Quick Guide to the ‘bc’ Package

Jie Ding

February 12, 2019

Contents

Variable selection with unknown candidate subsets and high dimensionality 1

Variable selection with known candidate subsets 2

Model selection based on likelihood 2

This vignette gives a high level overview on how to use the ‘bc’ R package to perform variable selection in regression models, or general model selection based on likelihood. The main reference can be found here.

Variable selection with unknown candidate subsets and high dimensionality

Suppose that we are interested in selecting the most appropriate subset from a large volume of variables whose size may be even larger than sample size, and the noise variance is unknown. The following function can be used to select the most appropriate subset of variables. It returns indices of the selected variables (corresponding to each criterion used), and the parametricness index (PI) of the regression problem. The default PI (from 0 to 1) being closer to 1 indicates more parametricness and better interpretability.

```r
n <- 150
p <- 200
X <- genDesignMat(n,p,0.5)
beta <- rep(0, p)
beta[c(99,199)] = c(10, 5)
mu <- X %*% as.matrix(beta, ncol=1)
y <- mu + rnorm(n)
dat <- list(X = X, y= y)

res <- regBCVO(dat)
sprintf('Criterion: %s', paste(res$cri_name, collapse=" "))
sprintf('Selected subset: %s', paste(res$cri_selection, collapse=" "))
sprintf('Estimated coefficients: %s', paste(res$cri_coef, collapse=" "))
sprintf('PI: %s', paste(round(res$PI, 2), collapse=" "))
```

Users may also output the variable importance returned by this function:

```r
plot(res$var_imp, ylab = 'Variable importance', xlab = 'Variable index')
```

The following function applies the result on testing data, to obtain predicted values.

```
X_test <- genDesignMat(100,p,0.5)
y_pred <- predict(res, X_test)
```

In addition to the above use, one may change the suggested arguments, e.g.

```
res <- regBCVO(dat, hd_methods = c("MCP", "SCAD", "lasso"), dfmax = NULL, ratio = 0.7,
               weight_method = 'ARM', criteria = c('AIC', 'BC'), penaltyBC = NULL,
               adaptiveBC = FALSE, methodPI = 'BC', dat_test = NULL)
```

Variable selection with known candidate subsets

Suppose that we are interested in selecting the most appropriate subset from several candidate subsets in regression variable selection and the noise variance is unknown. The following function can be used to select the most appropriate subset of variables. It returns an index of the selected model (corresponding to each criterion used), and the parametricness index (PI) of the model class under consideration.

```
n <- 150
p <- 200
X <- genDesignMat(n,p,0.5)
beta <- rep(0, p)
beta[c(99,199)] = c(10, 5)
mu <- X %*% as.matrix(beta, ncol=1)
y <- mu + rnorm(n)
candidates <- vector('list', 0)
candidates[[1]] <- c(1, 99)
candidates[[2]] <- c(99, 199)
candidates[[3]] <- c(1, 99, 199)
dat <- list(X = X, y= y)
res <- varSelection(candidates, dat)
sprintf('Criterion: %s', paste(res$cri_name, collapse=" "))
sprintf('Selected model: %s', paste(res$cri_opt, collapse=" "))
sprintf('PI: %s', paste(round(res$PI, 2), collapse=" "))
```

In addition to the above use, one may change the suggested arguments, e.g.

```
varSelection(candidates, X, y, criteria=c('Cp, AIC, BC'), adaptiveBC=TRUE)
```

Here argument adaptiveBC appends a result using another version of BC based on adaptively chosen penalty parameter (instead of the suggested one).

Model selection based on likelihood

Suppose that the candidate models are parametric and the log-likelihood evaluated at MLE has been calculated for each model (in a vector loglik). Suppose that the dimensions of each model are given in a vector (dim). The following function can be used to select the most appropriate model. The following function returns an index of the selected model (corresponding to each criterion used), and the parametricness index (PI) of the model class under consideration.

```
res <- modelSelection(loglik = c(4.1, 5.2, 6.3), dim = c(1, 2, 3), n = 100,
                      criteria='AIC, BIC, BC', penaltyBC=NULL)
sprintf('Criterion: %s', paste(res$cri_name, collapse=" "))
sprintf('Selected model: %s', paste(res$cri_opt, collapse=" "))
sprintf('PI: %s', paste(round(res$PI, 2), collapse=" "))
```