

# Package ‘rFSA’

May 9, 2026

**Type** Package

**Title** Feasible Solution Algorithm for Finding Best Subsets and Interactions

**Version** 0.9.6

**Date** 2020-06-10

**Description**

Assists in statistical model building to find optimal and semi-optimal higher order interactions and best subsets. Uses the `lm()`, `glm()`, and other R functions to fit models generated from a feasible solution algorithm. Discussed in Subset Selection in Regression, A Miller (2002). Applied and explained for least median of squares in Hawkins (1993) <[doi:10.1016/0167-9473\(93\)90246-P](https://doi.org/10.1016/0167-9473(93)90246-P)>. The feasible solution algorithm comes up with model forms of a specific type that can have fixed variables, higher order interactions and their lower order terms.

**License** GPL-2

**LazyData** TRUE

**Imports** parallel, methods, tibble, rPref, tidy, hash

**RoxygenNote** 7.1.0

**NeedsCompilation** no

**Author** Joshua Lambert [aut, cre],  
Liyu Gong [aut],  
Corrine Elliott [aut],  
Sarah Janse [ctb]

**Maintainer** Joshua Lambert <[joshua.lambert@uc.edu](mailto:joshua.lambert@uc.edu)>

**Repository** CRAN

**Date/Publication** 2020-06-10 20:20:03 UTC

## Contents

<code>adj.r.squared</code> . . . . .	2
<code>apress</code> . . . . .	3

bdist . . . . .	3
fitmodels . . . . .	4
fitted.FSA . . . . .	4
FSA . . . . .	5
int.p.val . . . . .	7
list.criterion . . . . .	7
max_abs_resid . . . . .	8
nextswap . . . . .	8
pFSA . . . . .	9
plot.FSA . . . . .	11
predict.FSA . . . . .	11
print.FSA . . . . .	12
QICu.geeglm . . . . .	13
r.squared . . . . .	13
rmse . . . . .	13
summary.FSA . . . . .	14
swaps . . . . .	14
twFSA . . . . .	15
which.max.na . . . . .	16
which.min.na . . . . .	17
<b>Index</b>	<b>18</b>

---

adj.r.squared	<i>An rFSA Criterion Function.</i>
---------------	------------------------------------

---

### Description

rFSA Criterion Function to compute Adjusted R-Squared.

### Usage

```
adj.r.squared(model, name = "Adj R Squared")
```

### Arguments

model	lm or glm fit to be passed.
name	passed to print.FSA

---

apress	<i>An rFSA Criterion Function.</i>
--------	------------------------------------

---

**Description**

rFSA Criterion Function to Allen's Press Statistic.

**Usage**

```
apress(model, name = "PRESS")
```

**Arguments**

model	lm or glm fit to be passed.
name	passed to print.FSA

---

bdist	<i>An rFSA Criterion Function.</i>
-------	------------------------------------

---

**Description**

rFSA Criterion Function to compute the Bhattacharyya distance.

**Usage**

```
bdist(model, name = "B Distance")
```

**Arguments**

model	lm or glm fit to be passed.
name	passed to print.FSA

**Examples**

```
#To use Bhattacharyya Distance and FSA the response must be binary, and you must
#be considering searching for two way continuous interactions.
data(mtcars)
fit<-FSA(formula = "am~gear*hp",data = mtcars,
fitfunc = glm,family="binomial",m = 2,cores=1,
interactions = TRUE,criterion = bdist,minmax = "max")
```

---

fitmodels	<i>Model fitting function for FSA solutions</i>
-----------	---

---

**Description**

Model fitting function for FSA solutions

**Usage**

```
fitmodels(object, ...)
```

**Arguments**

object	FSA object to construct models on.
...	other parameters passed to lm or glm. See help(lm) or help(glm) for other potential arguments.

**Value**

list of FSA models that have been fitted.

**Examples**

```
#use mtcars package see help(mtcars)
data(mtcars)
colnames(mtcars)
fit<-lmFSA(formula="mpg~hp*wt", data=mtcars, fixvar="hp",
            quad=FALSE, m=2, numrs=10, save_solutions=FALSE, cores=1)
fitmodels(fit)
```

---

fitted.FSA	<i>Fitted Values for FSA solutions</i>
------------	--

---

**Description**

Fitted Values for FSA solutions

**Usage**

```
## S3 method for class 'FSA'
fitted(object, ...)
```

**Arguments**

object	FSA object to get fitted values from.
...	other parameters passed to fitmodels or fitted function. See help(fitmodels) or help(fitted) for assistance.

**Value**

list of fitted values from each FSA model.

**Examples**

```
#use mtcars package see help(mtcars)
data(mtcars)
colnames(mtcars)
fit<-lmFSA(formula="mpg~hp*wt", data=mtcars, fixvar="hp",
            quad=FALSE, m=2, numrs=10, save_solutions=FALSE, cores=1)
fitted(fit)
```

---

FSA

*FSA: Feasible Solution Algorithm*

---

**Description**

A function using a Feasible Solution Algorithm to find a set of feasible solutions for a statistical model of a specific form that could include  $m$ th-order interactions (Note that these solutions are optimal in the sense that no one swap to any of the variables will increase the criterion function.)

**Usage**

```
FSA(
  formula,
  data,
  fitfunc = lm,
  fixvar = NULL,
  quad = FALSE,
  m = 2,
  numrs = 1,
  cores = 1,
  interactions = T,
  criterion = AIC,
  minmax = "min",
  checkfeas = NULL,
  var4int = NULL,
  min.nonmissing = 1,
  return.models = FALSE,
  fix.formula = NULL,
  ...
)

lmFSA(...)

glmFSA(...)
```

**Arguments**

<code>formula</code>	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
<code>data</code>	a data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model.
<code>fitfunc</code>	the method that should be used to fit the model. For Example: <code>lm</code> , <code>glm</code> , or other methods that rely on <code>formula</code> , <code>data</code> , and other inputs.
<code>fixvar</code>	variable(s) to fix in the model. Usually a covariate that should always be included (Example: Age, Sex). Will still consider it with interactions. Default is <code>NULL</code> .
<code>quad</code>	Include quadratic terms or not. Logical.
<code>m</code>	order of terms to include. If <code>interactions</code> is set to <code>TRUE</code> then <code>m</code> is the order of interactions to be considered. For Subset selection ( <code>interaction=F</code> ), <code>m</code> is the size of the subset to examine. Defaults to 2.
<code>numrs</code>	number of random starts to perform.
<code>cores</code>	number of cores to use while running. Note: Windows can only use 1 core. See <code>mclapply</code> for details. If function detects a Windows user it will automatically set <code>cores=1</code> .
<code>interactions</code>	whether to include interactions in model. Defaults to <code>TRUE</code> .
<code>criterion</code>	which criterion function to either maximize or minimize. For linear models one can use: <code>r.squared</code> , <code>adj.r.squared</code> , <code>cv5.lmFSA</code> (5 Fold Cross Validation error), <code>cv10.lmFSA</code> (10 Fold Cross Validation error), <code>apress</code> (Allen's Press Statistic), <code>int.p.val</code> (Interaction P-value), <code>AIC</code> , <code>BIC</code> .
<code>minmax</code>	whether to minimize or maximize the criterion function
<code>checkfeas</code>	vector of variables that could be a feasible solution. These variables will be used as the last random start.
<code>var4int</code>	specification of which variables to check for marginal feasibility. Default is <code>NULL</code>
<code>min.nonmissing</code>	the combination of predictors will be ignored unless this many of observations are not missing
<code>return.models</code>	bool value to specify whether return all the fitted models which have been checked
<code>fix.formula</code>	...
...	other arguments passed to <code>fitfunc</code> .

**Value**

matrix of results

**Functions**

- `FSA`: find best set of variables for statistical models
- `lmFSA`: alias for `FSA(fitfunc=lm, ...)`
- `glmFSA`: alias for `FSA(fitfunc=glm, ...)`

**Examples**

```

N <- 10 #number of obs
P <- 100 #number of variables
data <- data.frame(matrix(rnorm(N*(P+1)), nrow = N, ncol = P+1))

sln <- FSA(formula = "X101~1", data = data, cores = 1, m = 2,
interactions = FALSE, criterion = AIC, minmax = "min",
numrs = 10)
sln

```

---

int.p.val                      *An rFSA Criterion Function.*

---

**Description**

rFSA Criterion Function to compute Likelihood Ratio Test Statistics p-value for the largest order interaction term.

**Usage**

```
int.p.val(model, name = "Interaction P-Value")
```

**Arguments**

model	lm or glm fit to be passed.
name	passed to print.FSA

---

list.criterion                      *List all included Criteria function for lmFSA and glmFSA.*

---

**Description**

List all included Criteria function for lmFSA and glmFSA.

**Usage**

```
list.criterion()
```

**Value**

list of functions and whether lmFSA or glmFSA work with those functions.

**Examples**

```
list.criterion()
```

---

max_abs_resid	<i>Return maximum absolute residual from a model</i>
---------------	--

---

**Description**

Return maximum absolute residual from a model

**Usage**

```
max_abs_resid(model)
```

**Arguments**

model	model obj
-------	-----------

---

nextswap	<i>Variables to include in the &gt;1st step of an mth order interaction model determined from the Feasible Soution Alorithm.</i>
----------	--

---

**Description**

Finds the swaps available given a current position given previous picks.

**Usage**

```
nextswap(curpos, n, prevpos, quad)
```

**Arguments**

curpos	A vector of length greater than 2 of what current explanatory variables are being used in the model.
n	The number of explanatory variables in available to swap.
prevpos	A vector of previous best spots
quad	Whether to include quadratic terms. ie (x1*x1) as potential swaps.

**Value**

a matrix with the possible forms by column.

---

pFSA

*pFSA: Pareto Feasible Solution Algorithm*


---

### Description

A function using a Feasible Solution Algorithm to estimate a set of models which are on the Pareto frontiers for chosen criteria

### Usage

```
pFSA(
  numFronts = 2,
  pselExpr = NULL,
  plot.it = TRUE,
  formula,
  data,
  fitfunc = lm,
  fixvar = NULL,
  quad = FALSE,
  m = 2,
  numrs = 1,
  cores = 1,
  interactions = T,
  criterion = AIC,
  minmax = "min",
  checkfeas = NULL,
  var4int = NULL,
  min.nonmissing = 1,
  return.models = FALSE,
  fix.formula = NULL,
  ...
)
```

### Arguments

numFronts	integer number of estimated frontiers to return
pselExpr	expression used by function psel to estimate pareto frontiers. <code>help(psel)</code> .
plot.it	TRUE/FALSE for whether to plot the pareto frontiers
formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data	a data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model.
fitfunc	the method that should be used to fit the model. For Example: <code>lm</code> , <code>glm</code> , or other methods that rely on formula, data, and other inputs.

fixvar	variable(s) to fix in the model. Usually a covariate that should always be included (Example: Age, Sex). Will still consider it with interactions. Default is NULL.
quad	Include quadratic terms or not. Logical.
m	order of terms to include. If interactions is set to TRUE then m is the order of interactions to be considered. For Subset selection (interaction=F), m is the size of the subset to examine. Defaults to 2.
numrs	number of random starts to perform.
cores	number of cores to use while running. Note: Windows can only use 1 core. See mclapply for details. If function detects a Windows user it will automatically set cores=1.
interactions	whether to include interactions in model. Defaults to TRUE.
criterion	which criterion function to either maximize or minimize. For linear models one can use: r.squared, adj.r.squared, cv5.lmFSA (5 Fold Cross Validation error), cv10.lmFSA (10 Fold Cross Validation error), apress (Allen's Press Statistic), int.p.val (Interaction P-value), AIC, BIC.
minmax	whether to minimize or maximize the criterion function
checkfeas	vector of variables that could be a feasible solution. These variables will be used as the last random start.
var4int	specification of which variables to check for marginal feasibility. Default is NULL
min.nonmissing	the combination of predictors will be ignored unless this many of observations are not missing
return.models	bool value to specify whether return all the fitted models which have been checked
fix.formula	...
...	see arguments taken by function FSA or other functions. help(FSA).

### Value

list of a matrix of all models obtained from FSA (fits) and their criteria. Also a matrix of the estimated frontiers that were requested. The Key column in fits, and pbound refers to the column number of the variables contained in the model fit. For instance, Key="42,96" would refer to the model which contains the variable in the 42nd column and 96th column of the designated dataset.

### Examples

```
N <- 1000 #number of obs
P <- 100 #number of variables
data <- data.frame(matrix(rnorm(N*(P+1)), nrow = N, ncol = P+1))
sln <- pFSA(formula = "X101~1", data = data, m = 2, criterion = c(max_abs_resid,r.squared),
  minmax = c("min","max"),numrs = 10,numFronts = 2,
  pselExpr =rPref::low(max_abs_resid)*rPref::high(r.squared),plot.it = TRUE)
```

---

plot.FSA                      *Diagnostic Plots for FSA solutions*

---

**Description**

Diagnostic Plots for FSA solutions

**Usage**

```
## S3 method for class 'FSA'  
plot(x, ask = F, easy = T, ...)
```

**Arguments**

x	FSA object to see diagnostic plots on.
ask	logical; if TRUE, the user is asked before each plot. See help(plot.lm).
easy	logical; should diagnostic plots be presented in easy to read format?
...	arguments to be passed to other functions.

**Value**

diagnostic plots to plot window.

**Examples**

```
#use mtcars package see help(mtcars)  
data(mtcars)  
colnames(mtcars)  
fit<-lmFSA(formula="mpg~hp*wt",data=mtcars,fixvar="hp",  
          quad=FALSE,m=2,numrs=10,save_solutions=FALSE,cores=1)  
plot(x=fit)
```

---

predict.FSA                      *Prediction function for FSA solutions*

---

**Description**

Prediction function for FSA solutions

**Usage**

```
## S3 method for class 'FSA'  
predict(object, ...)
```

**Arguments**

object            FSA object to conduct predictions on.  
 ...               other parameters passed to fitmodels or predict functions. See help(fitmodels)  
 or help(predict) for assistance.

**Value**

list of predicted values from each FSA model.

**Examples**

```
#use mtcars package see help(mtcars)
data(mtcars)
colnames(mtcars)
fit<-lmFSA(formula="mpg~hp*wt",data=mtcars,fixvar="hp",
            quad=FALSE,m=2,numrs=10,save_solutions=FALSE,cores=1)
predict(fit)
predict(fit,newdata=mtcars[1:15,])
```

---

print.FSA

*Printing function for FSA solutions*

---

**Description**

Printing function for FSA solutions

**Usage**

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

**Arguments**

x                    FSA object to print details about.  
 ...                  arguments to be passed to other functions.

**Value**

list of Feasible Solution Formula's, Original Fitted model formula and criterion function and times converged to details.

**Examples**

```
#use mtcars package see help(mtcars)
data(mtcars)
colnames(mtcars)
fit<-lmFSA(formula="mpg~hp*wt",data=mtcars,fixvar="hp",
            quad=FALSE,m=2,numrs=10,save_solutions=FALSE,cores=1)
print(fit)
```

---

QICu.geeglm	<i>Return QICu for geepack::geeglm</i>
-------------	--

---

**Description**

Computes quasi-likelihood under the independence criterion (QICu)

**Usage**

```
QICu.geeglm(gee.obj)
```

**Arguments**

gee.obj	geeglm obj
---------	------------

---

r.squared	<i>An rFSA Criterion Function.</i>
-----------	------------------------------------

---

**Description**

rFSA Criterion Function to compute R squared.

**Usage**

```
r.squared(model, name = "R Squared")
```

**Arguments**

model	lm or glm fit to be passed.
name	passed to print.FSA

---

rmse	<i>An rFSA Criterion Function.</i>
------	------------------------------------

---

**Description**

rFSA Criterion Function to compute Root Mean Squared Error.

**Usage**

```
rmse(model, name = "RMSE")
```

**Arguments**

model	lm or glm fit to be passed.
name	passed to print.FSA

---

summary.FSA

*Summary function for FSA solutions*


---

### Description

Summary function for FSA solutions

### Usage

```
## S3 method for class 'FSA'
summary(object, ...)
```

### Arguments

object            FSA object to see summaries on.  
 ...              arguments to be passed to other functions.

### Value

list of summarized lm or glm output.

### Examples

```
#use mtcars package see help(mtcars)
data(mtcars)
colnames(mtcars)
fit<-lmFSA(formula="mpg~hp*wt", data=mtcars, fixvar="hp",
            quad=FALSE, m=2, numrs=10, save_solutions=FALSE, cores=1)
summary(fit)
```

---

swaps

*Variables to include in first steip of an mth order interaction model determined from the Feasible Soution Alorithm.*


---

### Description

Finds the swaps available given a current position.

### Usage

```
swaps(cur, n, quad = FALSE, yindex)
```

**Arguments**

cur	A vector of length greater than 2 of what current explanatory variables are being used in the model.
n	The number of explanatory variables in available to swap.
quad	Whether to include quadratic terms. ie (x1*x1) as potential swaps.
yindex	index of response variable.

**Value**

a matrix with the possible forms by column.

---

twFSA	<i>twFSA</i>
-------	--------------

---

**Description**

A function for termwise feasibility

**Usage**

```
twFSA(
  formula,
  data,
  fitfunc = lm,
  fixvar = NULL,
  quad = FALSE,
  cores = 1,
  criterion = AIC,
  minmax = "min",
  checkfeas = NULL,
  var4int = NULL,
  min.nonmissing = 1,
  ...
)
```

**Arguments**

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data	a data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model.
fitfunc	the method that should be used to fit the model. For Example: <code>lm</code> , <code>glm</code> , or other methods that rely on formula, data, and other inputs.
fixvar	variable(s) to fix in the model. Usually a covariate that should always be included (Example: Age, Sex). Will still consider it with interactions. Default is <code>NULL</code> .

quad	Include quadratic terms or not. Logical.
cores	number of cores to use while running. Note: Windows can only use 1 core. See mclapply for details. If function detects a Windows user it will automatically set cores=1.
criterion	which criterion function to either maximize or minimize. For linear models one can use: r.squared, adj.r.squared, cv5.lmFSA (5 Fold Cross Validation error), cv10.lmFSA (10 Fold Cross Validation error), apress (Allen's Press Statistic), int.p.val (Interaction P-value), AIC, BIC.
minmax	whether to minimize or maximize the criterion function
checkfeas	vector of variables that could be a feasible solution. These variables will be used as the last random start.
var4int	specification of which variables to check for marginal feasibility. Default is NULL
min.nonmissing	the combination of predictors will be ignored unless this many of observations are not missing
...	other arguments passed to fitfunc.

**Value**

matrix of results

---

which.max.na                      *An rFSA Internal Function.*

---

**Description**

rFSA function to compute the maximum value from a vector with NA's.

**Usage**

```
which.max.na(vec)
```

**Arguments**

vec                      Vector to be passed.

---

`which.min.na`                    *An rFSA Internal Function.*

---

**Description**

rFSA function to compute the minimum value from a vector with NA's.

**Usage**

`which.min.na(vec)`

**Arguments**

`vec`                    Vector to be passed.

# Index

adj.r.squared, [2](#)  
apress, [3](#)

bdist, [3](#)

fitmodels, [4](#)  
fitted.FSA, [4](#)  
FSA, [5](#)

glmFSA (FSA), [5](#)

int.p.val, [7](#)

list.criterion, [7](#)  
lmFSA (FSA), [5](#)

max\_abs\_resid, [8](#)

nextswap, [8](#)

pFSA, [9](#)  
plot.FSA, [11](#)  
predict.FSA, [11](#)  
print.FSA, [12](#)

QICu.geeglm, [13](#)

r.squared, [13](#)  
rmse, [13](#)

summary.FSA, [14](#)  
swaps, [14](#)

twFSA, [15](#)

which.max.na, [16](#)  
which.min.na, [17](#)