Package ‘optimsimplex’
October 14, 2022

Type Package
Title R Port of the 'Scilab' Optimsimplex Module
Version 1.0-8
Date 2022-01-28

Depends optimbase (>= 1.0-8),methods
Suggests knitr (>= 1.28),rmarkdown (>= 2.2)
License CeCILL-2
Encoding UTF-8
VignetteBuilder knitr
LazyLoad yes
NeedsCompilation no
Author Sebastien Bihorel [aut, cre],
                   Michael Baudin [aut]
Maintainer Sebastien Bihorel <sb.pmlab@gmail.com>
Repository CRAN
Date/Publication 2022-01-28 21:10:06 UTC

R topics documented:

  optimsimplex-package .................................................. 2
  Function evaluations ..................................................... 3
  Get functions ............................................................ 4
  optimsimplex ............................................................ 5
## Description

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: 1.0-8
Date: 2022-01-28
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>)

---

### Function evaluations

**Computation of Function Value(s)**

**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.computefv` determines how many vertices are stored in the simplex object and delegates the calculation of the function values to `optimsimplex.compsomefv`.

**Usage**

```r
optimsimplex.computefv(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 vertice 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 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.
- **indices**: A vector of increasing integers from 1 to nbve.

**Value**

`optimsimplex.computefv` 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 vertice 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

S3 optimsimplex class

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

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

- **coords**: 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:

  ```
  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.

- **len**: 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 `optim simplex` are optional. If no input is provided, the new `optim simplex` object is empty.

If `method` is `NULL`, the new `optim simplex` object is created by `optim simplex.coords`. If `coords` is `NULL`, the `optim simplex` 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:

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

If `method` is set to 'pfeffer', the new `optim simplex` 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 `optim simplex` is arbitrary.

If `method` is set to 'spendley', the new `optim simplex` 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 `optim simplex` 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 `optim simplex` may be used, as Kelley suggests, for a restart of Nelder-Mead algorithm.

The `optim simplex.tostring` function is a utility function, which formats the content of a `optim simplex` object into a single string of characters.

Value

The `optim simplex` 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 `optim simplex` 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(x,this){return(list(f=sum(x^2),this=this))}
mat <- matrix(c(0,1,0,0,0,1),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)
```

**optimsimplex.destroy**

Erase Simplex Object

**Description**

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

Description
This function prints a message to screen (or log file).

Usage
optimsimplex.log(this = NULL, msg = NULL)

Arguments
this An simplex object.
msg A message to print.

Value
Do not return any value but print msg to screen if the verbose in 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
optimsimplex
**optimsimplex.reflect**  \hspace{5ex} *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
optimsimplex.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 optimsimplex module: Michael Baudin (INRIA - Digiteo)

Author of R adaptation: Sebastien Bihorel (&lt;sb.pmlab@gmail.com&gt;)

### See Also

- `optimsimplex`
**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**

```r
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
  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
**Optimsimplex Utility Functions**

**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**

- `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

optim simplex.center Return a vector of length nbve, where nbve is the number of vertices in the current simplex.

optim simplex.check Return an error message if the dimensions of the various elements of the current simplex do not match.

optim simplex.deltafv Return a column vector of length nbve-1.

optim simplex.deltafvmax Return a numeric scalar.

optim simplex.dirmat Return a n x n numeric matrix, where n is the dimension of the space of the simplex.

optim simplex.fvmean Return a numeric scalar.

optim simplex.fvstdev Return a numeric scalar.

optim simplex.fvvariance Return a numeric scalar.

optim simplex.size Return a numeric scalar.

optim simplex.sort Return an updated simplex object.

optim simplex.xbar Return a row vector of length n.

Author(s)

Author of Scilab optim simplex 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
See Also

optim simplex

S3 osimplex and vertex classes

Description

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

Usage

osimplex(verbose,x,n,fv,nbve)

vertex(x,n,fv)

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

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

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

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

Arguments

verbose The 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

osimplex 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>)

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

```r
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 gradient

• 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 optim simplex.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 optim simplex.setve and optim simplex.setx,
x is expected to be a row matrix.

eive 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 optim simplex module: Michael Baudin (INRIA - Digiteo)

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

See Also

optim simplex

Simplex gradient

Description

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

Usage

optim simplex.gradientfv(this = NULL, fun = NULL, method = "forward",
data = NULL)
optim simplex.gradcenter(this = NULL, fun = NULL, data = NULL)
optim simplex.gradforward(this = NULL)

Arguments

this An simplex object

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

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

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 \texttt{optimsimplex.dirmat} and \texttt{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**

\texttt{optimsimplex.gradientfv} returns a list with the following elements:

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

\texttt{optimsimplex.gradcenter} returns a list with the following elements:

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

\texttt{optimsimplex.gradforward} returns a column vector of function gradient (with length \( n \)).

**Author(s)**

Author of Scilab \texttt{optimsimplex} module: Michael Baudin (INRIA - Digiteo)

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

**See Also**

\texttt{optimsimplex}, \texttt{optimsimplex.dirmat}, \texttt{optimsimplex.deltafv}
Index

* method
  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
  optim simplex.utils, 12
  osimplex, 14
  Set functions, 15
  Simplex gradient, 16
* package
  optim simplex-package, 2

Function evaluations, 3
Get functions, 4

is.optim simplex (optim simplex), 5
is.osimplex (osimplex), 14
is.vertex (osimplex), 14

optim simplex, 4, 5, 5, 9–11, 14, 16, 17
optim simplex-package, 2
optim simplex.center
  (optim simplex.utils), 12
optim simplex.check
  (optim simplex.utils), 12
optim simplex.compsomefv (Function evaluations), 3
optim simplex.computefv (Function evaluations), 3
optim simplex.deltafv, 17
optim simplex.deltafv
  (optim simplex.utils), 12
optim simplex.deltafvmax
  (optim simplex.utils), 12
optim simplex.destroy, 8
optim simplex.dirmat, 17
optim simplex.dirmat
  (optim simplex.utils), 12
optim simplex.fvmean
  (optim simplex.utils), 12
optim simplex.fvstdev
  (optim simplex.utils), 12
optim simplex.fvvariance
  (optim simplex.utils), 12
optim simplex.getall (Get functions), 4
optim simplex.getallfv (Get functions), 4
optim simplex.getallx (Get functions), 4
optim simplex.getn (Get functions), 4
optim simplex.getnbve (Get functions), 4
optim simplex.getve (Get functions), 4
optim simplex.getx (Get functions), 4
optim simplex.gradcenter (Simplex gradient), 16
optim simplex.gradforward (Simplex gradient), 16
optim simplex.gradientfv (Simplex gradient), 16
optim simplex.log, 9
optim simplex.reflect, 10
optim simplex.setall (Set functions), 15
optim simplex.setallfv (Set functions), 15
optim simplex.setallx (Set functions), 15
optim simplex.setf (Set functions), 15
optim simplex.setn (Set functions), 15
optim simplex.setnbve (Set functions), 15
optim simplex.setve (Set functions), 15
optim simplex.setx (Set functions), 15
optim simplex.shrink, 11
optim simplex.size (optim simplex.utils), 12
optim simplex.sort (optim simplex.utils), 12
optim simplex.utils, 12
INDEX

optimsimplex.xbar (optimsimplex.utils), 12
osimplex, 14

print.optimsimplex (optimsimplex), 5
print.osimplex (osimplex), 14
print.vertex (osimplex), 14

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