Package CIpostSelect

Boubacar DIALLO

October 25, 2024

Abstract

During my internship, we focused on post-selection statistical inference, specifically the construction of confidence intervals after a model selection procedure. Traditional methods for calculating confidence intervals, which involve selecting the model on the full dataset and then computing intervals on the same sample, have certain limitations. Indeed, the study by Zhang et al. (2022) highlights that this approach does not ensure adequate empirical coverage probability. In some cases, it tends to produce overly narrow confidence intervals.

At one point during the internship, we compared our approach to the PoSI (Post-Selection Inference) method. However, PoSI generates highly conservative confidence intervals, ensuring that the true coefficients are contained within the intervals 100% of the time for nominal 95% intervals.

Our approach involved repeated data splitting, where the data were reshuffled at each iteration. This allowed us to generate matrices for both model selection and coefficient estimation. We then incorporated a voting system: one for the coefficients and another for the selected models.

Our results demonstrate that our confidence intervals are empirically validated through a simulation study. Additionally, this methodology achieves better performance in identifying the true model compared to selection methods such as Lasso and BIC.

1 Methodology

Our method relies on repeated data splitting with two types of voting, which we will present below.

1.1 Data splitting

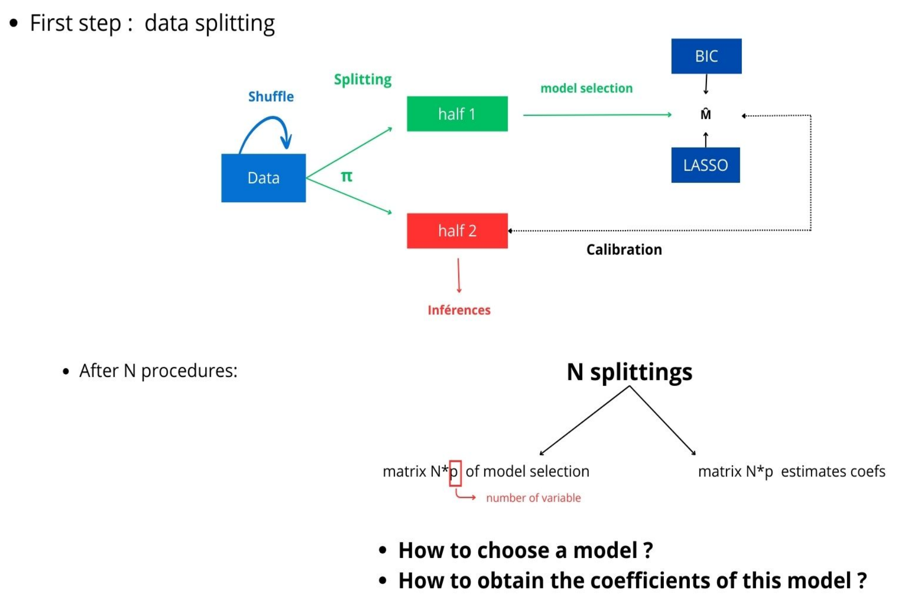

• First step : data splitting
Shuffle
Splitting
Data
π
half 1
half 2
model selection
BIC
M̂
LASSO
Calibration
Inférences
• After N procedures:
N splittings
matrix N*p of model selection
number of variable
matrix N*p estimates coefs
• How to choose a model ?
• How to obtain the coefficients of this model ?

Figure 1: **Data splitting strategy**

Given our data, for a certain number N of repetitions, we shuffle the data and split it into two parts with a probability π. On one part, we perform model selection $\hat{M}^{(k)}$, and on the other part, we calibrate the model to obtain coefficient estimates $\hat{\beta}^{(k)}$ with $k \in [1,\dots,N]$. As a result, we end up with an $N \times p$ matrix for model selections and an $N \times p$ matrix for the corresponding coefficient estimates for p explanatory variables.

The next question is how to choose a final model from this matrix of models. Once the model is selected, the following challenge is determining how to compute confidence intervals and obtain coefficient estimates corresponding to the chosen model.

1.2 Voting system

We will explore two types of voting: one on the coefficients and one on the models. Each of them has a specific way of obtaining the estimates.

1.2.1 Vote on models

For the voting on models, we proceed as follows:

$$\hat{M}_{\text{vote}} = \hat{M}^{(j)} \quad \text{s.t} \quad \forall i \neq j : \sharp\hat{M}^{(j)} > \sharp\hat{M}^{(i)}$$

We choose the model recorded during the N splittings that has been selected the most frequently. Next, we determine

$$A = \{k \in \{1,\dots,N\} : \hat{M}^{(k)} = \hat{M}_{\text{vote}}\}$$

Then, we compute the coefficient estimates as follows:

$$\hat{\beta}_j^{\hat{M}_{\text{vote}}} = \frac{1}{|A|}\sum_{k \in A} \hat{\beta}_j^{\hat{M}^{(k)}} \quad \forall j \in [1,\dots,p]$$

We will then search for the rows corresponding to the voted model in the coefficient estimates and average them across the columns to obtain the coefficient estimates, as illustrated in the following graphic :

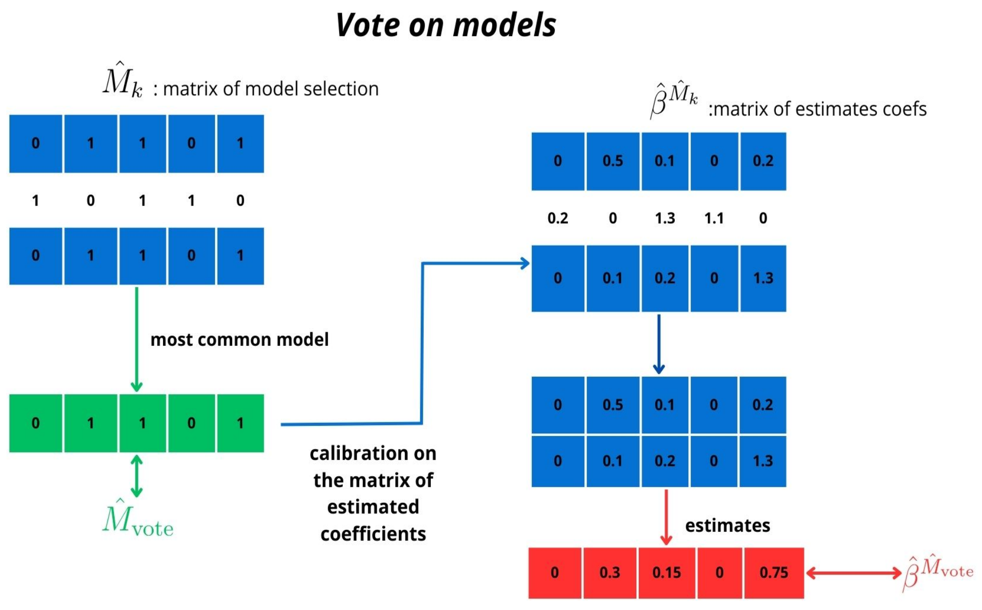


Figure 2: **Example of voting on models with $N = 3$ splittings and $p = 5$ explanatory variables**

1.2.2 Vote on the coefficients

For the voting on coefficients, we will proceed as follows:

We will start by selecting the model by checking for each variable whether it has been selected a certain percentage of the time, for example, 50%.

$$\hat{M}_{\text{vote}} = \left\{ \frac{1}{N} \sum_{i=1}^{N} \hat{M}_{ij} > s \right\}_{j \in [1,\ldots,p]}$$

where s is the threshold we set to decide whether or not to select the coefficient. Next, we will focus exclusively on the indices corresponding to the selected variables, which we will call index J:

$$J = \left\{ j \in P = \{1, \ldots, p\} \quad / \quad \hat{M}_{\text{vote}}^{P} = 1 \right\}$$

We will obtain the estimates by averaging the rows where the retained variables were selected:

$$(\hat{M}_k)_J = \begin{cases} 0_N & \text{if } j \notin J \\ \hat{M}_k^{(j)} & \text{if } j \in J \end{cases} \quad \forall j = 1, \ldots, p$$

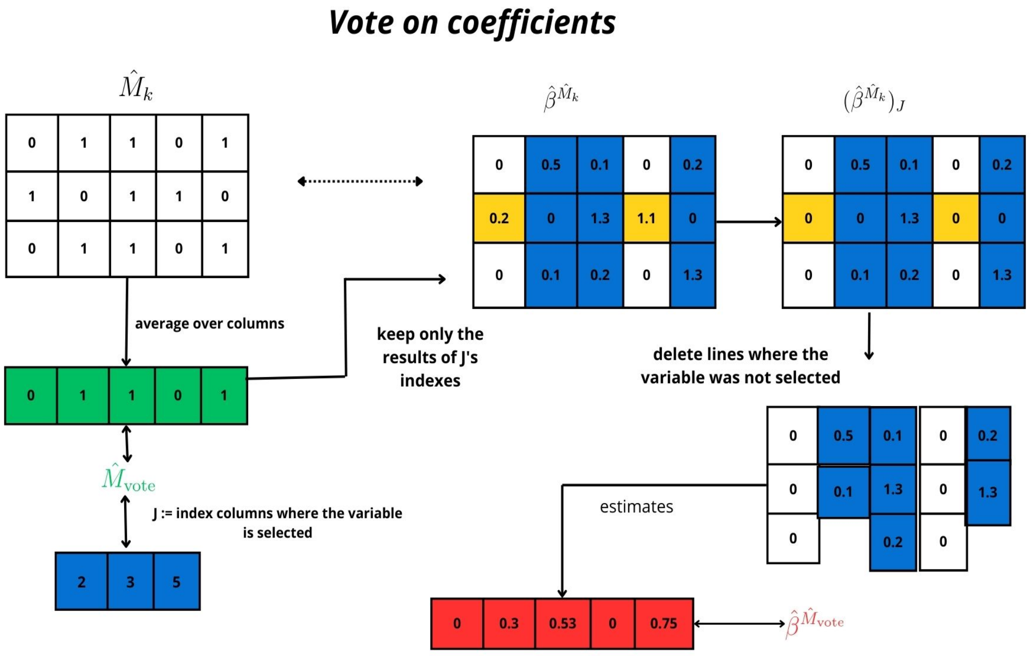
For all $j \in P$:

$$a_j = \left\{ k : (\hat{M}_k)_J^{(j)} = 1 \right\}$$

Then, we compute:

$$\hat{\beta}_j = \frac{1}{|a_j|} \sum_{a_j} \hat{\beta}_j^{\hat{M}_k}$$

All of this procedure is summarized in the example below:



Vote on coefficients
$\hat{M}_k$
$\hat{\beta}^{\hat{M}_k}$
$(\hat{\beta}^{\hat{M}_k})_J$
average over columns
keep only the results of J's indexes
delete lines where the variable was not selected
$\hat{M}_{\text{vote}}$
J := index columns where the variable is selected
estimates
$\hat{\beta}^{\hat{M}_{\text{vote}}}$

Figure 3: **Example of voting on the coefficients with $N = 3$ splittings and $p = 5$ explanatory variables**

1.3 Empirical method

I spoke only about coefficient estimates and not about calculating confidence intervals. It is important to note that at the end of the N splittings, with N large, we will have, as seen in the last motifs of our examples before averaging to obtain the coefficients, each column representing a vector of coefficient estimates for a given variable.

Thus, for the confidence intervals, we will then take the empirical quantile for each coefficient vector, as illustrated in the graphic below:

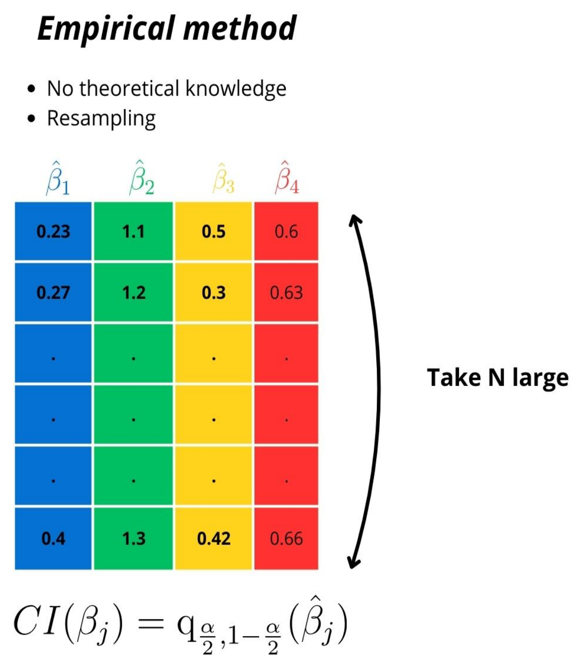


Figure 4: **Empirical method for confidence intervalle**

2 CIpostSelect

After this work on the methodology, we developed the ***CIpostSelect*** package, also known as *"Confidence Interval Post Selection of variables"*, which is available on CRAN.
This package features two main functions:

- `lmps` provides the model selection matrices from the various splittings of our data, as well as the matrix of coefficient estimates for the selected models during the different splittings.
- `CIps` takes an `lmps` object as a parameter, along with other parameters that specify the type of voting, for example, and the confidence level of our confidence intervals.

For more details, please refer to the documentation: `help("CIpostSelect")`

2.1 Example using a dataset in R

We will apply our package to the **BostonHousing** dataset.
The idea here is not to interpret the results but to show you how to apply the package. We will revisit the performance of the package a little later.

BostonHousing
The BostonHousing dataset, from the mlbench package in R, contains 506 observations and 14 variables that describe characteristics influencing house prices in Boston. The variables include socio-economic and environmental factors, such as the crime rate per capita, the number of rooms, the distance to employment centers, and the property tax rate. The target variable, `medv`, represents the median value of owner-occupied homes in thousands of dollars.

```r
library(CIpostSelect)
library(mlbench) # BostonHousing dataset
library(doParallel) # for parallelization
# Load the BostonHousing dataset
data("BostonHousing")
# Fit a linear model
model =  lm(medv ~ ., data = BostonHousing, method = "Lasso", N = 500,
     p_split = 0.5, cores = detectCores() - 2)
```

We recorded in the `model` the results of model selection as well as coefficient estimates from $N = 500$ splittings of our BostonHousing dataset with a splitting probability of $\pi = 0.5$ and a post-selection Lasso.

```
> summary(model)
Execution time in seconds:
[1] 10.5

Number of Splits
[1] 500

Frequency of variable selection :
  crim zn indus chas nox rm age dis rad tax ptratio    b lstat
Nb  193  57    43  303  124  500  15 183  14 165  500  379  500

The number of times the most frequent model was selected :
[1] 65
```

The results from calling the `summary` function are very important, as they will inform us later on which voting strategy is best suited for calculating confidence intervals.
In this example, we notice that the number of times we find the most frequent model is quite low compared to the number of divisions we performed. This is why we will opt for a vote based on the coefficients.

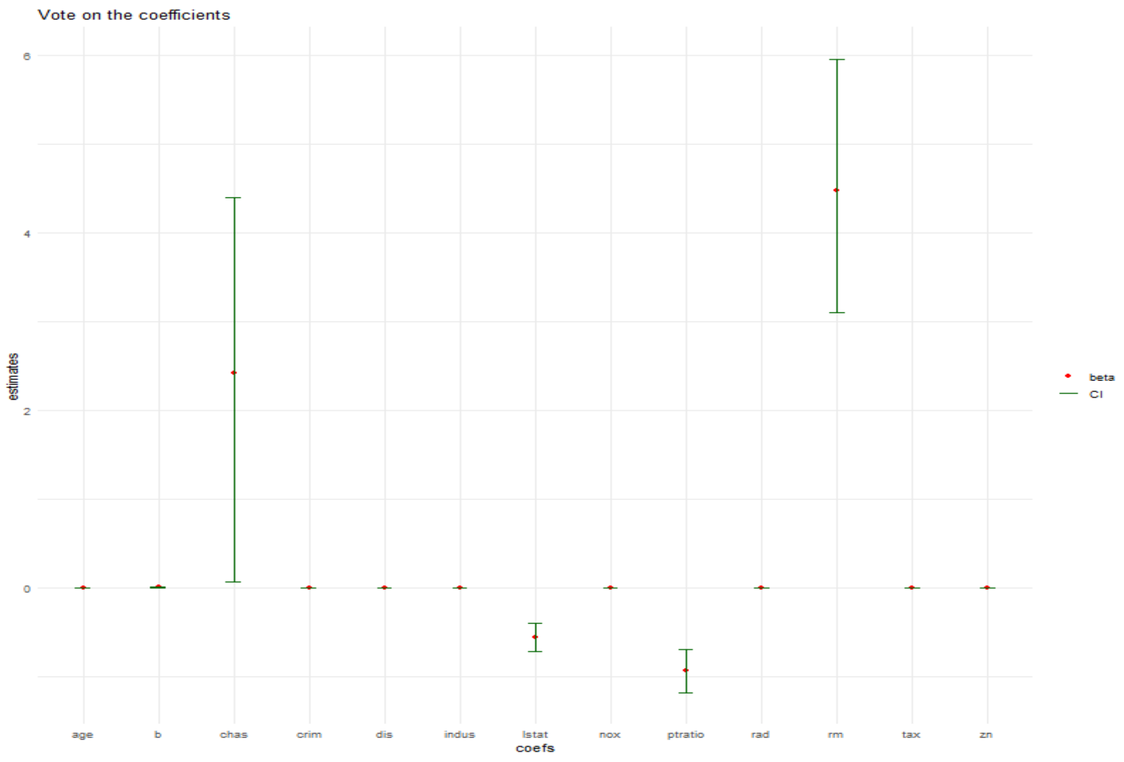
```r
cips = CIps(model, vote = "coef", alpha = 0.05, s.vote_coef = 0.5)
```

Thus, we decide to perform a vote on the coefficients for confidence intervals at 95% ($\alpha = 5\%$) with a selection threshold of $s = 0.5$.

```
> print(cips)
Variable selection :
           crim    zn  indus  chas   nox    rm   age   dis   rad   tax ptratio    b  lstat
selected? FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE

Estimates :
           chas        rm     ptratio         b       lstat
lower    0.0701    3.0192    -1.1788    0.0045    -0.7110
upper    4.2346    5.9552    -0.6922    0.0150    -0.3544
beta     2.4262    4.4975    -0.9247    0.0098    -0.5604
```

```
> plot(cips)
```



As I mentioned earlier, we will not focus on interpretations here; this was simply an example to explain how to use the `CIpostSelect` package to calculate post-selection confidence intervals for variable selection.
We will now discuss the performance of the package, particularly in comparison to classical methods.

3 Performance – Confidence Interval

In this section, we will study a simulation very similar to that of Zhang et al. (2022), which showed that the confidence intervals provided after a Lasso post-selection on the entire dataset tended to be very narrow and did not achieve their empirical coverage probability.
We will use this type of simulation to first compare the ***width of the confidence intervals*** of our different **methods**(classics versus CIpostSelect), and then examine their ***empirical coverage***.

Notation:

- ***"CIpostSelect"*** := will denote a BIC post-selection strategy
- ***"CIpostSelect²"*** := will denote a Lasso post-selection strategy

Simulation:

- $n = 60, \quad p = 18, \quad po = 5$
  We have 60 observations of 18 explanatory variables, of which 5 are relevant (true non-zero coefficients).
- $\beta_{j\in[po]} = 1.5$
  The relevant coefficients are all equal.
- $X \sim \mathcal{N}(0,1)$ and $\epsilon \sim \mathcal{N}(0,\sigma^2) \quad \sigma^2 = 2$

Methods:

- `Classical methods` (Lasso or BIC + OLS)
  Selection on the entire dataset and inference on the entire dataset while restricting to the selected subset.
- `CIpostSelect`

3.1 Width of Confidence Intervals

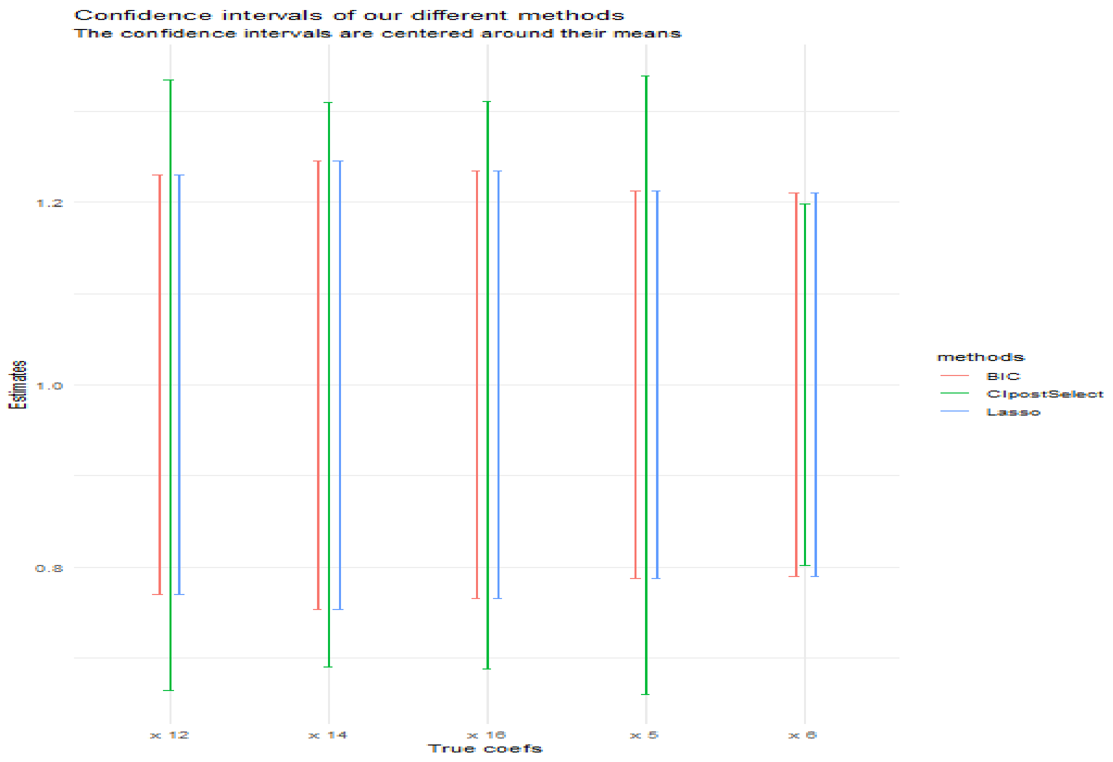
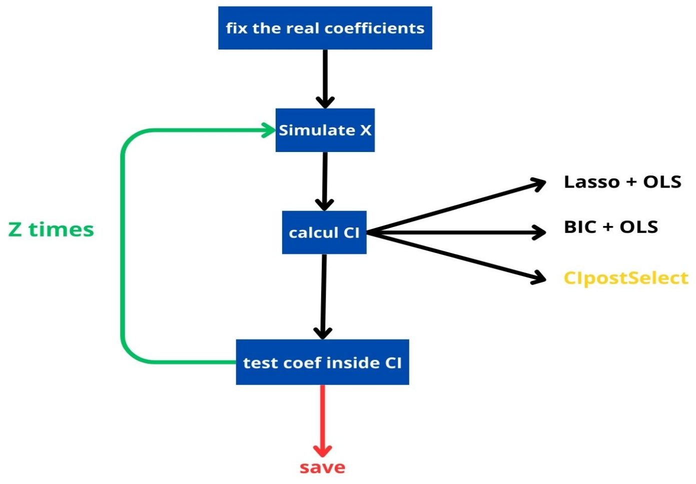


Figure 5: **Width of Confidence Intervals of Our Different Methods**:
We can notice that the confidence intervals provided by our method, `CIpostSelect`, shown in green, are much wider than those of classical methods.

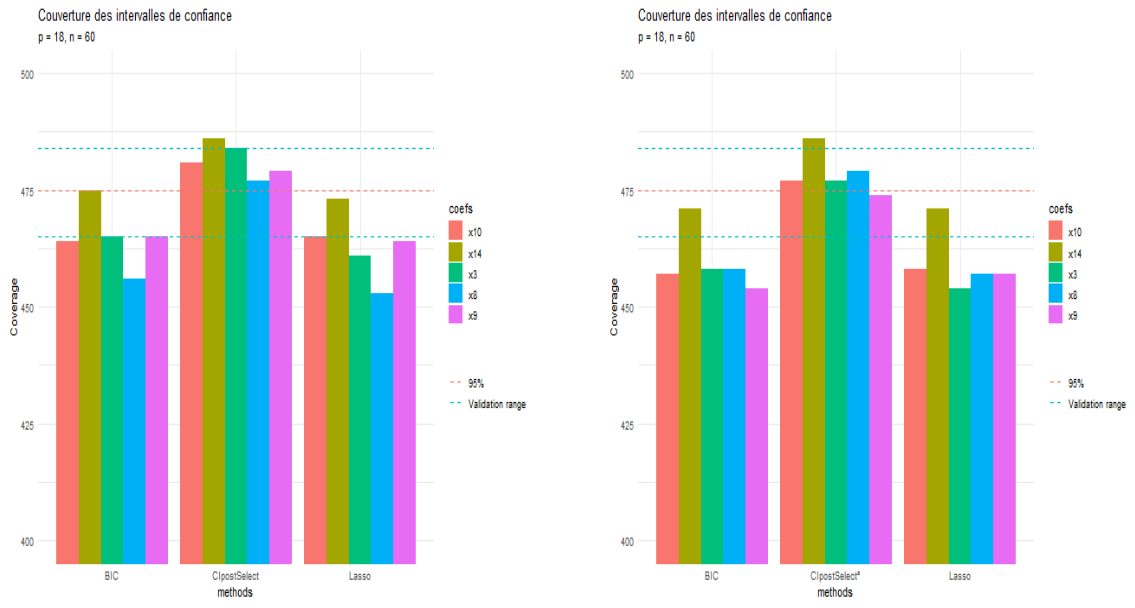
3.2 Empirical coverage

Now we will move on to the empirical coverage probability of the confidence intervals. We will repeat the calculation of confidence intervals a certain number of times, Z times, and at each repetition, we will test whether the true coefficients $\beta_{[j \in po]}$ are within the confidence intervals of the different methods.



We will also define a validation range for the 95% confidence intervals, which is given by:

$$[q_{\frac{\alpha}{2}}(\mathscr{B}(Z, 0.95)), \quad q_{1-\frac{\alpha}{2}}(\mathscr{B}(Z, 0.95)]$$



(a) BIC post-selection strategy for `CIpostSelect` (b) Lasso post-selection strategy for `CIpostSelect`

Figure 6: **Empirical coverage of our methods**:
We can notice that our method for calculating post-selection confidence intervals from our package is empirically validated, in stark contrast to classical methods.

4 Performance – Selection of the true model

This is not the primary goal of this project development. However, we observed that throughout the simulations and tests on the methodology, we were able to accurately identify the true model Mo. This prompted us to test the model selection performance of our package `CIpostSelect` compared to classical methods.

The strategy will be the same as that adopted for empirical coverage. The only difference here is that we will stop at model selection, and at this stage, we will test in each simulation whether the model we voted for using our various methods is the same as the fixed model Mo.

Independent Simulation

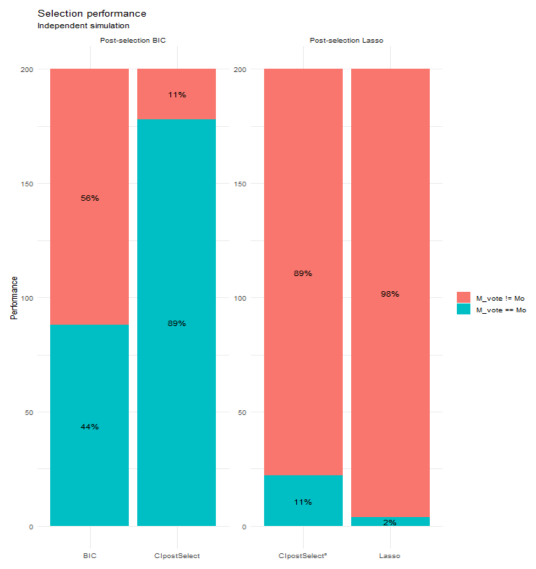
- $n = 300, \quad p = 50, \quad po = 15$
- $\beta_{[j \in po]} \sim \mathscr{U}[1,5]$
- $X \sim \mathscr{N}(0_p, I_p)$

Dependent Simulation

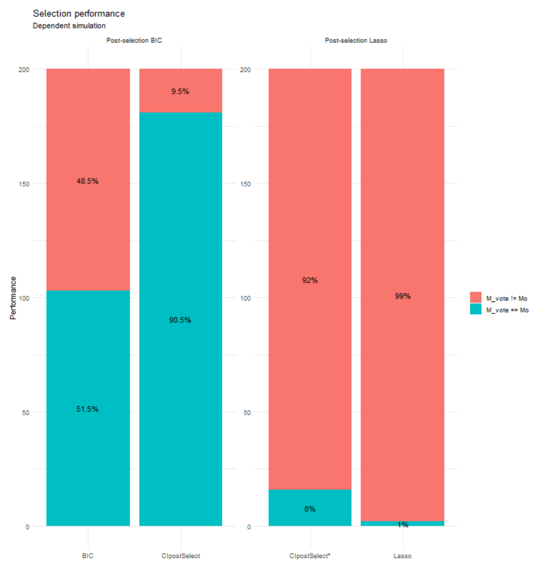
We have the same parameters as in the independent simulation, but this time we introduce dependence among the variables as follows:

- $X \sim \mathscr{N}(0_p, \Sigma)$
- $\Sigma = \begin{bmatrix} I_{20} & & \\ & I_{20} + \alpha A_{20} & \\ & & I_{10} + \gamma A_{10} \end{bmatrix} \in \mathbb{R}^{p \times p}$ with $\alpha = 0.1$ and $\gamma = 0.5$
  Thus, we have dependent data, and we ensured that among the relevant variables, there are independent ones, weakly dependent ones ($\alpha = 0.1$), and strongly dependent ones ($\gamma = 0.5$).

To test the selection performance for each of the two types of simulations, we will conduct $Z = 200$ simulations of our data, and the frequency of retrieving the true model is given by the results below:



(a) Independent Simulation



(b) Dependent Simulation

Figure 7: **Selection Performance of Our Methods on Two Types of Simulations**:
We can observe that our package provides a significant improvement in selecting the true model compared to classical methods.
For example, with dependent data using a BIC post-selection strategy, our package `CIpostSelect` retrieves the true model 90% of the time, while BIC alone only succeeds 51% of the time.

5 CIpostSelect-advanced: force a variable

In a project involving variable selection to address a specific issue, such as the ***impact of antibiotics on children's growth***, it may happen that the variable of interest (here, the use of antibiotics) is not selected by standard variable selection methods. This might suggest that, from the perspective of explaining the target (children's growth), the use of antibiotics is not as significant as other variables, like the child's age, birth weight, etc.

The question then arises: **how do we address the issue if the variable of interest is not selected?** One solution is to force the selection of this variable. In our case, this means forcing the constant selection of the *use of antibiotics* variable during the N splits.

What remains to be determined is whether the resulting inferences are valid. To do this, we will force, in two types of simulations (independent and dependent, as before), the selection of variable "X1" under the following conditions:

- $\beta_1 = 0.2$
- $\beta_{[j\neq 1\epsilon po]} \sim \mathscr{U}[1,5]$

Thus, this variable "$X1$" is less influential than the others, making it more difficult to select under normal conditions. We will then observe the resulting outcomes.

To force a variable, you need to specify the name of the variable you want to force when calling the `lmps` function, as shown below:

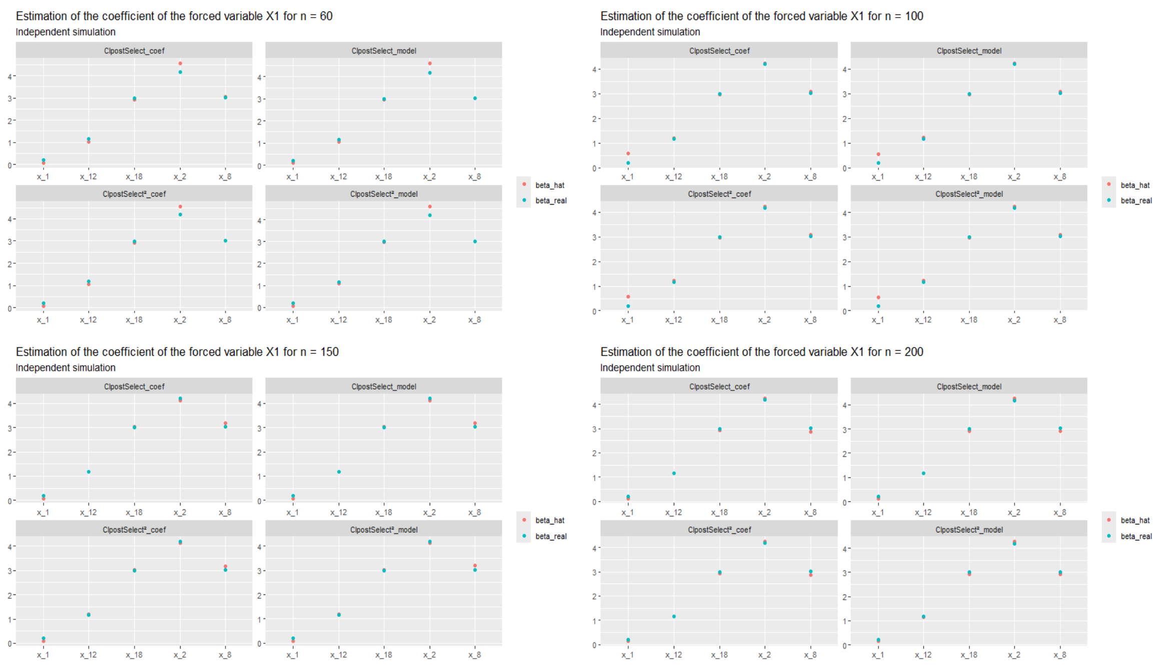
```
model = lmps(y ~ ., data = data, method = "BIC", N = 1000, p_split = 0.5, cores =
    detectCores() - 2, forced_var = "X1")
```

Independent Simulation:

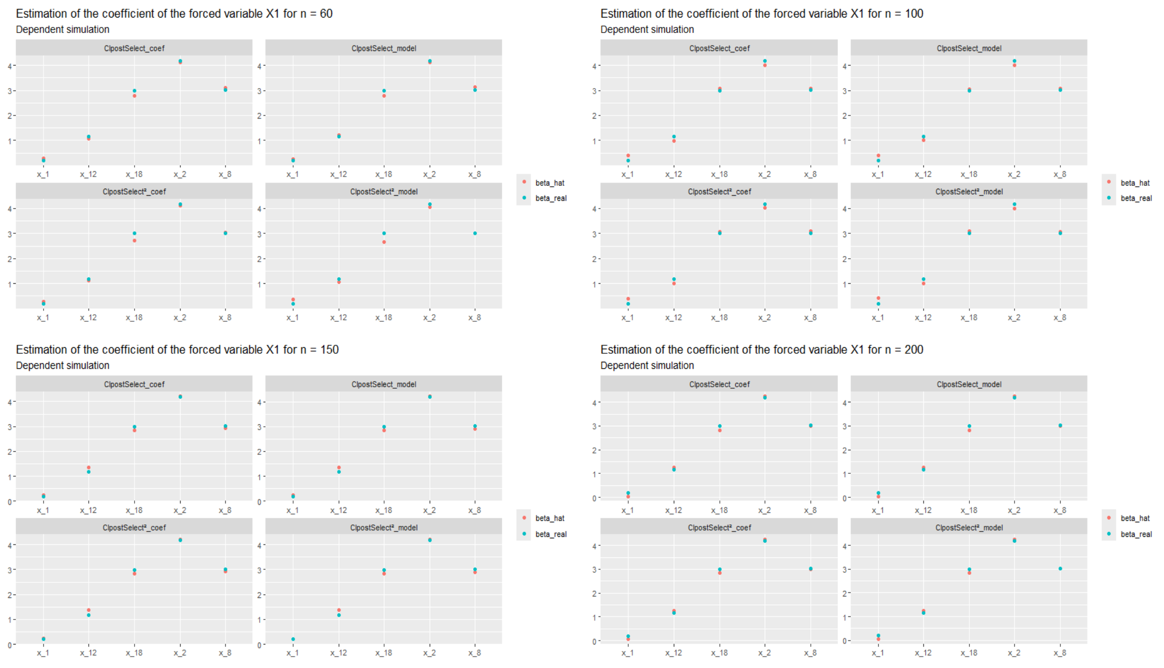
- **Number of variables (p)**: 18, **Number of relevant variables (po)**: 5
- $\beta_1 = 0.2$ and $\beta_{[2,po]} \sim \mathscr{U}[1,5]$
  Simulate the 4 relevant variables among the 5 uniformly from [1, 5], with the coefficient of the forced variable $X1$ set to 0.2.
- $X \sim \mathcal{N}(0,1)$ and $\epsilon \sim \mathcal{N}(0,2)$
- Sample sizes n in [60, 100, 150, 200]

For each of these simulations, we will run **CIpostSelect** and **CIpostSelect²** with each of their two votes to compare the estimates of the forced variable.

Dependent Simulation: The same approach will be used, but we will add dependence between the variables as described in **section 4**.



(a) Independent Simulation



(b) Dependent Simulation

Figure 8: **Estimation of the coefficient of the forced variable X1**
We can observe that the estimated coefficients for the variable $X1$ across different methods and simulations are close to the true value of the actual coefficient. This gives us confidence in using this option when the variable of interest is not selected and we want to interpret it.