

# Package ‘BMSC’

April 16, 2019

**Title** Bayesian Model Selection under Constraints

**Version** 0.2.0

**Description** A Bayesian regression package supporting constrained coefficient estimation and variable selection using Stan. This includes a robust variable selection algorithm by a horseshoe prior (<doi:10.1093/biomet/asq017>) that finds the optimal model considering main effects, interactions as well as powers of given variables under potential parameter constraints.

**Depends** R (>= 3.4.0), Rcpp (>= 0.12.0), methods

**Imports** dplyr (>= 0.7.4), ggplot2 (>= 2.2.1), loo (>= 2.0.0), rstan (>= 2.18.1), rstantools (>= 1.5.1), R.utils (>= 2.6.0)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**LinkingTo** StanHeaders (>= 2.18.0), rstan (>= 2.18.2), BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0)

**Suggests** lintr, testthat

**RoxygenNote** 6.1.0

**NeedsCompilation** yes

**Author** Marcus Groß [aut, cre],  
Ricardo Fernandes [aut],  
Mira Celine Klein [ctb]

**Maintainer** Marcus Groß <marcus.gross@inwt-statistics.de>

**Repository** CRAN

**Date/Publication** 2019-04-16 15:25:42 UTC

## R topics documented:

addInteractionToVars . . . . .	2
addPowToVars . . . . .	3
ConstrainedLinReg-class . . . . .	3
constrSelEst . . . . .	4

createFormula . . . . .	5
createFormulaInternal . . . . .	6
extractVarname . . . . .	6
getBestModel . . . . .	7
getBetaMatrix . . . . .	7
handleMissingData . . . . .	8
makeInteractions . . . . .	8
makePoly . . . . .	9
plotModelFit . . . . .	10
plotModels . . . . .	10
predict,ConstrainedLinReg-method . . . . .	11
prepColorVec . . . . .	11
prepDatForPredict . . . . .	12
prepModelNames . . . . .	12
prepPlotData . . . . .	13
print.ConstrainedLinReg . . . . .	13
show,ConstrainedLinReg-method . . . . .	14
sortAndPaste . . . . .	14
tryAsFormula . . . . .	15

## Index 16

---

addInteractionToVars *Add interactions of a specific order to a vector of variables*

---

### Description

Add interactions of a specific order to a vector of variables

### Usage

```
addInteractionToVars(order, vars)
```

### Arguments

order	integer: order of the interaction
vars	character: variables

### Details

Interactions of variables with themselves (including polynomials of themselves) are not included.

### Value

Character vector

### Examples

```
BMSC::addInteractionToVars(3, c("x1", "x2", "x3"))
```

---

addPowToVars	<i>Add exponent to a vector of variables</i>
--------------	--

---

**Description**

Remark: Since this function is to be used only within `createFormula`, the validity of the input is not checked here but in `createFormula`.

**Usage**

```
addPowToVars(vars, power)
```

**Arguments**

vars	character: variable names
power	integer: exponent

**Value**

Vector of same length as vars

**Examples**

```
BMSC:::addPowToVars(c("x1", "x2"), 2)
```

---

ConstrainedLinReg-class	<i>S4 class for constrained linear regression models</i>
-------------------------	--

---

**Description**

Inherits from `stanfit`

**Slots**

formula	model formula (class formula)
hasIntercept	logical: Does the model formula include an intercept?
scaleCenter	numeric: location scale of betas
scaleScale	numeric: scale scale of betas

---

 constrSelEst

*Model selection algorithm for constrained estimation*


---

### Description

Model selection algorithm for constrained estimation

### Usage

```
constrSelEst(formula, data, mustInclude = "", maxExponent = 1,
  interactionDepth = 1, intercept = TRUE, constraint_1 = FALSE,
  yUncertainty = rep(0, nrow(data)), xUncertainty = NULL,
  maxNumTerms = 10, scale = FALSE, chains = 4, iterations = 2000)
```

### Arguments

formula	formula object: formula object without exponents or interactions. If formula is not of class formula, it is turned into one.
data	data.frame: dataset
mustInclude	character vector: variables to include in any case; use ":" for interactions and "I(.)" for powers, e.g.: "I(x1^2):I(x2^3)".
maxExponent	positive integer: highest exponent included in the formula. Default is 1, e.g., only linear effects.
interactionDepth	positive integer: maximum order of interaction. Default is 1, e.g., only main effects (no interactions).
intercept	logical: Should the intercept be included in the estimation or not?
constraint_1	logical: Should the all beta variables add up to 1?
yUncertainty	numeric vector: optional, uncertainties in y variable given in standard deviations
xUncertainty	data.frame: optional, uncertainties in x variables. variable names must match with names in formula
maxNumTerms	positive integer: maximum number of variables to include
scale	logical: should the variables be scaled to mean 0 and sd 1?
chains	positive integer: number of chains for MCMC sampling
iterations	positive integer: number of iterations per chain for MCMC sampling

### Value

A list of potential models

**Examples**

```
## Not run:
suppressWarnings(RNGversion("3.5.0"))
set.seed(44)
n <- 80
x1 <- rnorm(n, sd = 1)
x2 <- rnorm(n, sd = 1)
x3 <- rnorm(n, sd = 1)
y <- 0.4 + 0.3 * x1 + 0.3 * x1 * x3 + 0.4 * x1 ^ 2 * x2 ^ 3 + rnorm(n, sd = 0.3)
yUncertainty <- rexp(n, 10) * 0.01
#optional (slow)
#xUncertainty <- data.frame(x3 = rep(0.1, n), x1 = rep(0.1, n), x2 = rep(1, n))
data <- data.frame(x1, x2, x3, y, yUncertainty)
models <- constrSelEst(y ~ x1 + x2 + x3, mustInclude = "x1", maxExponent = 3,
                      interactionDepth = 3, intercept = TRUE,
                      constraint_1 = TRUE, data = data,
                      yUncertainty = yUncertainty,
                      xUncertainty = NULL,
                      maxNumTerms = 10)

plotModelFit(models)
bestModel <- getBestModel(models, thresholdSE = 2)
print(bestModel)

## End(Not run)
```

---

createFormula	<i>Create a formula with interactions and polynomials up to a desired order</i>
---------------	---

---

**Description**

Creates a formula with interactions and polynomials up to a desired order. If the input formula already includes interactions, exponents or other functions (e.g., [sqrt](#)), they are ignored.

**Usage**

```
createFormula(formula, maxExponent = 1, interactionDepth = 1,
              intercept = TRUE)
```

**Arguments**

formula	formula object: formula object without exponents or interactions. If formula is not of class formula, it is turned into one.
maxExponent	positive integer: highest exponent included in the formula. Default is 1, e.g., only linear effects.
interactionDepth	positive integer: maximum order of interaction. Default is 1, e.g., only main effects (no interactions).
intercept	logical: include intercept or not?

**Value**

A formula containing the original independent variables and their polynomials and interactions.

**Examples**

```
createFormula("y ~ x1 + x2", 2, 3)
createFormula(as.formula("y ~ x1 + x2"), interactionDepth = 2)

carFormula <- createFormula("mpg ~ cyl + disp + drat", 2, 3)
summary(lm(carFormula, mtcars))
```

---

`createFormulaInternal` *Create formula with interactions and polynomials if all checks in `createFormula` have passed*

---

**Description**

Create formula with interactions and polynomials if all checks in `createFormula` have passed

**Usage**

```
createFormulaInternal(formula, allVars, maxExponent, interactionDepth,
  intercept)
```

**Arguments**

<code>formula</code>	formula object
<code>allVars</code>	object returned by <code>all.vars</code>
<code>maxExponent</code>	positive integer
<code>interactionDepth</code>	positive integer
<code>intercept</code>	boolean

---

`extractVarname` *Extract variable name from polynomial expression*

---

**Description**

Extract variable name from polynomial expression

**Usage**

```
extractVarname(x)
```

**Arguments**

x                    Character: variables

**Examples**

```
BMSC:::extractVarname(c("x1",
  "I(x2^2)"))
```

---

getBestModel	<i>Get Best Model after Models Selection</i>
--------------	--

---

**Description**

Get Best Model after Models Selection

**Usage**

```
getBestModel(models, thresholdSE = 1, plotModels = TRUE)
```

**Arguments**

models                list of models fitted by [constrSelEst](#) function

thresholdSE          numeric: How much standard errors in leave-one-out prediction performance can the sparse model be worse than the best model

plotModels          boolean: Plot models in leave-one-out evaluation plot TRUE/FALSE

**Value**

The best sparse model concerning leave-one-out performance within a threshold

---

getBetaMatrix	<i>Extract beta matrix from <a href="#">ConstrainedLinReg</a> model</i>
---------------	---

---

**Description**

Extracts matrix of beta estimates

**Usage**

```
getBetaMatrix(model, hasIntercept)
```

**Arguments**

model                model object: Model of class [ConstrainedLinReg](#)

hasIntercept        logical: Does the model formula include an intercept?

**Value**

matrix of estimates

---

handleMissingData	<i>Exclude rows with missing values</i>
-------------------	---

---

**Description**

All rows with missing values on the variables from the model formula are excluded. If all rows are excluded, an error occurs. If only some of the rows are excluded, the number and percentage of excluded rows is printed via a message. In addition, the corresponding positions from the yUncertainty vector are excluded.

**Usage**

```
handleMissingData(data, formula, yUncertainty)
```

**Arguments**

data	data.frame
formula	formula object
yUncertainty	numeric: vector

**Value**

A list with the elements "data" (data frame containing only the relevant variables and complete rows) and "yUncertainty".

---

makeInteractions	<i>Add all interactions up to a desired order</i>
------------------	---

---

**Description**

Add all interactions up to a desired order

**Usage**

```
makeInteractions(vars, interactionDepth)
```

**Arguments**

vars	character: variable names (potentially including polynomial expressions)
interactionDepth	integer: highest interaction order



**Details**

Interactions of variables with themselves (including polynomials of themselves) are not included.

**Value**

Character vector

**Examples**

```
BMSC::makeInteractions(vars = c("x1", "x2",  
"I(x1^2)", "I(x2^2)"), interactionDepth = 3)
```

---

makePoly

*Create polynomial of degree maxExponent from variable names*

---

**Description**

Remark: Since this function is to be used only within [createFormula](#), the validity of the input is not checked here but in [createFormula](#).

**Usage**

```
makePoly(vars, maxExponent)
```

**Arguments**

vars	character: variable names
maxExponent	integer: highest exponent

**Value**

Character vector of length(vars) times maxExponent

**Examples**

```
BMSC::makePoly(vars = c("x1", "x2"), maxExponent = 3)
```

---

plotModelFit *Plot errors of all models*

---

### Description

This plot is automatically produced with the execution of `getBestModel`.

### Usage

```
plotModelFit(models, thresholdSE = 1, loos = NULL,
             markBestModel = TRUE)
```

### Arguments

models	List with models of class <code>ConstrainedLinReg</code>
thresholdSE	numeric: Factor multiplied with standard error to obtain ends of error bars
loos	List with the model fit results for all models as returned by <code>BMSC:::getLoo</code> . If not provided, they are computed from the model list, which can take some time.
markBestModel	boolean: highlight position of the best model in the model list

---

plotModels *Plot model errors with errorbars*

---

### Description

Plot model errors with errorbars

### Usage

```
plotModels(datPlot, colours, thresholdSE)
```

### Arguments

datPlot	data.frame with prepared plot data
colours	character: colour(s) for the points, bars and x-axis labels
thresholdSE	numeric: Factor multiplied with standard error to obtain ends of error bars

---

predict, ConstrainedLinReg-method  
*Compute predictions from constraint estimation model*

---

**Description**

Computes prediction from model of class `ConstrainedLinReg` and a data.frame.

**Usage**

```
## S4 method for signature 'ConstrainedLinReg'
predict(object, newdata)
```

**Arguments**

object	Model of class <code>ConstrainedLinReg</code>
newdata	data.frame containing all variables that appear in the model formula

**Value**

Numeric vector of predictions. For observations with missing values on the explanatory variables, a prediction of NA is returned.

---

prepColorVec            *Prepare colour vector*

---

**Description**

Prepare colour vector

**Usage**

```
prepColorVec(posBestModel, length)
```

**Arguments**

posBestModel	numeric: position of best Model
length	numeric: Length of colour vector

**Value**

Vector of length length. It contains "black" expect for the position provided in posBestModel, which is "chartreuse4" (green)

---

```
prepDatForPredict
```

*Exclude rows with missing data on predictor variables*

---

### Description

Rows with missing values on predictor variables are excluded. An unused column for the dependent variable is added to avoid errors.

### Usage

```
prepDatForPredict(formula, newdata)
```

### Arguments

formula	Model formula
newdata	data.frame containing all variables that appear in the model

### Details

A column of ones for the dependent variable is added. Otherwise `model.matrix` tries to take it from the formula's environment, which is the original data. This usually results in an error due to unequal variable length. This column is however not used.

### Value

Object of class `na.exclude`

---

```
prepModelNames
```

*Extract model names from model objects*

---

### Description

Extracts the model formulae from a list of model objects of class `ConstrainedLinReg`. Elements that are superfluous for reading (e.g., brackets) are removed.

### Usage

```
prepModelNames(models)
```

### Arguments

models	List with models of class <code>ConstrainedLinReg</code>
--------	--

---

```
prepPlotData          Prepare data to plot model fit
```

---

**Description**

Prepare data to plot model fit

**Usage**

```
prepPlotData(loos, modelNames, thresholdSE)
```

**Arguments**

loos	List with the model fit results for all models as returned by <code>BMSC::getLoo</code> . If not provided, they are computed from the model list, which can take some time.
modelNames	Names for the models in the same order as they appear in <code>loos</code>
thresholdSE	numeric: Factor multiplied with standard error to obtain ends of error bars

**Value**

A data.frame with the columns Estimate (Estimate of the looic), SE, model, lower, and upper

---

```
print.ConstrainedLinReg
          Print constraint estimation model
```

---

**Description**

Print constraint estimation model

**Usage**

```
## S3 method for class 'ConstrainedLinReg'
print(x, ...)
```

**Arguments**

x	model object of class <code>ConstrainedLinReg</code>
...	arguments passed from or to other methods

---

show, ConstrainedLinReg-method  
*Print constraint estimation model*

---

### Description

Prints the model formula and estimates as well as sigma with the corresponding 95

### Usage

```
## S4 method for signature 'ConstrainedLinReg'  
show(object)
```

### Arguments

object            Model of class "MPIconstraintModel"

---

sortAndPaste            *Sort a vector and collapse elements together using ":"*

---

### Description

Sort a vector and collapse elements together using ":"

### Usage

```
sortAndPaste(x)
```

### Arguments

x                    Vector

### Examples

```
BMSC:::sortAndPaste(c("var1", "var2"))
```

---

tryAsFormula	<i>Turn character vector into formula, return error if not possible</i>
--------------	---

---

**Description**

Turn character vector into formula, return error if not possible

**Usage**

```
tryAsFormula(input)
```

**Arguments**

input	character
-------	-----------

**Value**

Formula or error

# Index

addInteractionToVars, 2  
addPowToVars, 3  
all.vars, 6

ConstrainedLinReg, 7, 10–13  
ConstrainedLinReg  
    (ConstrainedLinReg-class), 3  
ConstrainedLinReg-class, 3  
constrSelEst, 4, 7  
createFormula, 3, 5, 6, 9  
createFormulaInternal, 6

extractVarname, 6

getBestModel, 7, 10  
getBetaMatrix, 7

handleMissingData, 8

makeInteractions, 8  
makePoly, 9  
model.matrix, 12

na.exclude, 12

plotModelFit, 10  
plotModels, 10  
predict, ConstrainedLinReg-method, 11  
prepColorVec, 11  
prepDatForPredict, 12  
prepModelNames, 12  
prepPlotData, 13  
print.ConstrainedLinReg, 13

show, ConstrainedLinReg-method, 14  
sortAndPaste, 14  
sqrt, 5  
stanfit, 3

tryAsFormula, 15