Package ‘optimsimplex’

July 29, 2017

Type Package
Title R Port of the 'Scilab' Optimsimplex Module
Version 1.0-6
Date 2017-07-28
Author Sebastien Bihorel, Michael Baudin (author of the original module)
Maintainer Sebastien Bihorel <sb.pmlab@gmail.com>
Depends optimbase (>= 1.0-8),methods
License CeCILL-2
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2017-07-28 22:34:32 UTC

R topics documented:

optim simplex-package ........................................ 2
Function evaluations ........................................... 3
Get functions ................................................... 4
optim simplex ................................................... 5
optim simplex.destroy ......................................... 8
optim simplex.log .............................................. 9
optim simplex.reflect ......................................... 10
optim simplex.shrink ......................................... 11
The goal of this package is to provide a building block for optimization algorithms based on a simplex. The `optimsimplex` package may be used in the following optimization methods:

- the simplex method of Spendley et al.,
- the method of Nelder and Mead,
- the Box’s algorithm for constrained optimization,
- the multi-dimensional search by Torczon,
- etc ...

**Features** The following is a list of features currently provided:

- Manage various simplex initializations
  - initial simplex given by user,
  - initial simplex computed with a length and along the coordinate axes,
  - initial regular simplex computed with Spendley et al. formula,
  - initial simplex computed by a small perturbation around the initial guess point,
  - initial simplex computed from randomized bounds.
- sort the vertices by increasing function values,
- compute the standard deviation of the function values in the simplex,
- compute the simplex gradient with forward or centered differences,
- shrink the simplex toward the best vertex,
- etc...

**Details**

- Package: optimsimplex
- Type: Package
- Version: 1.0-6
- Date: 2017-07-28
- License: CeCILL-2
- LazyLoad: yes
See vignette('optimsimplex',package='optimsimplex') for more information.

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

Description

These functions compute the value of the function at the vertices points stored in the current simplex object and stored them back into the simplex object. optimsimplex.computevf determines how many vertices are stored in the simplex object and delegates the calculation of the function values to optimsimplex.compsomefv.

Usage

optimsimplex.computevf(this = NULL, fun = NULL, data = NULL)
optimsimplex.compsomefv(this = NULL, fun = NULL, indices = NULL, data = NULL)

Arguments

this The current simplex object, containing the nbve x n matrix of vertex coordinates (i.e. x element), where n is the dimension of the space and nbve the number of vertices.

fun The function to compute at vertices. The function is expected to have the following input and output arguments:

```r
myfunction <- function(x, this){
...  return(list(f=f,this=this))
}
```

where x is a row vector and this a user-defined data, i.e. the data argument.

data A user-defined data passed to the function. If data is provided, it is passed to the callback function both as an input and output argument. data may be used if the function uses some additional parameters. It is returned as an output parameter because the function may modify the data while computing the function value. This feature may be used, for example, to count the number of times that the function has been called.

indices A vector of increasing integers from 1 to nbve.

Value

optimsimplex.computevf and optimsimplex.compsomefv return a list with the following ele-
Get functions

ments:

**this**  The updated simplex object.
**data**  The updated user-defined data.

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

optimsimplex

---

Get functions  Optimsimplex Get Function Class

Description

The functions extract the content to various elements of a simplex object:

- `optimsimplex.getall`  Get all the coordinates and the function values of all the vertices.
- `optimsimplex.getallfv`  Get all the function values of all the vertices.
- `optimsimplex.getallx`  Get all the coordinates of all the vertices.
- `optimsimplex.getfv`  Get the function value at a given index.
- `optimsimplex.getn`  Get the dimension of the space of the simplex.
- `optimsimplex.getnbve`  Get the number of vertices of the simplex.
- `optimsimplex.getve`  Get the vertex at a given index in the current simplex.
- `optimsimplex.getx`  Get the coordinates of the vertex at a given index in the current simplex.

Usage

```
optimsimplex.getall(this = NULL)
optimsimplex.getallfv(this = NULL)
optimsimplex.getallx(this = NULL)
optimsimplex.getfv(this = NULL, ive = NULL)
optimsimplex.getn(this = NULL)
optimsimplex.getnbve(this = NULL)
optimsimplex.getve(this = NULL, ive = NULL)
optimsimplex.getx(this = NULL, ive = NULL)
```

Arguments

- **this**  A simplex object.
- **ive**  Vertex index.
Value

- `optimsimplex.getall` Return a nbve x n+1 matrix, where n is the dimension of the space, nbve is the number of vertices and with the following content:
  - `simplex[k,1]` is the function value of the vertex k, with k = 1 to nbve,
  - `simplex[k,2:(n+1)]` is the coordinates of the vertex k, with k = 1 to nbve.

- `optimsimplex.getallfv` Return a row vector of function values, which k^th element is the function value for the vertex k, with k = 1 to nbve.

- `optimsimplex.getallx` Return a nbve x n matrix of vertex coordinates; any given vertex is expected to be stored at row k, with k = 1 to nbve.

- `optimsimplex.getfv` Return a numeric scalar.

- `optimsimplex.getn` Return a numeric scalar.

- `optimsimplex.getnbve` Return a numeric scalar.

- `optimsimplex.getve` Return an object of class ‘vertex’, i.e. a list with the following elements:
  - `n` The dimension of the space of the simplex.
  - `x` The coordinates of the vertex at index ive.
  - `fv` The value of the function at index ive.

- `optimsimplex.getx` Return a row vector, representing the coordinates of the vertex at index ive.

Author(s)

- Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
- Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

- `optimsimplex`

Description

These functions support the S3 class ‘optimsimplex’ and are intended to either create objects of this class or check if an object is of this class.

Usage

```r
optimsimplex(coords = NULL, fun = NULL, data = NULL, method = NULL,
             x0 = NULL, len = NULL, deltausual = NULL, deltazero = NULL,
             boundsmax = NULL, boundsmin = NULL, nbve = NULL,
             simplex0 = NULL)

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

## S3 method for class 'optimsimplex'
is(x)

### Arguments

coods
The matrix of point estimate coordinates in the simplex. The coords matrix is expected to be a nbve x n matrix, where n is the dimension of the space and nbve is the number of vertices in the simplex, with nbve>= n+1. Only used if method is set to NULL.

fun
The function to compute at vertices. The function is expected to have the following input and output arguments:

```r
myfunction <- function(x, this){
  ...
  return(list(f=f,this=this))
}
```

where x is a row vector and this a user-defined data, i.e. the data argument.

data
A user-defined data passed to the function. If data is provided, it is passed to the callback function both as an input and output argument. data may be used if the function uses some additional parameters. It is returned as an output parameter because the function may modify the data while computing the function value. This feature may be used, for example, to count the number of times that the function has been called.

method
The method used to create the new optimsimplex object, either 'axes', 'pfeffer', 'randbounds', 'spendley' or 'oriented'.

x0
The initial point estimates, as a row vector of length n.

gen
The dimension of the simplex. If length is a value, that unique length is used in all directions. If length is a vector with n values, each length is used with the corresponding direction. Only used if method is set to 'axes' or 'spendley'.

deltausual
The absolute delta for non-zero values. Only used if method is set to 'pfeffer'.

deltazero
The absolute delta for zero values. Only used if method is set to 'pfeffer'.

boundsmin
A vector of minimum bounds. Only used if method is set to 'randbounds'.

boundsmax
A vector of maximum bounds. Only used if method is set to 'randbounds'.

nbve
The total number of vertices in the simplex. Only used if method is set to 'randbounds'.

simplex0
The initial simplex. Only used if method is set to 'oriented'.

x
An object of class 'optimsimplex'.

...optional arguments to 'print' or 'plot' methods.
Details

All arguments of `optimsimplex` are optional. If no input is provided, the new `optimsimplex` object is empty.

If `method` is `NULL`, the new `optimsimplex` object is created by `optimsimplex.coords`. If `coords` is `NULL`, the `optimsimplex` object is empty; otherwise, `coords` is used as the initial vertex coordinates in the new simplex.

If `method` is set to 'axes', the initial vertex coordinates are stored in a `nbve x n` matrix built as follows:

\[
\begin{bmatrix}
.1 & x0[1] & \ldots & x0[n] & | & len[1] & \ldots & 0 \\
[\ldots] & \ldots & \ldots & \ldots & | & \ldots & \ldots & \ldots \\
[.nbve] & x0[1] & \ldots & x0[n] & | & 0 & \ldots & len[n]
\end{bmatrix}
\]

If `method` is set to 'pfeffer', the new `optimsimplex` object is created using the Pfeffer's method, i.e. a relative delta for non-zero values and an absolute delta for zero values.

If `method` is set to 'randbounds', the initial vertex coordinates are stored in a `nbve x n` matrix consisting of the initial point estimates (on the first row) and a `(nbve-1) x n` matrix of randomly sampled numbers between the specified the bounds. The number of vertices `nbve` in the `optimsimplex` is arbitrary.

If `method` is set to 'spendley', the new `optimsimplex` object is created using the Spendely's method, i.e. a regular simplex made of `nbve = n+1` vertices.

If `method` is set to 'oriented', the new `optimsimplex` object is created in sorted order. The new simplex has the same sigma-length of the base simplex, but is "oriented" depending on the function value. The created `optimsimplex` may be used, as Kelley suggests, for a restart of Nelder-Mead algorithm.

The `optimsimplexNtostring` function is a utility function, which formats the content of a `optimsimplex` object into a single string of characters.

Value

The `optimsimplex` function returns a list with the following elements:

- `newobj` An object of class 'simplex', i.e. a list with the following elements:
  - `verbose` The verbose option, controlling the amount of messages. Set to `FALSE`.
  - `x` The coordinates of the vertices, with size `nbve x n`.
  - `n` The dimension of the space.
  - `fv` The values of the function at given vertices. It is a column matrix of length `nbve`.
  - `nbve` The number of vertices.
- `data` The updated `data` input argument.

Author(s)

Author of Scilab `optimsimplex` module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)
References


"Detection and Remediation of Stagnation in the Nelder-Mead Algorithm Using a Sufficient Decrease Condition", SIAM J. on Optimization, Kelley C.T., 1999


"Grid Restrained Nelder-Mead Algorithm", Arpad Burmen, Janez Puhan, Tadej Tuma, Computational Optimization and Applications, Volume 34, Issue 3 (July 2006), Pages: 359 - 375


Examples

```r
myfun <- function(xLthis){return(list(f=sum(x^2)Lthis)Lthis)}
mat <- matrix(c(0L1L0L0L0L1),ncol=2)
optimsimplex()
optimsimplex(coords=mat,x0=1:4,fun=myfun)
optimsimplex(method=’axes’,x0=1:4,fun=myfun)
optimsimplex(method=’pfeffer’,x0=1:6,fun=myfun)
opt <- optimsimplex(method=’randbounds’,x0=1:6,boundsmin=rep(0,6),
                      boundsmax=rep(10,6),fun=myfun)
opt
optimsimplex(method=’spendley’,x0=1:6,fun=myfun,len=10)
optimsimplex(method=’oriented’,simplex=opt$newobj,fun=myfun)
```

Erase Simplex Object

This function erases the coordinates of the vertices (x) and the function values (fv) in a simplex object.
Usage

```c
optimsimplex.destroy(this = NULL)
```

Arguments

this A simplex object.

Value

Return an updated simplex object for which the content of the \( x \) and \( fv \) elements were set to NULL.

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

```c
optimsimplex
```

---

Usage

```c
optimsimplex.log(this = NULL, msg = NULL)
```

Arguments

this An simplex object.

msg A message to print.

Value

Do not return any value but print \( \text{msg} \) to screen if the verbose in \( \text{this} \) is set to 1.

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

```c
optimsimplex
```
Simplex Reflection

Description

This function returns a new simplex by reflection of the current simplex with respect to the first vertex in the simplex. This move is used in the centered simplex gradient.

Usage

```r
optim simplex.reflect(this = NULL, fun = NULL, data = NULL)
```

Arguments

- **this**: An simplex object.
- **fun**: The function to compute at vertices. The function is expected to have the following input and output arguments:
  ```r
  myfunction <- function(x, this){
    ...
    return(list(f=f,this=this))
  }
  
  where x is a row vector and this a user-defined data, i.e. the data argument.
- **data**: A user-defined data passed to the function. If data is provided, it is passed to the callback function both as an input and output argument. data may be used if the function uses some additional parameters. It is returned as an output parameter because the function may modify the data while computing the function value. This feature may be used, for example, to count the number of times that the function has been called.

Value

Return a list with the following elements:

- **r**: The reflected simplex object.
- **data**: The updated user-defined data.

Author(s)

- Author of Scilab optim simplex module: Michael Baudin (INRIA - Digiteo)
- Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

- **optim simplex**
optimsimplex.shrink  Simplex Shrink

Description
This function shrinks the simplex with given coefficient sigma and returns an updated simplex. The shrink is performed with respect to the first point in the simplex.

Usage
optimsimplex.shrink(this = NULL, fun = NULL, sigma = 0.5, data = NULL)

Arguments
- **this**: An simplex object
- **fun**: The function to compute at vertices. The function is expected to have the following input and output arguments:

  ```r
def myfunction <- function(x, this){
  ...
  return(list(f=f,this=this))
}
```

  where x is a row vector and this a user-defined data, i.e. the data.
- **sigma**: The shrinkage coefficient. The default value is 0.5.
- **data**: A user-defined data passed to the function. If data is provided, it is passed to the callback function both as an input and output argument. data may be used if the function uses some additional parameters. It is returned as an output parameter because the function may modify the data while computing the function value. This feature may be used, for example, to count the number of times that the function has been called.

Value
Return a list with the following elements:
- **this**: The updated simplex object.
- **data**: The updated user-defined data.

Author(s)
Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also
optimsimplex
Description

These functions enable various calculations and checks on the current simplex:

- `optimsimplex.center` Compute the center of the current simplex.
- `optimsimplex.check` Check the consistency of the data in the current simplex.
- `optimsimplex.deltafv` Compute the vector of function value differences with respect to the function value at the first vertex (the lowest).
- `optimsimplex.deltafvmax` Compute the difference of function value between the lowest and the highest vertices. It is expected that the first vertex (`this$x[1]`) is associated with the smallest function value and that the last vertex (`this$x[nbve]`) is associated with the highest function value.
- `optimsimplex.dirmat` Compute the matrix of simplex direction, i.e. the matrix of differences of vertex coordinates with respect to the first vertex.
- `optimsimplex.fvmean` Compute the mean of the function values in the current simplex.
- `optimsimplex.fvstdev` Compute the standard deviation of the function values in the current simplex.
- `optimsimplex.fvvariance` Compute the variance of the function values in the current simplex.
- `optimsimplex.size` Determines the size of the simplex.
- `optimsimplex.sort` Sort the simplex by increasing order of function value, so the smallest function is at the first vertex.
- `optimsimplex.xbar` Compute the center of n vertices, by excluding the vertex with index `iexcl`. The default of `iexcl` is the number of vertices: in that case, if the simplex is sorted in increasing function value order, the worst vertex is excluded.

Usage

```plaintext
optimsimplex.center(this = NULL)
optimsimplex.check(this = NULL)
optimsimplex.deltafv(this = NULL)
optimsimplex.deltafvmax(this = NULL)
optimsimplex.dirmat(this = NULL)
optimsimplex.fvmean(this = NULL)
optimsimplex.fvstdev(this = NULL)
optimsimplex.fvvariance(this = NULL)
optimsimplex.size(this = NULL, method = NULL)
optimsimplex.sort(this = NULL)
optimsimplex.xbar(this = NULL, iexcl = NULL)
```
Arguments

- **this**: The current simplex.
- **method**: The method to use to compute the size of the simplex. The available methods are the following:
  - `'sigmaplus'` (this is the default) The sigmamplus size is the maximum 2-norm length of the vector from each vertex to the first vertex. It requires one loop over the vertices.
  - `'sigmaminus'` The sigmaminus size is the minimum 2-norm length of the vector from each vertex to the first vertex. It requires one loop over the vertices.
  - `'Nash'` The 'Nash' size is the sum of the norm of the norm-1 length of the vector from the given vertex to the first vertex. It requires one loop over the vertices.
  - `'diameter'` The diameter is the maximum norm-2 length of all the edges of the simplex. It requires 2 nested loops over the vertices.
- **iexcl**: The index of the vertex to exclude in center computation.

Value

- `optimsimplex.center`: Return a vector of length nbve, where nbve is the number of vertices in the current simplex.
- `optimsimplex.check`: Return an error message if the dimensions of the various elements of the current simplex do not match.
- `optimsimplex.deltav`: Return a column vector of length nbve-1.
- `optimsimplex.deltavmax`: Return a numeric scalar.
- `optimsimplex.dirmat`: Return a n x n numeric matrix, where n is the dimension of the space of the simplex.
- `optimsimplex.fvmean`: Return a numeric scalar.
- `optimsimplex.fvstdev`: Return a numeric scalar.
- `optimsimplex.fvvariance`: Return a numeric scalar.
- `optimsimplex.size`: Return a numeric scalar.
- `optimsimplex.sort`: Return an updated simplex object.
- `optimsimplex.xbar`: Return a row vector of length n.

Author(s)

Author of Scilab `optimsimplex` module: Michael Baudin (INRIA - Digiteo)

Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

References


"Iterative Methods for Optimization", C.T. Kelley, 1999, Chapter 6., section 6.2
Set functions

See Also

- `optimsimplex`

---

Set functions

Optimsimplex Set Function Class

Description

The functions assign content to various elements of a simplex object:

- `optimsimplex.setall` Set all the coordinates and the function values of all the vertices.
- `optimsimplex.setallfv` Set all the function values of all the vertices.
- `optimsimplex.setallx` Set all the coordinates of all the vertices.
- `optimsimplex.setfv` Set the function value at a given index.
- `optimsimplex.setn` Set the dimension of the space of the simplex.
- `optimsimplex.setnbve` Set the number of vertices of the simplex.
- `optimsimplex.setve` Set the coordinates of the vertex and the function values at a given index in the current simplex.
- `optimsimplex.setx` Set the coordinates of the vertex at a given index in the current simplex.

Usage

```plaintext
optimsimplex.setall(this = NULL, simplex = NULL)
optimsimplex.setallfv(this = NULL, fv = NULL)
optimsimplex.setallx(this = NULL, x = NULL)
optimsimplex.setfv(this = NULL, ive = NULL, fv = NULL)
optimsimplex.setn(this = NULL, n = NULL)
optimsimplex.setnbve(this = NULL, nbve = NULL)
optimsimplex.setve(this = NULL, ive = NULL, fv = NULL, x = NULL)
optimsimplex.setx(this = NULL, ive = NULL, x = NULL)
```

Arguments

- `this` A simplex object.
- `simplex` The simplex to set. It is expected to be a nbve x n+1 matrix where n is the dimension of the space, nbve is the number of vertices and with the following content:
  - `simplex[k,1]` is the function value of the vertex k, with k = 1 to nbve,
  - `simplex[k,2:(n+1)]` is the coordinates of the vertex k, with k = 1 to nbve.
- `fv` A row vector of function values; `fv[k]` is expected to be the function value for the vertex k, with k = 1 to nbve. For `optimsimplex.setfv`, `fv` is expected to be a numerical scalar.
x The nbve x n matrix of vertex coordinates; the vertex is expected to be stored in x[k,1:n], with k = 1 to nbve. For optimsimplex.setve and optimsimplex.setx, x is expected to be a row matrix.

ive Vertex index.
n The dimension of the space of the simplex.
nbve The number of vertices of the simplex.

Value

Return a updated simplex object this.

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

optim simplex

---

**Description**

These functions support the S3 classes 'simplex' and 'vertex'. They are intended to either create objects of these classes or check if an object is of these classes

**Usage**

```r
simplex(verbatim,x,n,fv,nbve)
vertex(x,n,fv)
```

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

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

```r
## S3 method for class 'simplex'
is(x)
```

```r
## S3 method for class 'vertex'
is(x)
```
Arguments

verbosethe verbose option, controlling the amount of messages
x
The coordinates of the vertices, with size nbve x n in a simplex object or 1 x n in a vertex.
n
The dimension of the space.
fv
The values of the function at given vertices. It is a column matrix of length nbve in a simplex or a single value in a vertex.
nbve
The number of vertices in a simplex.
... optional arguments to 'print' or 'plot' methods.

Details

A simplex of size n x nbve is essentially a collection of vertex of size n.

Value

simplex returns a list with the following elements: verbose, x, n, fv, and nbve. vertex returns a list with the following elements: x, n, and fv.

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)
Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

Simplex gradient

Description

optimSimplex.gradientfv determines the simplex gradient of the function which is computed by the secondary functions optimSimplex.gradcenter and optimSimplex.gradforward.

Usage

optimSimplex.gradientfv(this = NULL, fun = NULL, method = "forward", data = NULL)
optimSimplex.gradcenter(this = NULL, fun = NULL, data = NULL)
optimSimplex.gradforward(this = NULL)
Arguments

this An simplex object
fun The function to compute at vertices. The function is expected to have the following input and output arguments:

```r
myfunction <- function(x, this){
  ...
  return(list(f=f, this=this))
}
```

where x is a row vector and this a user-defined data, i.e. the data argument.

method The method used to compute the simplex gradient. Two methods are available: 'forward' and 'centered'. The 'forward' method uses the current simplex to compute the gradient (using `optimsimplex.dirmat` and `optimsimplex.deltafv`). The 'centered' method creates an intermediate simplex and computes the average.

data A user-defined data passed to the function. If data is provided, it is passed to the callback function both as an input and output argument. data may be used if the function uses some additionnal parameters. It is returned as an output parameter because the function may modify the data while computing the function value. This feature may be used, for example, to count the number of times that the function has been called.

Value

`optimsimplex.gradientfv` returns a list with the following elements:

- **g** A column vector of function gradient (with length `this$n`).
- **data** The updated user-defined data.

`optimsimplex.gradcenter` returns a list with the following elements:

- **g** A column vector of function gradient (with length `this$n`).
- **data** The updated user-defined data.

`optimsimplex.gradforward` returns a column vector of function gradient (with length `this$n`).

Author(s)

Author of Scilab optimsimplex module: Michael Baudin (INRIA - Digiteo)

Author of R adaptation: Sebastien Bihorel (<sb.pmlab@gmail.com>)

See Also

`optimsimplex, optimsimplex.dirmat, optimsimplex.deltafv`
# Index

**Topic method**
- Function evaluations, 3
- Get functions, 4
- optimsimplex, 5
- optimsimplex.destroy, 8
- optimsimplex.log, 9
- optimsimplex.reflect, 10
- optimsimplex.shrink, 11
- optimsimplex.utils, 12
- Set functions, 14
- simplex, 15
- Simplex gradient, 16

**Topic package**
- optimsimplex-package, 2

## Function evaluations, 3

## Get functions, 4

- is.optimsimplex (optimsimplex), 5
- is.simplex (simplex), 15
- is.vertex (simplex), 15

- optimsimplex, 4, 5, 5, 9–11, 14, 15, 17
- optimsimplex-package, 2
- optimsimplex.center (optimsimplex.utils), 12
- optimsimplex.check (optimsimplex.utils), 12
- optimsimplex.compsomefv (Function evaluations), 3
- optimsimplex.computefv (Function evaluations), 3
- optimsimplex.deltafv, 17
- optimsimplex.deltafv (optimsimplex.utils), 12
- optimsimplex.deltafvmax (optimsimplex.utils), 12
- optimsimplex.destroy, 8
- optimsimplex.dirmat, 17

- optimsimplex.dirmat (optimsimplex.utils), 12
- optimsimplex.fvmean (optimsimplex.utils), 12
- optimsimplex.fvstdev (optimsimplex.utils), 12
- optimsimplex.fvvvariance (optimsimplex.utils), 12
- optimsimplex.getall (Get functions), 4
- optimsimplex.getallfv (Get functions), 4
- optimsimplex.getallx (Get functions), 4
- optimsimplex.getfv (Get functions), 4
- optimsimplex.getn (Get functions), 4
- optimsimplex.getnbve (Get functions), 4
- optimsimplex.getve (Get functions), 4
- optimsimplex.getx (Get functions), 4
- optimsimplex.gradcenter (Simplex gradient), 16
- optimsimplex.gradforward (Simplex gradient), 16
- optimsimplex.gradients (Simplex gradient), 16
- optimsimplex.log, 9
- optimsimplex.reflect, 10
- optimsimplex.setall (Set functions), 14
- optimsimplex.setallfv (Set functions), 14
- optimsimplex.setallx (Set functions), 14
- optimsimplex.setfv (Set functions), 14
- optimsimplex.setn (Set functions), 14
- optimsimplex.setnbve (Set functions), 14
- optimsimplex.setve (Set functions), 14
- optimsimplex.setx (Set functions), 14
- optimsimplex.shrink, 11
- optimsimplex.size (optimsimplex.utils), 12
- optimsimplex.sort (optimsimplex.utils), 12
- optimsimplex.utils, 12
optimsimplex.xbar (optimsimplex.utils), 12
print.optimsimplex (optimsimplex), 5
print.simplex (simplex), 15
print.vertex (simplex), 15

Set functions, 14
simplex, 15
Simplex gradient, 16
vertex (simplex), 15