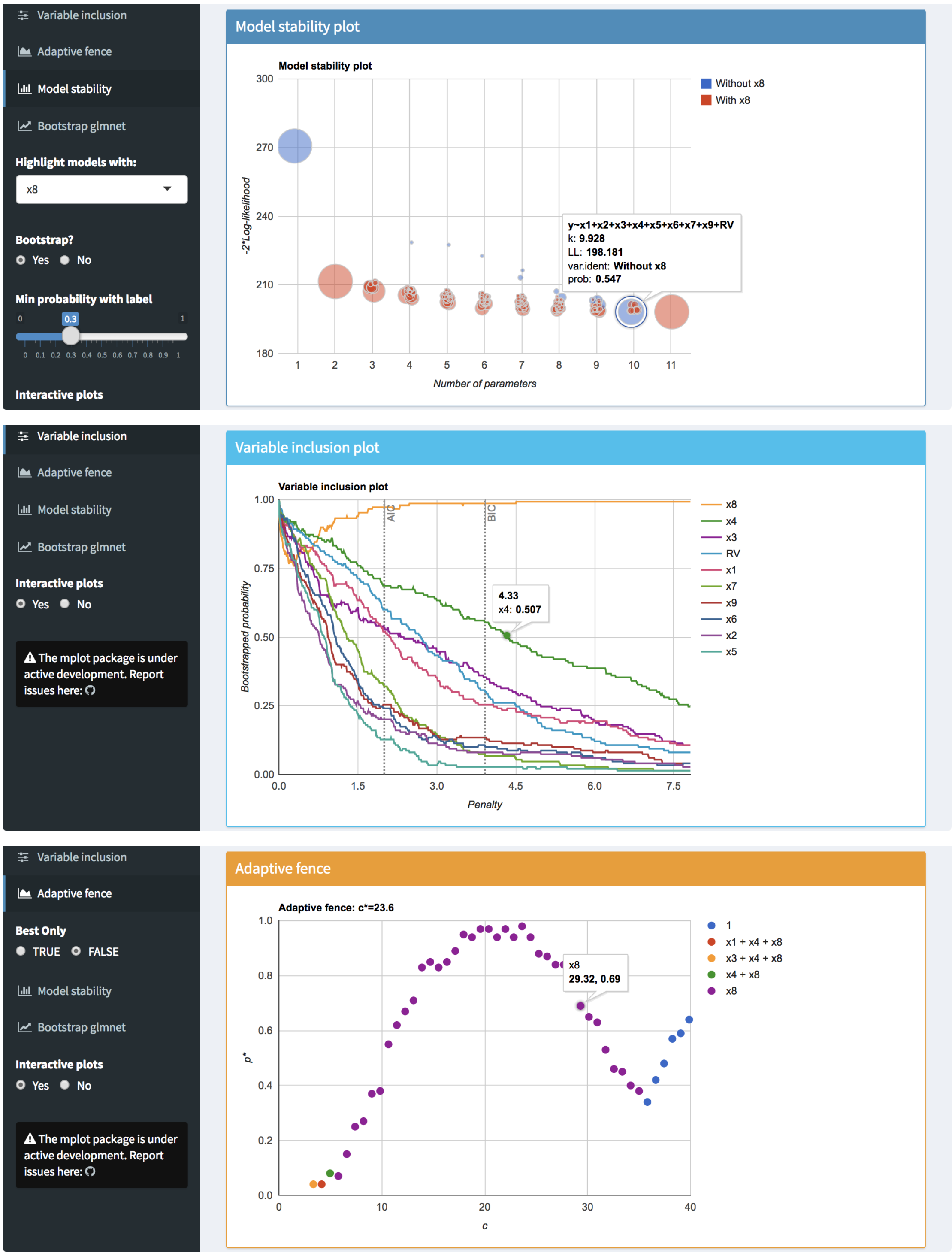
5. Interactive graphics

To facilitate that researchers can more easily gain value from the static plots given in Figures 2 and 3 and to help them interact with the model selection problem more closely, we have provided a set of interactive graphics based on the **googleVis** package and wrapped them in a **shiny** user interface. It is still quite novel for a package to provide a **shiny** interface for its methods, but there is precedent, see, for example McMurdie and Holmes (2013) or Gabry (2017).

Among the most important contributions of these interactive methods is: the provision of tooltips to identify the models and/or variables; pagination of the legend for the variable inclusion plots; and a way to quickly select which variable to highlight in the model stability plots. These interactive plots can be generated when the `plot()` function is run on an '`af`' or '`vis`' object by specifying `interactive = TRUE`.

The **mplot** package takes interactivity a step further, embedding these plots within a **shiny** web interface. This is done through a call to the `mplot()` function, which requires the full fitted model as the first argument and then a '`vis`' object and/or '`af`' object (in any order).

```
R> mplot(lm.art, vis.art, af.art)
```



Highlight models with: x8

Bootstrap? ○ Yes ● No

Min probability with label: 0.3 (0 – 1; 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1)

y~x1+x2+x3+x4+x5+x6+x7+x9+RV
k: 9.928
LL: 198.181
var.ident: Without x8
prob: 0.547

Interactive plots ○ Yes ● No

⚠ The mplot package is under active development. Report issues here:

4.33
x4: 0.507

Best Only ○ TRUE ● FALSE

Interactive plots ○ Yes ● No

⚠ The mplot package is under active development. Report issues here:

x8
29.32, 0.69

Figure 4: Screen shots from the web interface generated using `mplot()`.

Note that the `vis()` and `af()` functions need to be run and the results stored prior to calling the `mplot()` function. The result of a call to this function is a web page built using the **shiny** package with **shinydashboard** stylings (Chang *et al.* 2017; Chang 2017). Figure 4 shows a series of screen shots for the artificial example, equivalent to Figures 2 and 3, resulting from the above call to `mplot()`.

The top panel of Figure 4 shows a model stability plot where the full model that does not contain x_8 has been selected and a tooltip has been displayed. It gives details about the model specification, the log-likelihood and the bootstrap selection probability within models of size 10. The tooltip makes it easier for users to identify which variables are included in dominant models than the static plot equivalent. On the left hand side of the **shiny** interface, a drop down menu allows users to select the variable to be highlighted. This is passed through the `highlight` argument discussed in Section 3.1. Models with the highlighted variable are displayed as red circles whereas models without the highlighted variable are displayed as blue circles. The ability for researchers to quickly and easily see which models in the stability plot contain certain variables enhances their understanding of the relative importance of different components in the model. Selecting "No" at the "Bootstrap?" radio buttons yields the plot of description loss against dimension shown in the top left panel of Figure 2.

The middle panel of Figure 4 is a screen shot of an interactive variable inclusion plot. When the mouse hovers over a line, the tooltip gives information about the bootstrap inclusion probability and which variable the line represents. Note that in comparison to the bottom panel of Figure 2, the legend is now positioned outside of the main plot area. When the user clicks a variable in the legend, the corresponding line in the plot is highlighted. This can be seen in Figure 4, where the x_8 variable in the legend has been clicked and the corresponding x_8 line in the variable inclusion plot has been highlighted. The highlighting is particularly useful with the redundant variable, so it can easily be identified. If the number of predictor variables is such that they no longer fit neatly down the right hand side of the plot, they simply paginate, that is an arrow appears allowing users to toggle through to the next page of variables. This makes the interface cleaner and easier to interpret than the static plots. Note also the vertical lines corresponding to traditional AIC and BIC penalty values.

The bottom panel of Figure 4 is an interactive adaptive fence plot. The tooltip for a particular point gives information about the explanatory variable(s) in the model, the $\alpha^* = \arg\max_{\alpha\in\mathcal{A}} p^*(c,\alpha)$ value and the $(c, p^*(c))$ pair that has been plotted. Hovering or clicking on a model in the legend highlights all the points in the plot corresponding to that model. In this example, the x_8 legend has been clicked on and an additional circle has been added around all points corresponding to the model that has x_8 as the sole explanatory variable. The **shiny** interface on the left allows users to toggle between `best.only = TRUE` and `best.only = FALSE`.

The interactive graphics and **shiny** interface are most useful in the exploratory stage of model selection. Once the researcher has found the most informative plot through interactive analysis, the more traditional static plots may be used in a formal write up of the problem.

6. Timing

Any bootstrap model selection procedure is time consuming. However, for linear models, we have leveraged the efficiency of the branch-and-bound algorithm provided by the **leaps** package (Miller 2002; Lumley and Miller 2017). The **bestglm** package is used for GLMs; but in the absence of a comparably efficient algorithm the computational burden is much greater (McLeod and Xu 2017).

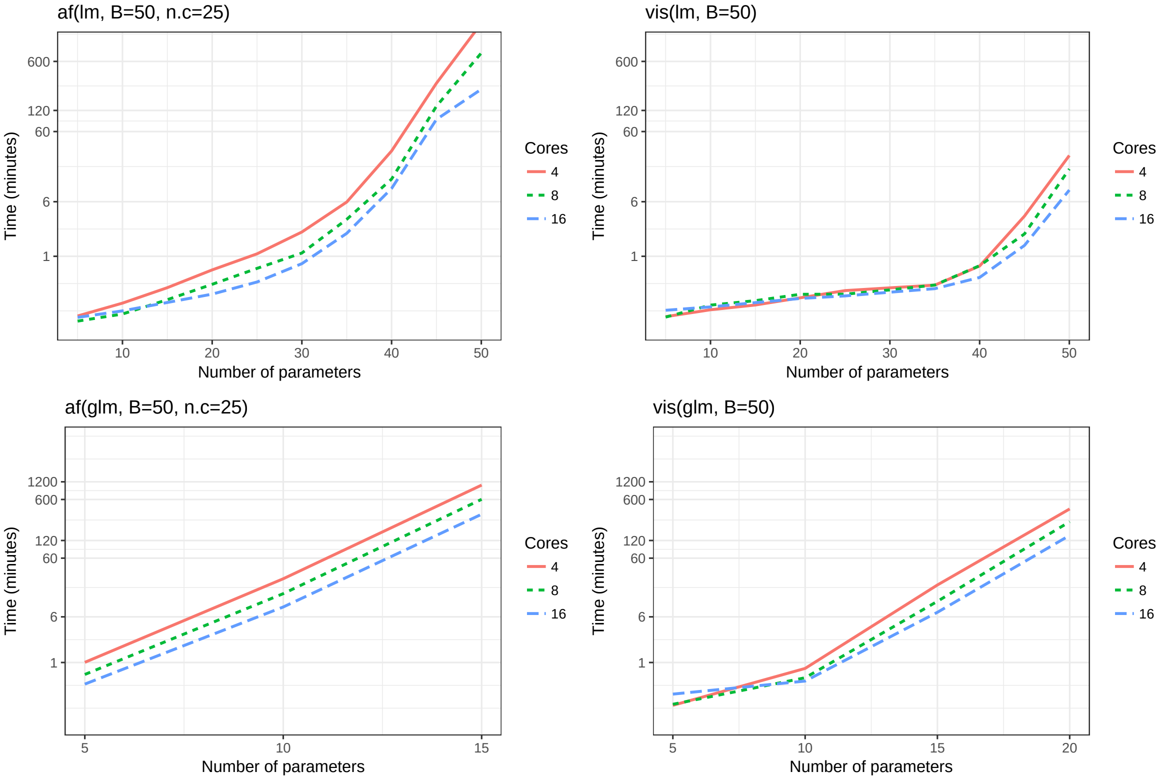


Figure 5: Average time required to run the `af()` and `vis()` functions when $n = 100$. A binomial regression was used for the GLM example.

Furthermore, we have taken advantage of the embarrassingly parallel nature of bootstrapping, utilizing the **doParallel**, **foreach** and **doRNG** packages to provide cross platform multicore support, available through the `cores` argument (Kane, Emerson, and Weston 2013; Revolution Analytics and Weston 2017, 2015; Gaujoux 2017). By default it will detect the number of cores available on your computer and leave one free.

Figure 5 shows the timing results of simulations run for standard use scenarios with 4, 8 or 16 cores used in parallel. Each observation plotted is the average of four runs of a given model size. The simulated models had a sample size of $n = 100$ with $5, 10, \ldots, 50$ candidate variables, of which 30% were active in the true model.

The results show both the `vis()` and `af()` functions are quite feasible on standard desktop hardware with 4 cores even for moderate dimensions of up to 40 candidate variables. The adaptive fence takes longer than the `vis()` function, though this is to be expected as the effective number of bootstrap replications is `B` × `n.c`, where `n.c` is the number of divisions in the grid of the parameter c.

The results for GLMs are far less impressive, even when the maximum dimension of a candidate solution is set to `nvmax = 10`. In its current implementation, the adaptive fence is only really feasible for models of around 10 predictors and the `vis()` function for 15. Future improvements could see approximations of the type outlined by Hosmer, Jovanovic, and Lemeshow (1989) to bring the power of the linear model branch-and-bound algorithm to GLMs. An example of how this works in practice is given in Section 7.2.

An alternative approach for high dimensional models would be to consider subset selection with convex relaxations as in Shen, Pan, and Zhu (2012) or combine bootstrap model selection with regularization. In particular, we have implemented variable inclusion plots and model stability plots for package **glmnet** (Friedman, Hastie, and Tibshirani 2010). In general, this is very fast for models of moderate dimension, but it does not consider the full model space. Restrictions within the **glmnet** package imply that this approach is only applicable to linear models, binomial logistic regression, and Poisson regression with the log link function. The **glmnet** package also allows for `"multinomial"`, `"cox"`, and `"mgaussian"` families, though we have not yet incorporated these into the **mplot** package.

7. Real examples

7.1. Diabetes example

Table 1 shows a subset of the diabetes data used in Efron *et al.* (2004). There are 10 explanatory variables, including age (`age`), sex (`sex`), body mass index (`bmi`) and mean arterial blood pressure (`map`) of 442 patients as well as six blood serum measurements (`tc`, `ldl`, `hdl`, `tch`, `ltg` and `glu`). The response is a measure of disease progression one year after the baseline measurements.

Figure 6 shows the results of the main methods for the diabetes data obtained using the following code.

```
R> lm.d <- lm(y ~ ., data = diabetes)
R> vis.d <- vis(lm.d, B = 200, seed = 1)
R> af.d <- af(lm.d, B = 200, n.c = 100, c.max = 100, seed = 1)
R> plot(vis.d, interactive = FALSE, which = "vip")
R> plot(vis.d, interactive = FALSE, which = "boot", max.circle = 10,
+    highlight = "hdl") + scale_x_continuous(breaks = c(2, 4, 6, 8, 10, 12))
R> plot(af.d, interactive = FALSE, best.only = TRUE,
+    legend.position = "right")
R> plot(af.d, interactive = FALSE, best.only = FALSE,
+    legend.position = "right")
```

A striking feature of the variable inclusion plot is the non-monotonic nature of the `hdl` line.

					Serum measurements						Response
Patient	age	sex	bmi	map	tc	ldl	hdl	tch	ltg	glu	y
1	59	2	32.1	101	157	93.2	38	4	4.9	87	151
2	48	1	21.6	87	183	103.2	70	3	3.9	69	75
3	72	2	30.5	93	156	93.6	41	4	4.7	85	141
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
441	36	1	30.0	95	201	125.2	42	5	5.1	85	220
442	36	1	19.6	71	250	133.2	97	3	4.6	92	57

Table 1: Measurements on 442 diabetes patients over 10 potential predictor variables and the response variable, a measure of disease progression after one year.

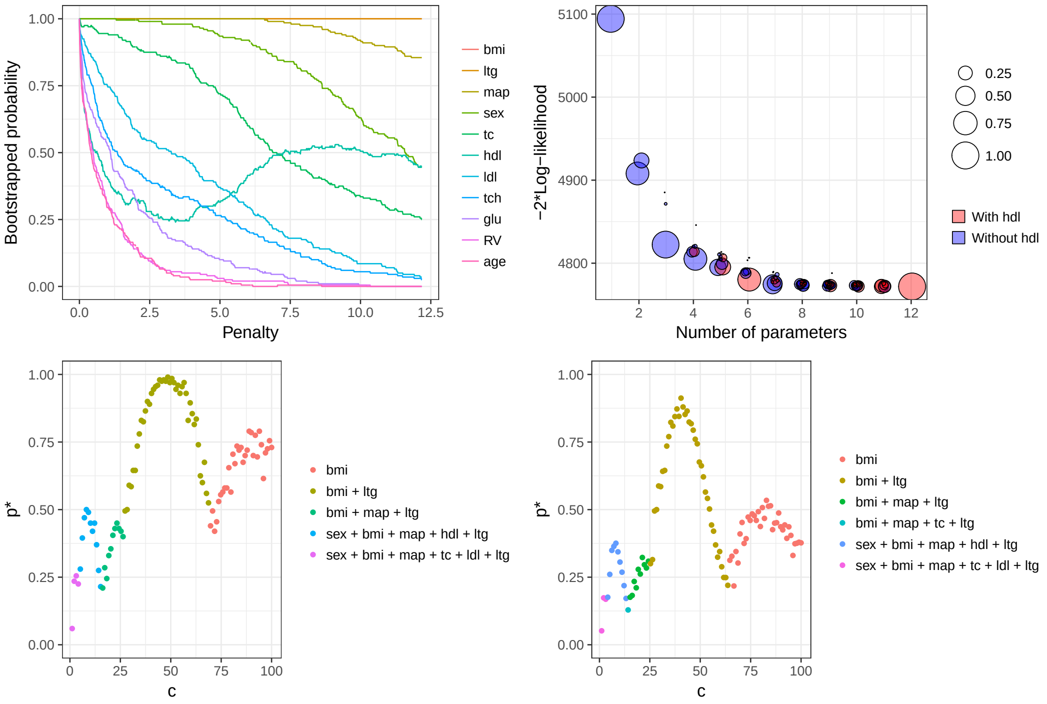


Figure 6: Diabetes main effects example.

As the penalty value increases, and a more parsimonious model is sought, the `hdl` variable is selected more frequently while at the same time other variables with similar information are dropped. Such paths occur when a group of variables contains similar information to another variable. The `hdl` line is a less extreme example of what occurs with x_8 in the artificial example (see Figure 2). The path for the age variable lies below the path for the redundant variable, indicating that it does not provide any useful information. The `bmi` and `ltg` paths are horizontal with a bootstrap probability of 1 for all penalty values indicating that they are very important variables, as are `map` and `sex`. From the variable inclusion plot alone, it is not obvious whether `tc` or `hdl` is the next most important variable. Some guidance on this issue is provided by the model stability and adaptive fence plots.

In order to determine which circles correspond to which models in the static version of the bootstrap stability plot, we need to consult the print output of the ‘`vis`’ object.

```
R> vis.d

                          name prob logLikelihood
                           y~1 1.00      -2547.17
                         y~bmi 0.70      -2454.02
                     y~bmi+ltg 0.98      -2411.20
                 y~bmi+map+ltg 0.70      -2402.61
              y~bmi+map+tc+ltg 0.36      -2397.48
             y~bmi+map+hdl+ltg 0.33      -2397.71
         y~sex+bmi+map+hdl+ltg 0.72      -2390.13
      y~sex+bmi+map+tc+ldl+ltg 0.48      -2387.30
```

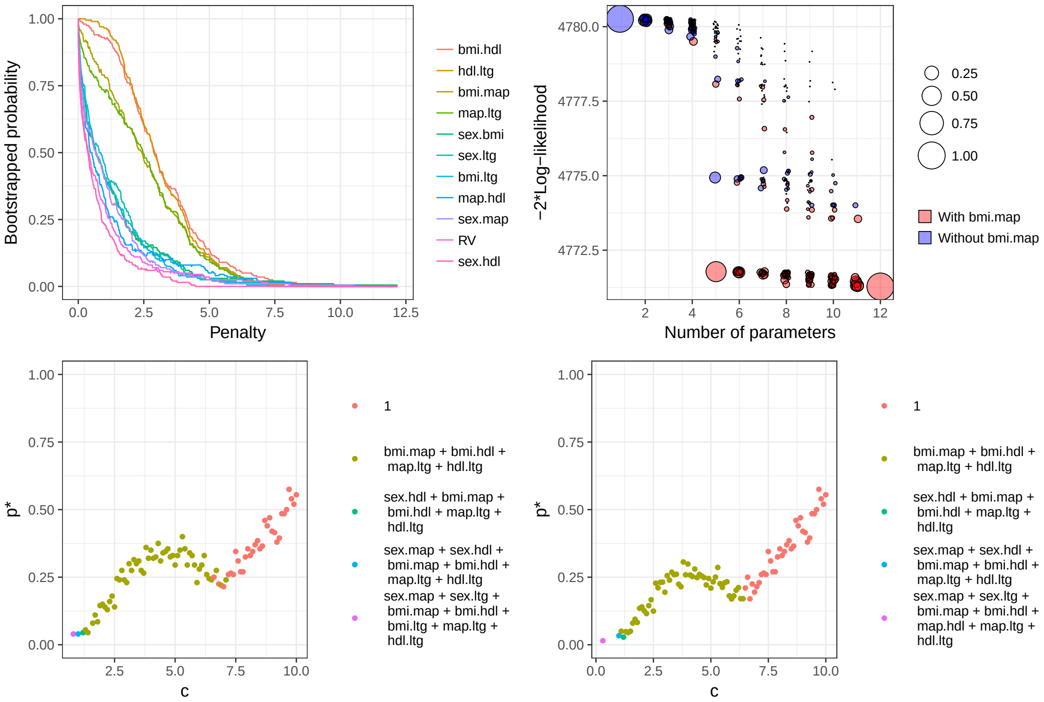


Figure 7: Diabetes interactions terms example.

As in the variable inclusion plots, it is clear that the two most important variables are `bmi` and `ltg`, and the third most important variable is `map`. In models of size four (including the intercept), the model with `bmi`, `ltg` and `map` was selected in 70% of bootstrap resamples. There is no clear dominant model in models of size five, with `tc` and `hdl` both competing to be included. In models of size six, the combination of `sex` and `hdl` with the core variables `bmi`, `map` and `ltg`, is the most stable option; it is selected in 72% of bootstrap resamples. As the size of the model space in dimension six is much larger than the size of the model space for dimension four, it could be suggested that the 0.72 empirical probability for the {`bmi`, `map`, `ltg`, `sex`, `hdl`} model is a stronger result than the 0.70 result for the {`bmi`, `ltg`, `map`} model.

The adaptive fence plots in the bottom row of Figure 6 show a clear peak for the model with just `bmi` and `ltg`. There are two larger models that also occupy regions of stability, albeit with much lower peaks. These are {`bmi`, `map`, `ltg`} and {`bmi`, `map`, `ltg`, `sex`, `hdl`} which also showed up as dominant models in the model stability plots. Contrasting `best.only = TRUE` in the lower left panel with `best.only = FALSE` in the lower right panel, we can see that the peaks tend to be more clearly distinguished, though the regions of stability remain largely unchanged.

Stepwise approaches using a forward search or backward search with the AIC or BIC all yield a model with {`bmi`, `map`, `ltg`, `sex`, `ldl`, `tc`}. This model was selected 48% of the time in models of size 7. The agreement between the stepwise methods may be comforting for the researcher, but it does not aid a discussion about what other models may be worth exploring.

An interactive version of the plots in Figure 6 is available at `http://garthtarr.com/apps/mplot`.

To incorporate interaction terms, we suggest selecting the main effects first, then regressing the relevant interaction terms on the residuals from the main effects model. This approach ensures that the main effects are always taken into account. In this example, we estimate the dominant model of dimension six and obtain the fitted residuals. The interaction terms are then regressed on the fitted residuals.

```
R> lm.d.main <- lm(y ~ sex + bmi + map + hdl + ltg, data = diabetes)
R> summary(lm.d.main)
R> db.main <- diabetes[, c("sex", "bmi", "map", "hdl", "ltg")]
R> db.main$y <- residuals(lm.d.main)
R> lm.d.int <- lm(y ~ . * . - sex - bmi - map - hdl - ltg, data = db.main)
R> vis.d.int <- vis(lm.d.int, B = 200)
R> af.d.int <- af(lm.d.int, B = 200, n.c = 100, c.max = 10, seed = 2017)
R> vis.d.int
```

```
                                 name prob logLikelihood
                                  y~1 1.00      -2390.13
 y~bmi.map+bmi.hdl+map.ltg+hdl.ltg 0.55      -2385.89
```

The result can be found in Figure 7. The variable inclusion plots suggest that the most important interaction terms are `hdl.ltg`, `bmi.hdl`, `map.ltg` and `bmi.map`. The model stability plot suggests that there are no dominant models of size 2, 3 or 4. Furthermore there are no models of size 2, 3 or 4 that make large improvements in description loss. There is a dominant model of dimension 5 that is selected in 55% of bootstrap resamples. The variables selected in the dominant model are {`bmi.map`, `bmi.hdl`, `map.ltg`, `hdl.ltg`}, which can be found in the print output above. Furthermore, this model does make a reasonable improvement in description loss, almost in line with the full model. This finding is reinforced in the adaptive fence plots where there are only two regions of stability, one for the null model and another for the {`bmi.map`, `bmi.hdl`, `map.ltg`, `hdl.ltg`} model. In this instance, the difference between `best.only = TRUE` and `best.only = FALSE` is minor.

Hence, as a final model for the diabetes example we suggest including the main effects {`bmi`, `map`, `ltg`, `sex`, `hdl`} and the interaction effects {`bmi.map`, `bmi.hdl`, `map.ltg`, `hdl.ltg`}. Further investigation can also be useful. For example, we could use cross-validation to compare the model with interaction effects, the model with just main effects and other simpler models that were identified as having peaks in the adaptive fence. Researchers should also incorporate their specialist knowledge of the predictors and evaluate whether or not the estimated model is sensible from a scientific perspective.

7.2. Birth weight example

The second example is the `birthwt` dataset from the **MASS** package (Venables and Ripley 2002) which has data on 189 births at the Baystate Medical Centre, Springfield, Massachusetts during 1986. The main variable of interest is low birth weight, a binary response variable `low` (Hosmer and Lemeshow 1989). We have taken the same approach to modelling the full model as in Venables and Ripley (2002, pp. 194–197), where `ptl` is reduced to a binary indicator of past history and `ftv` is reduced to a factor with three levels.

```
R> library("MASS")
R> bwt <- with(birthwt, {
+    race <- factor(race, labels = c("white", "black", "other"))
+    ptd <- factor(ptl > 0)
+    ftv <- factor(ftv)
+    levels(ftv)[-(1:2)] <- "2+"
+    data.frame(low = factor(low), age, lwt, race, smoke = (smoke > 0),
+      ptd, ht = (ht > 0), ui = (ui > 0), ftv)
+  })
R> options(contrasts = c("contr.treatment", "contr.poly"))
R> bw.glm <- glm(low ~ ., family = binomial, data = bwt)
R> round(coef(summary(bw.glm)), 2)

            Estimate Std. Error z value Pr(>|z|)
(Intercept)     0.82       1.24    0.66     0.51
age            -0.04       0.04   -0.96     0.34
lwt            -0.02       0.01   -2.21     0.03
raceblack       1.19       0.54    2.22     0.03
raceother       0.74       0.46    1.60     0.11
smokeTRUE       0.76       0.43    1.78     0.08
ptdTRUE         1.34       0.48    2.80     0.01
htTRUE          1.91       0.72    2.65     0.01
uiTRUE          0.68       0.46    1.46     0.14
ftv1           -0.44       0.48   -0.91     0.36
ftv2+           0.18       0.46    0.39     0.69
```

The ‘vis’ and ‘af’ objects are generated using the fitted full model object as an argument to the vis() and af() functions. The results are shown in Figure 8, where screen shots have been taken of the interactive plots because they display the larger set of variables more clearly than the static plot methods.

```
R> af.bw <- af(bw.glm, B = 150, c.max = 20, n.c = 40, seed = 1)
R> vis.bw <- vis(bw.glm, B = 150, seed = 1)
R> plot(vis.bw, which = "vip", interactive = TRUE)
R> plot(vis.bw, which = "boot", highlight = "htTRUE", interactive = TRUE)
R> plot(af.bw, interactive = TRUE)
R> print(vis.bw, min.prob = 0.10)

                   name prob logLikelihood
                 low~1 1.00       -117.34
           low~ptdTRUE 0.43       -110.95
               low~lwt 0.16       -114.35
            low~uiTRUE 0.13       -114.80
       low~age+ptdTRUE 0.15       -108.65
        low~lwt+htTRUE 0.13       -110.57
       low~lwt+ptdTRUE 0.11       -108.75
 low~lwt+ptdTRUE+htTRUE 0.15       -105.06
...
```

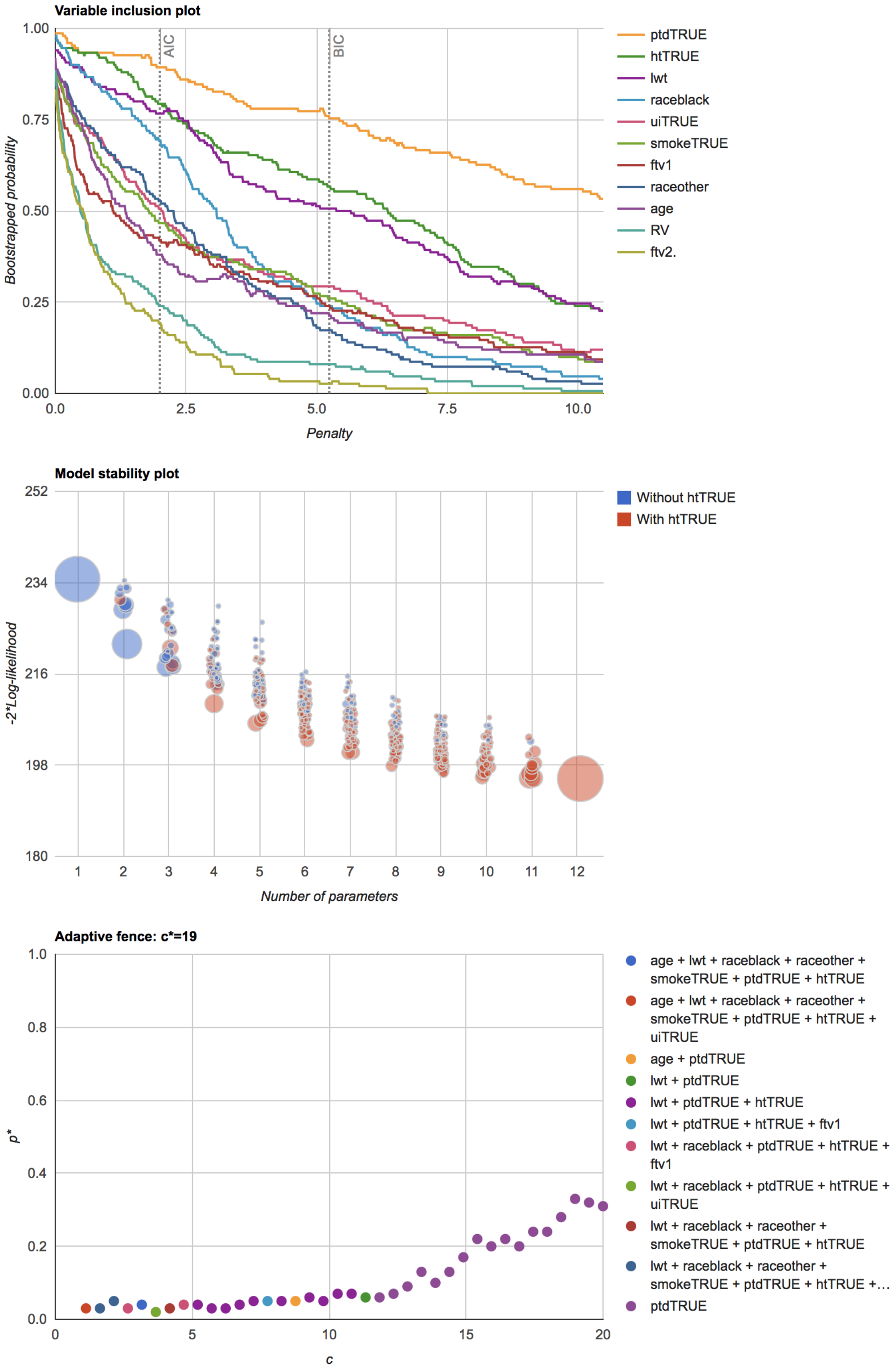


Figure 8: Birth weight example.

In this example, it is far less clear which is the best model, or if indeed a "best model" exists. The majority of the the curves in the variable inclusion plot lie above the redundant variable curve, except `ftv2+` the least important variable. It is possible to infer an ordering of variable importance from the variable inclusion plots, but there is no clear cutoff as to which variables should be included and which should be excluded. This is also clear in the model stability plots, where apart from the bivariate regression with `ptd`, there are no obviously dominant models.

In the adaptive fence plot, the only model more complex than a single covariate regression model that shows up with some regularity is the model with `lwt`, `ptd` and `ht`, though at such low levels, it is just barely a region of stability. This model also stands out slightly in the model stability plot, where it is selected in 16% of bootstrap resamples and has a slightly lower description loss than other models of the same dimension. It is worth recalling that the bootstrap resamples generated for the adaptive fence are separate from those generated for the model stability plots. Indeed the adaptive fence procedure relies on a parametric bootstrap, whereas the model stability plots rely on an exponential weighted bootstrap. Thus, to find some agreement between these methods is reassuring.

Stepwise approaches using AIC or BIC yield conflicting models, depending on whether the search starts with the full model or the null model. As expected, the BIC stepwise approach returns smaller models than AIC, selecting the single covariate logistic regression, `low ~ ptd`, in the forward direction and the larger model, `low ~ lwt + ptd + ht` when stepping backwards from the full model. Forward selection from the null model with the AIC yielded `low ~ ptd + age + ht + lwt + ui` whereas backward selection the slightly larger model, `low ~ lwt + race + smoke + ptd + ht + ui`. Some of these models appear as features in the model stability plots. Most notably the dominant single covariate logistic regression and the model with `lwt`, `ptd` and `ht` identified as a possible region of stability in the adaptive fence plot. The larger models identified by the AIC are reflective of the variable importance plot in that they show there may still be important information contained in a number of other variables not identified by the BIC approach.

Calcagno and de Mazancourt (2010) also consider this data set, but they allow for the possibility of interaction terms. Using their approach, they identify "two" best models

```
low ~ smoke + ptd + ht + ui + ftv + age + lwt + ui:smoke + ftv:age
low ~ smoke + ptd + ht + ui + ftv + age + lwt + ui:smoke + ui:ht + ftv:age
```

As a general rule, we would warn against the `. * .` approach, where all possible interaction terms are considered, as it does not consider whether or not the interaction terms actually make practical sense. Calcagno and de Mazancourt (2010) conclude that "Having two best models and not one is an extreme case where taking model selection uncertainty into account rather than looking for a single best model is certainly recommended!" The issue here is that the software did not highlight that these models are identical as the `ui:ht` interaction variable is simply a vector of zeros, and as such, is ignored by the GLM fitting routine.

As computation time can be an issue for GLMs, it is useful to approximate the results using weighted least squares (Hosmer *et al.* 1989). In practice this can be done by fitting the logistic regression and extracting the estimated logistic probabilities, $\hat{\pi}_i$. A new dependent variable is then constructed,

$$z_i = \log\left(\frac{\hat{\pi}_i}{1-\hat{\pi}_i}\right) + \frac{y_i - \hat{\pi}_i}{\hat{\pi}_i(1-\hat{\pi}_i)},$$

along with observation weights $v_i = \hat{\pi}_i(1 - \hat{\pi}_i)$. For any submodel α this approach produces the approximate coefficient estimates of Lawless and Singhal (1978) and enables us to use the **leaps** package to perform the computations for best subsets logistic regression as follows.

```
R> pihat <- fitted(bw.glm)
R> r <- residuals(bw.glm, type = "working")
R> z <- log(pihat / (1 - pihat)) + r
R> v <- pihat * (1 - pihat)
R> nbwt <- bwt
R> nbwt$z <- z
R> nbwt$low <- NULL
R> bw.lm <- lm(z ~ ., data = nbwt, weights = v)
R> bw.lm.vis <- vis(bw.lm, B = 150, seed = 1)
R> bw.lm.af <- af(bw.lm, B = 150, c.max = 20, n.c = 40, seed = 1)
R> plot(bw.lm.vis, which = "vip", interactive = TRUE)
R> plot(bw.lm.vis, which = "boot", highlight = "htTRUE", interactive = TRUE)
R> plot(bw.lm.af, interactive = TRUE)
```

The coefficients from `bw.lm` are identical to `bw.glm`. This approximation provides similar results, shown in Figure 9, in a fraction of the time.

8. Conclusion

In the rejoinder to their least angle regression paper, Efron *et al.* (2004) comment,

> "In actual practice, or at least in good actual practice, there is a cycle of activity between the investigator, the statistician and the computer ... The statistician examines the output critically, as did several of our commentators, discussing the results with the investigator, who may at this point suggest adding or removing explanatory variables, and so on, and so on."

We hope the suite of methods available in the **mplot** package adds valuable information to this cycle of activity between researchers and statisticians. In particular, providing statisticians and researchers alike with a deeper understanding of the relative importance of different models and the variables contained therein.

In the artificial example, we demonstrated a situation where giving the researcher more information in a graphical presentation can lead to choosing the "correct" model when standard stepwise procedures would have failed.

The diabetes data set suggested the existence of a number of different dominant models at various model sizes which could then be investigated further, for example, statistically using cross-validation to determine predictive ability, or in discussion with researchers to see which makes the most practical sense. In contrast, there are no clear models suggested for the birth weight example. The adaptive fence has no peaks, nor is there a clearly dominant model in the model stability plot even though all but one variable are more informative than the added redundant variable in the variable inclusion plot.

While the core of the **mplot** package is built around exhaustive searches, this becomes computationally infeasible as the number of variables grows. We have implemented similar vi-

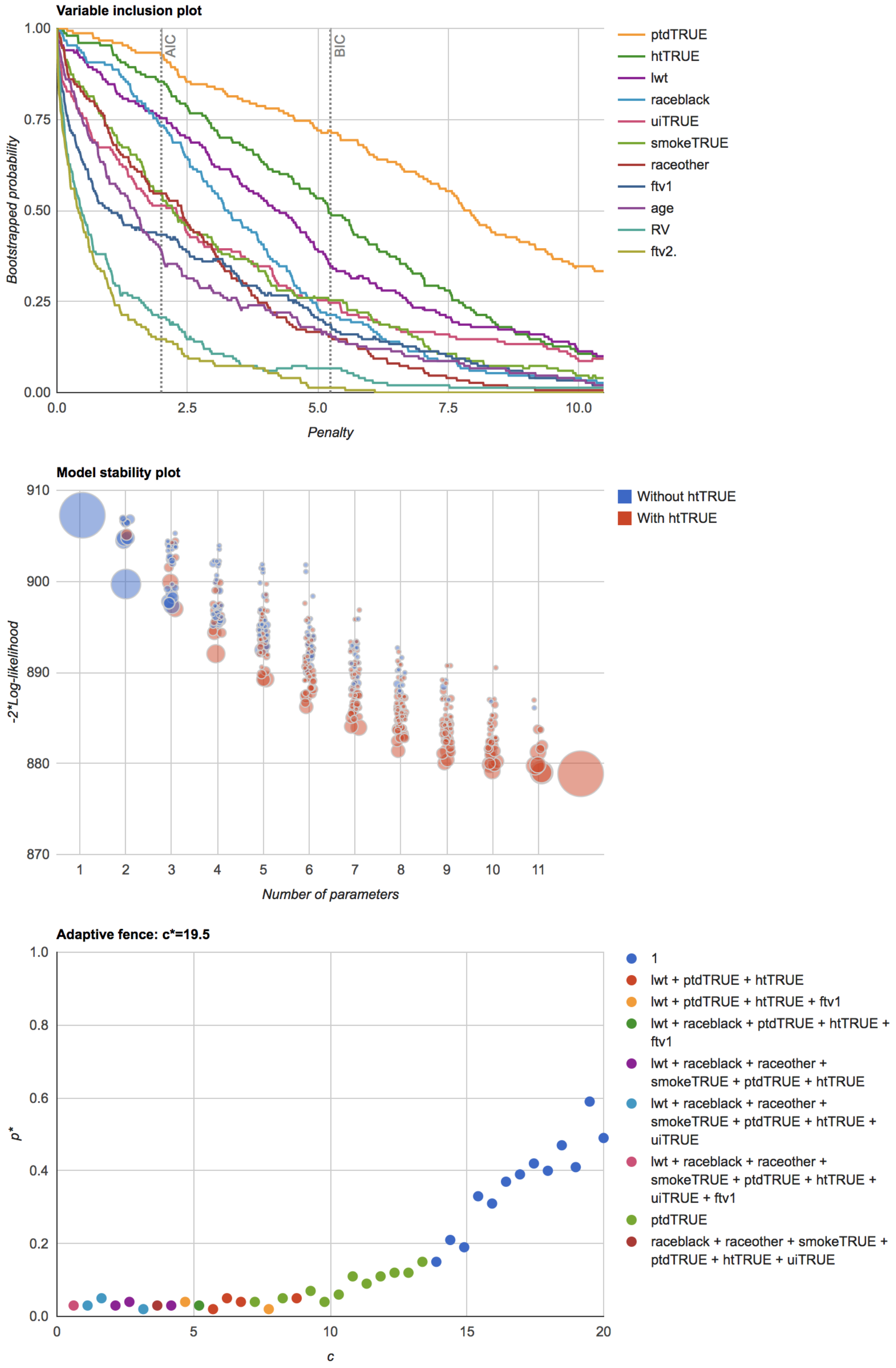


Figure 9: Birth weight example with linear model approximation.

sualizations to model stability plots and variable inclusion plots for package **glmnet** which brings the concept of model stability to much larger model sizes, though it will no longer be based around exhaustive searches.

The graphs provided by the **mplot** package are a major contribution. A large amount of information is generated by the various methods and the best way to interpret that information is through effective visualizations. For example, as is shown in Section 7.1, the path a variable takes through the variable inclusion plot is often more important than the average inclusion probability over the range of penalty values considered. It can also be instructive to observe when there are no peaks in the adaptive fence plot as this indicates that the variability of the log-likelihood is limited and no single model stands apart from the others. Such a relatively flat likelihood over various models would also be seen in the model stability plot where there was no dominant model over the range of model sizes considered.

Although interpretation of the model selection plots provided here is something of an "art", this is not something to shy away from. We accept and train young statisticians to interpret QQ-plots and residual plots. There is a wealth of information in our plots, particularly the interactive versions enhanced with the **shiny** interface, that can better inform a researchers' model selection choice.

Acknowledgments

The authors acknowledge the University of Sydney HPC service at The University of Sydney and the National Computational Infrastructure (NCI), which is supported by the Australian Government, for providing high performance computing resources that have contributed to the research results reported within this paper. Samuel Müller and Alan Welsh were supported by the Australian Research Council (DP140101259). We also gratefully acknowledge two anonymous referees and the editor for their helpful comments and suggestions for the paper and package.

References

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.

Chang W (2017). ***shinydashboard**: Create Dashboards with **shiny***. R package version 0.6.1, URL https://CRAN.R-project.org/package=shinydashboard.

Chang W, Cheng J, Allaire JJ, Xie Y, McPherson J (2017). ***shiny**: Web Application Framework for R*. R package version 1.0.5, URL https://CRAN.R-project.org/package=shiny.

Efron B, Hastie T, Johnstone I, Tibshirani R (2004). "Least Angle Regression." *The Annals of Statistics*, **32**(2), 407–451. doi:10.1214/009053604000000067.

Friedman JH, 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.

Gabry J (2017). ***shinystan**: Interactive Visual and Numerical Diagnostics and Posterior Analysis for for Bayesian Models.* R package version 2.4.0, URL https://CRAN.R-project.org/package=shinystan.

Gaujoux R (2017). ***doRNG**: Generic Reproducible Parallel Backend for **foreach** Loops.* R package version 1.6.6, URL https://CRAN.R-project.org/package=doRNG.

Gesmann M, de Castillo D (2011). "Using the Google Visualisation API with R." *The R Journal*, **3**(2), 40–44.

Harrell FE (2001). *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis.* Springer-Verlag, New York.

Hosmer DW, Jovanovic B, Lemeshow S (1989). "Best Subsets Logistic Regression." *Biometrics*, **45**(4), 1265–1270. doi:10.2307/2531779.

Hosmer DW, Lemeshow S (1989). *Applied Logistic Regression.* John Wiley & Sons, New York. doi:10.1002/0471722146.

Jiang J (2014). "The Fence Methods." *Advances in Statistics*, **2014**(83082), 1–14. doi:10.1155/2014/830821.

Jiang J, Nguyen T, Rao JS (2009). "A Simplified Adaptive Fence Procedure." *Statistics & Probability Letters*, **79**(5), 625–629. doi:10.1016/j.spl.2008.10.014.

Jiang J, Rao JS, Gu Z, Nguyen T (2008). "Fence Methods for Mixed Model Selection." *The Annals of Statistics*, **36**(4), 1669–1692. doi:10.1214/07-aos517.

Kane M, Emerson J, Weston S (2013). "Scalable Strategies for Computing with Massive Data." *Journal of Statistical Software*, **55**(14), 1–19. doi:10.18637/jss.v055.i14.

Konishi S, Kitagawa G (1996). "Generalised Information Criteria in Model Selection." *Biometrika*, **83**(4), 875–890. doi:10.1093/biomet/83.4.875.

Lawless JF, Singhal K (1978). "Efficient Screening of Nonnormal Regression Models." *Biometrics*, **34**(2), 318–327. doi:10.2307/2530022.

Lumley T, Miller A (2017). ***leaps**: Regression Subset Selection.* R package version 3.0, URL https://CRAN.R-project.org/package=leaps.

Mallows CL (2000). "Some Comments on C_p." *Technometrics*, **42**(1), 87–94. doi:10.1080/00401706.2000.10485984.

McLeod AI, Xu C (2017). ***bestglm**: Best Subset GLM.* R package version 0.36, URL https://CRAN.R-project.org/package=bestglm.

McMurdie PJ, Holmes S (2013). "**phyloseq**: An R Package for Reproducible Interactive Analysis and Graphics of Microbiome Census Data." *PLoS ONE*, **8**(4), e61217. doi:10.1371/journal.pone.0061217.

Meinshausen N, Bühlmann P (2010). "Stability Selection." *Journal of the Royal Statistical Society B*, **72**(4), 417–473. doi:10.1111/j.1467-9868.2010.00740.x.

Miller A (2002). *Subset Selection in Regression*. CRC Monographs on Statistics & Applied Probability. Chapman & Hall, Boca Raton. doi:10.1201/9781420035933.

Müller S, Scealy JL, Welsh AH (2013). "Model Selection in Linear Mixed Models." *Statistical Science*, **28**(2), 135–167. doi:10.1214/12-sts410.

Müller S, Vial C (2009). "Partially Linear Model Selection by the Bootstrap." *Australian & New Zealand Journal of Statistics*, **51**(2), 183–200. doi:10.1111/j.1467-842x.2009.00540.x.

Müller S, Welsh AH (2005). "Outlier Robust Model Selection in Linear Regression." *Journal of the American Statistical Association*, **100**(472), 1297–1310. doi:10.1198/016214505000000529.

Müller S, Welsh AH (2009). "Robust Model Selection in Generalized Linear Models." *Statistica Sinica*, **19**(3), 1155–1170.

Müller S, Welsh AH (2010). "On Model Selection Curves." *International Statistical Review*, **78**(2), 240–256. doi:10.1111/j.1751-5823.2010.00108.x.

Murray K, Heritier S, Müller S (2013). "Graphical Tools for Model Selection in Generalized Linear Models." *Statistics in Medicine*, **32**(25), 4438–4451. doi:10.1002/sim.5855.

Park H, Sakaori F, Konishi S (2014). "Robust Sparse Regression and Tuning Parameter Selection via the Efficient Bootstrap Information Criteria." *Journal of Statistical Computation and Simulation*, **84**(7), 1596–1607. doi:10.1080/00949655.2012.755532.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Revolution Analytics, Weston S (2015). ***foreach**: Foreach Looping Construct for R*. R package version 1.4.3, URL https://CRAN.R-project.org/package=foreach.

Revolution Analytics, Weston S (2017). ***doParallel**: Foreach Parallel Adaptor for the Parallel Package*. R package version 1.0.11, URL https://CRAN.R-project.org/package=doParallel.

Shang J, Cavanaugh JE (2008). "Bootstrap Variants of the Akaike Information Criterion for Mixed Model Selection." *Computational Statistics & Data Analysis*, **52**(4), 2004–2021. doi:10.1016/j.csda.2007.06.019.

Shao J (1996). "Bootstrap Model Selection." *Journal of the American Statistical Association*, **91**(434), 655–665. doi:10.2307/2291661.

Shao J, Tu D (1995). *The Jackknife and Bootstrap*. Springer-Verlag, New York.

Shen X, Pan W, Zhu Y (2012). "Likelihood-Based Selection and Sharp Parameter Estimation." *Journal of the American Statistical Association*, **107**(497), 223–232. doi:10.1080/01621459.2011.645783.

Tarr G, Müller S, Welsh AH (2018). ***mplot**: Graphical Model Stability and Model Selection Procedures*. R package version 1.0.1, URL https://CRAN.R-project.org/package=mplot.

Tibshirani R (1996). “Regression Shrinkage and Selection via the Lasso.” *Journal of the Royal Statistical Society B*, **58**(1), 267–288.

Tibshirani RJ, Johnstone I, Hastie T, Efron B (2004). “Least Angle Regression.” *The Annals of Statistics*, **32**(2), 407–499. doi:10.1214/009053604000000067.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. 4th edition. Springer-Verlag, New York. doi:10.1007/978-0-387-21706-2.

Wickham H (2016). ***ggplot2**: Elegant Graphics for Data Analysis*. Springer-Verlag, New York. doi:10.1007/978-3-319-24277-4.

Affiliation:

Garth Tarr, Samuel Müller
University of Sydney
School of Mathematics and Statistics
University of Sydney
Sydney NSW 2006
E-mail: garth.tarr@sydney.edu.au, samuel.mueller@sydney.edu.au

Alan H. Welsh
Australian National University
Mathematical Sciences Institute
Australian National University
Canberra ACT 2601
E-mail: alan.welsh@anu.edu.au

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/

February 2018, Volume 83, Issue 9 *Submitted:* 2015-09-25
doi:10.18637/jss.v083.i09 *Accepted:* 2017-02-13