Package 'labsimplex'

July 22, 2025

Version 0.1.2

Date 2020-05-18

Title Simplex Optimization Algorithms for Laboratory and Manufacturing Processes

Author Cristhian Paredes [aut, cre], Jesús Ágreda [aut]

Maintainer Cristhian Paredes <craparedesca@unal.edu.co>

Description Simplex optimization algorithms as firstly proposed by Spendley et al. (1962) <doi:10.1080/00401706.1962.10490033> and later modified by Nelder and Mead (1965) <doi:10.1093/comjnl/7.4.308> for laboratory and manufacturing processes. The package also provides tools for graphical representation of the simplexes and some example response surfaces that are useful in illustrating the optimization process.

License GPL (>= 2)

Encoding UTF-8

Imports scatterplot3d (>= 0.3-41), ggplot2

RoxygenNote 7.0.2.9000

Suggests knitr, rmarkdown, FrF2

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2020-06-03 16:10:06 UTC

Contents

`labsimplex-package` `labsimplex`*: Simplex Optimization Algorithms for Laboratory and Manufacturing Processes*

Description

The `labsimplex` package implements the simplex optimization algorithms firstly proposed by Spendley et al. (1962) <doi:10.1080/00401706.1962.10490033> and later modified by Nelder and Mead (1965) <doi:10.1093/comjnl/7.4.308> for laboratory and manufacturing processes. The package also provides tools for graphical representation of the simplexes and some example response surfaces that are useful for illustrating the optimization process.

Details

A simplex is a geometric element defined as the simpler polytope possible in an *n*-dimensional space. If the space has *n* dimensions, the simplexes there will have *n+1* corners called vertexes. The simplexes in two and three-dimensional spaces are the well-known triangle and tetrahedron, respectively.
In the simplex optimization algorithms, the experimental variables are represented by the dimensions in the abstract space. Each vertex in the simplex represents an experiment, then the coordinates of the vertex represent the values for the variables in that experimental setting. The experiments must be performed and a response must be assigned to each vertex. In the optimization process, one of the vertexes is discarded in favor of a new one that must be evaluated. In the first simplex, the vertex with the worst response is discarded. The second worst vertex in this simplex is discarded in the following simplex and the procedure is repeated until the optimum is reached or a response good enough is obtained. The process of discarding a vertex and generating a new one is known as a movement of the simplex.
In this document, the words vertex and experiment are used interchangeably. The same applies to dimensions and experimental variables.

`labsimplex` functions

This package uses list objects of class `'smplx'` to store the simplex information, including all the coordinates of the vertexes and their responses.
The `labsimplex` functions can generate a new `'smplx'` class object, assing responses to the vertices

to generate the next one and to visualize different spatial representations of the n-dimensional simplex in 2D or 3D projections. Detailed information can be found by typing `vignette('labsimplex')`.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

References

Nelder, J. A., and R. Mead. 1965. "A Simplex Method for Function Minimization." The Computer Journal 7 (4): 308–13.

Spendley, W., G. R. Hext, and F. R0. Himsworth. 1962. "Sequential Application of Simplex Designs in Optimization and Evolutionary Operation." Technometrics 4 (4): 441–61.

`addSimplex2Surface` *Adds the simplex movements to a response surface contour*

Description

The function complements the contour plot produced by using `cntr` function. Given a contour plot and a simplex (an object of class `smplx`) the function adds the simplex movements to the contour plot to illustrate the optimization process and the path that was followed.

Usage

```
addSimplex2Surface(p, simplex)
```

Arguments

`p`	contour plot produced by using `cntr` function
`simplex`	simplex object of class `smplx`. Usually produced using `exampleOptimization`

Value

a `ggplot` object with the optimization path over the contour plot provided.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

See Also

`cntr` `exampleOptimization`

Examples

```
simplex <- exampleOptimization(surface = exampleSurfaceR2,
                               centroid = c(7, 340),
                               stepsize = c(1.2, 15))
p <- cntr(surface = exampleSurfaceR2)
p <- addSimplex2Surface(p = p, simplex = simplex)
print(p)
```

`adjustVertex` *Modify the coordinates of given vertexes of a simplex*

Description

Changes the coordinates of previously generated vertexes when slight differences were impossible to avoid at the moment of setting the experiment variables (e.g. small differences in mass components when preparing a mixture). This function allows the correction of the vertexes of a simplex in order to produce movements of the simplex based on the actual coordinates.

Usage

```
adjustVertex(simplex, newcoords, overwrite = FALSE)
```

Arguments

`simplex`	object of class `smplx` with the simplex information. See `labsimplex`
`newcoords`	List with elements named like the vertexes to be modified. Each element must have a vector with the actual (ordered) coordinates used in the experiment. `NA` values may be used to indicate unchanged coordinates
`overwrite`	logical argument. If `TRUE`, the output simplex will replace the one provided in the `simplex` parameter. Default `overwrite = FALSE`

Value

An object of class `smplx` with the modified simplex information.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

Examples

```
simplex <- labsimplex(n = 3, start = c(7, 25, 0.15),
                      stepsize = c(0.2, 5, 0.02))
adjustVertex(simplex = simplex, newcoords = list(Vertex.1 = c(7.1, NA, NA),
                                                 Vertex.3 = c(6.9, NA, 0.155)),
             overwrite = TRUE)
```

cntr *Contour plot of example response surfaces*

Description

Plots a ggplot with the contour of the bivariate example response surfaces included in the package.

Usage

```
cntr(surface, length = 150, noise = 0, x1lim = c(278, 365), x2lim = c(0, 14))
```

Arguments

surface	example response surface to use. See exampleSurfaceR2 and exampleSurfaceR2.2pks.
length	number of levels to use in each explanatory variables
noise	absolute noise to be included in the results
x1lim	limits for the first variable (temperature in exampleSurfaceR2 and exampleSurfaceR2.2pks)
x2lim	limits for the second variable (pH in exampleSurfaceR2 and exampleSurfaceR2.2pks)

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Examples

```
p <- cntr(surface = exampleSurfaceR2, length = 200)
print(p)
```

exampleOptimization *Performs a complete simplex optimization over a response surface*

Description

The function uses the information in a simplex or creates a new one by using the defined centroid and step-size to perform a simplex optimization using the responses produced in the example response surfaces included in the package.

Usage

```
exampleOptimization(surface, simplex = NULL, centroid = c(7, 340),
  stepsize = c(0.6, 10), algor = "fixed", experiments = 17, noise = 0)
```

Arguments

surface	example response surface to be used. See exampleSurfaceR2, exampleSurfaceR3 and exampleSurfaceR2.2pks
simplex	object of class smplx with the simplex information. See labsimplex
centroid	numeric vector of size n with coordinates of the centroid
stepsize	numeric vector of size n with the step-sizes for each coordinate
algor	algorithm to be followed in the vertex generation. 'fixed' for a fixed step-size simplex or 'variable' for a variable step-size simplex
experiments	number of vertexes to evaluate
noise	absolute noise to be included in the results

Value

An object with class smplx with the simplex optimization data.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

ExampleSurfaces *Multivariate functions that define hypothetical response surfaces.*

Description

The functions in this section simulate the yield of hypothetical chemical reactions as a function of temperature, pH, and concentration (the latter only for exampleSurfaceR3()). Those functions are useful to illustrate most concepts of the simplex optimization algorithms implemented in the labsimplex package, as shown in the vignentte of the package. This vignette can be visualized by running vignette('labsimplex').

Usage

```
exampleSurfaceR2(x1, x2, noise = 0)

exampleSurfaceR2.2pks(x1, x2, noise = 0)

exampleSurfaceR3(x1, x2, x3, noise = 0)
```

Arguments

x1	temperature in Kelvin. Numeric between 278 and 365.
x2	pH. Numeric between 0 and 14.
noise	absolute noise included in the response surface result. Default to zero.
x3	concentration in arbitrary units. Numeric between 0 and 1. Only used in exampleSurfaceR3().

Details

Parameters `x1`, `x2`, and `x3` may be supplied as vectors in which case all must have the same length. Boundary values are proposed consistently with real-life limitations in aqueous media. If such boundaries are violated in the variables input, a negative result without physical meaning is returned. This negative value represents an *infinitely bad response* that will force the simplex to move in another direction.

Value

`exampleSurfaceR2(x1, x2, noise = 0)` defines a response surface with one maximum at pH 10 and 300 K.

`exampleSurfaceR2.2pks(x1, x2, noise = 0)` defines a response surface with global and local maxima at pH 10 and 300 K and pH 4.5 and 340 K, respectively.

`exampleSurfaceR3(x1, x2, x3, noise = 0)` defines a response surface with one maximum at pH 10, 300 K and a concentration of 0.5.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

See Also

`cntr`, `prspctv` and `exampleOptimization`

Examples

```
exampleSurfaceR2(x1 = 320, x2 = 4.5)
exampleSurfaceR2(x1 = c(310, 320), x2 = c(4.5, 5.8))
exampleSurfaceR2(x1 = c(310, 320), x2 = c(4.5, 5.8), noise = 5)
exampleSurfaceR2.2pks(x1 = 320, x2 = 4.5)
exampleSurfaceR2.2pks(x1 = c(310, 320), x2 = c(4.5, 5.8))
exampleSurfaceR2.2pks(x1 = c(310, 320), x2 = c(4.5, 5.8), noise = 5)
exampleSurfaceR3(x1 = 320, x2 = 4.5, x3 = 0.3)
exampleSurfaceR3(x1 = c(310, 320), x2 = c(4.5, 5.8), x3 = c(0.3, 0.5))
exampleSurfaceR3(x1 = c(310, 320), x2 = c(4.5, 5.8), x3 = c(0.3, 0.5),
                 noise = 5)
```

---

generateVertex *Generates the new vertex of a simplex optimization*

Description

Gives the coordinates for the new vertex that must be performed based on the responses for the vertexes on the current simplex and considering the optimization criteria.

Usage

```
generateVertex(simplex, qflv = NULL, crit = "max", algor = "fixed",
  overwrite = FALSE)
```

Arguments

simplex	object of class smplx with the simplex information. See labsimplex
qflv	response for the vertex (or vertexes) without responses
crit	optimization criteria indicating if the goal is maximize ('max') or minimize ('min') the response. It can also be a numeric value to which the response is supposed to approach
algor	algorithm to be followed in the vertex generation. 'fixed' for a fixed step-size simplex or 'variable' for a variable step-size simplex
overwrite	logical argument. If TRUE, the output simplex will replace the one provided in the simplex parameter. Default overwrite = FALSE

Details

When minimization is the criteria, the algorithm will tend to approach zero. If negative responses are possible and the most negative value is desired, a very large negative number must be provided in crit parameter.

Value

An object of class smplx with the new simplex information including the conditions for the new experiment to be permormed.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

Examples

```
simplex <- labsimplex(n = 3, centroid = c(320, 7, 0.4),
                      stepsize = c(35, 2, 0.3))
## The experiments must be performed and the responses passed to qflv.
## Here we get the responses by using an example response surface
## included in the package:
##
## Initially, the response must be provided for all the vertexes
response <- exampleSurfaceR3(x1 = simplex$coords[, 1],
                             x2 = simplex$coords[, 2],
                             x3 = simplex$coords[, 3])
simplex <- generateVertex(simplex = simplex, qflv = response)

## After this, the last vertex is the only one that must be evaluated
response <- exampleSurfaceR3(x1 = simplex$coords[nrow(simplex$coords), 1],
                             x2 = simplex$coords[nrow(simplex$coords), 2],
```

```
                              x3 = simplex$coords[nrow(simplex$coords), 3])
simplex <- generateVertex(simplex = simplex, qflv = response)

## Alternatively the simplex object can overwrite the older one:
generateVertex(simplex = simplex, qflv = response, overwrite = TRUE)
```

`labsimplex` *Generates a simplex object*

Description

The simplex (a list with class `smplx`) contains the coordinates of the n+1 vertices that define a simplex in an *n*-dimensional space. By default, the function produces a regular simplex centered at the origin. The coordinates of the regular simplex are transformed into the real variables space by using the information of the start or centroid and step-size. The only non-optional parameter is `n` that relates the simplex dimensionality. Once the simplex is generated, the experiments under the conditions indicated for each variable at each vertex must be carried and the response obtained. Those responses are assigned to the `smplx` object at the moment of generating the new vertex (see `generateVertex`).

Usage

```
labsimplex(n, start = NULL, centroid = NULL, stepsize = NULL,
  usrdef = NULL, var.name = NULL)
```

Arguments

`n`	dimensionality of the simplex (i.e. number of variables)
`start`	numeric vector of size `n` with coordinates of the first vertex
`centroid`	numeric vector of size `n` with coordinates of the centroid
`stepsize`	numeric vector of size `n` with the step-sizes for each coordinate
`usrdef`	`(n+1)xn` matrix containig in (n+1) rows the n coordinates for each vertex
`var.name`	vector containing the names for the variables

Details

The regular simplex coordinates are generated following the general algorithm for the cartesian coordinates of a regular n-dimensional simplex. This algorithm considers that all vertices must be equally distanced from simplex centroid and all angles subtended between any two vertexes and the centroid of a simplex are equal to *arccos(-1/n)*.
If the vertexes coordinates are manually given (in `usr.def` parameter), the function checks if the faces produced belong to different hyperplanes. This avoids the generation of a degenerated simplex.

Value

An object of class `smplx` with the information of the new simplex.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

References

Nelder, J. A., and R. Mead. 1965. "A Simplex Method for Function Minimization." The Computer Journal 7 (4): 308–13.

Spendley, W., G. R. Hext, and F. R0. Himsworth. 1962. "Sequential Application of Simplex Designs in Optimization and Evolutionary Operation." Technometrics 4 (4): 441–61.

Examples

```
  simplex <- labsimplex(n = 3)
  simplex <- labsimplex(n = 3, centroid = c(350, 7, 0.4),
                        stepsize = c(35, 2, 0.3),
                        var.name = c('temperature', 'pH', 'concentration'))
  simplex <- labsimplex(n = 3, usrdef = rbind(c(390, 8, 0.2), c(330, 8, 0.2),
                                  c(330, 6, 0.6), c(330, 6, 0.1)))
  ## Not run:
    ## User defined coordinates may define a degenerated simplex:
    simplex <- labsimplex(n = 3,
                          usrdef = rbind(c(390, 8, 0.3), c(340, 8, 0.3),
                                         c(355, 8, 0.3), c(340, 5, 0.1)))

## End(Not run)
```

---

plot.smplx *Draws a two dimentional plot of the vertexes in a simplex*

Description

The function generates a 2D plot of the vertexes in a simplex optimization when simplex dimensionality is at least 2. When dimensionality is higher than 2, the plot produced is a projection of the selected variables.

Usage

```
## S3 method for class 'smplx'
plot(x, sel.dim = NULL, all.ver = TRUE, all.lin = TRUE,
  expand = TRUE, exp.fac = 1.5, ...)
```

Arguments

x	object of class smplx.
sel.dim	numeric or char vector for variables to be considered when simplex dimensionality is higher than 2. By default, the first two are chosen. If the vector is numeric, it must contain the ordinal numbers corresponding to the desired variables. If the vector is of class char, it must contain the names of such dimensions.
all.ver	logical default to TRUE. Should all vertexes be plotted? If FALSE, the function draws only the vertexes of the current simplex.
all.lin	logical default to TRUE. Should all lines be drawn? If FALSE, the function draws only the lines of the last simplex.
expand	logical default to TRUE. Should the plot scales be expanded?
exp.fac	expansion factor used when expand = TRUE.
...	other graphical parameters used in plot

Details

For 3D representations of simplexes with dimensionality higher than 2 you can use plotSimplex3D.

Value

2D plot of the simplex coordinates.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

See Also

plotSimplex3D

Examples

```
plot(x = labsimplex(n = 2, centroid = c(7, 340), stepsize = c(1.2, 15)))

## Several options are posible when visualizing higher order simplexes
plot(x = labsimplex(n = 3))
plot(x = labsimplex(n = 3), sel.dim = c(2, 3))

## Simplex movements can be visualized after some experiments has been
## performed
simplex <- exampleOptimization(surface = exampleSurfaceR2,
                               centroid = c(7, 340),
                               stepsize = c(1.2, 15), experiments = 16)
plot(x = simplex)
```

plotSimplex3D *Draws a three dimentional plot of the vertexes in a simplex*

Description

The function generates a 3D plot of the vertexes in a simplex optimization when simplex dimensionality is at least 3. When dimensionality is higher than 3, the plot produced is a projection of the selected variables.

Usage

```
plotSimplex3D(simplex, sel.dim = NULL, all.ver = TRUE, all.lin = TRUE,
  main = NULL, angle = 30, ...)
```

Arguments

simplex	object of class smplx with the simplex information. See labsimplex
sel.dim	numeric or char vector for variables to be considered when simplex dimensionality is higher than 3. By default, the first three are chosen. If the vector is numeric, it must contain the ordinal numbers corresponding to the desired variables. If the vector is of class char, it must contain the names of such dimensions.
all.ver	logical default to TRUE. Should all vertexes be plotted? If FALSE, the function draws only the vertexes of the current simplex.
all.lin	logical default to TRUE. Should all lines be drawn? If FALSE, the function draws only the lines of the last simplex.
main	title for the plot.
angle	angle for perspective between x and y axis.
...	other arguments passed to scatterplot3d

Value

3D plot of the simplex coordinates.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

See Also

plot.smplx

Examples

```
plotSimplex3D(simplex = labsimplex(n = 3, centroid = c(350, 11, 0.7),
                                   stepsize = c(10, 0.5, 0.1),
                                   var.name = c('temperature', 'pH',
                                                'concentration')))

## Several options are posible when visualizing higher order simplexes
plotSimplex3D(simplex = labsimplex(n = 8))
plotSimplex3D(simplex = labsimplex(n = 8), sel.dim = c(4, 6, 8))

## Simplex movements can be visualized after some experiments has been
## performed
simplex <- exampleOptimization(surface = exampleSurfaceR3,
                               centroid = c(350, 11, 0.7),
                               stepsize = c(10, 0.5, 0.1),
                               experiments = 18)
plotSimplex3D(simplex = simplex, angle = 80)
```

---

plotSimplexResponse *Plots the response versus the vertex number of a simplex optimization.*

Description

The function generates a `ggplot` object from an object with class `smplx`. The response is plotted against the vertex number.

Usage

```
plotSimplexResponse(x, ...)
```

Arguments

`x`	object with class `smplx` containig the coordinates of the vertices and their responses.
`...`	other graphical parameters used in `plot`

Details

If the simplex object being ploted was obtained using a variable size algorithm, some experimental points could be disregarded and will be shown with a red mark indicating that the vertex was not used in the obtention of new vertexes.

Value

Plot of response against vertex number.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

Examples

```
simplex <- exampleOptimization(surface = exampleSurfaceR3,
                               centroid = c(350, 11, 0.7),
                               stepsize = c(10, 0.5, 0.1),
                               experiments = 18, algor = 'variable')
plotSimplexResponse(simplex)
```

---

print.smplx *S3 method* print *for simplex objects*

Description

Prints simplex information.

Usage

```
## S3 method for class 'smplx'
print(x, extended = FALSE, conventions = TRUE, ...)
```

Arguments

x	simplex (object of class to be printed
extended	logical, if TRUE, the object is printed as a list containing all hiden elements
conventions	logical, if TRUE (default), the conventions used are printed
...	more parameters passed to print

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

prspctv *3D perspective plot of example response surfaces*

Description

Plots a persp plot of the bivariate example response surfaces included in the package.

Usage

```
prspctv(surface, length = 45, noise = 0, x1lim = c(278, 365),
  x2lim = c(0, 14), par = NULL, theta = 22, phi = 15, shade = 0.2,
  ticktype = "detailed", ...)
```

Arguments

surface	example response surface to use. See exampleSurfaceR2 and exampleSurfaceR2.2pks.
length	number of levels to use in each explanatory variables
noise	absolute noise to be included in the results
x1lim	limits for the first variable (temperature in exampleSurfaceR2 and exampleSurfaceR2.2pks)
x2lim	limits for the second variable (pH in exampleSurfaceR2 and exampleSurfaceR2.2pks)
par	list with graphical parameters (par).
theta	angles defining the viewing direction. theta gives the azimuthal direction and phi the colatitude.
phi	angles defining the viewing direction. theta gives the azimuthal direction and phi the colatitude.
shade	the shade at a surface facet is computed as ((1+d)/2)^shade, where d is the dot product of a unit vector normal to the facet and a unit vector in the direction of a light source. Values of shade close to one yield shading similar to a point light source model and values close to zero produce no shading. Values in the range 0.5 to 0.75 provide an approximation to daylight illumination.
ticktype	character: "simple" draws just an arrow parallel to the axis to indicate direction of increase; "detailed" draws normal ticks as per 2D plots.
...	additional graphical parameters (see par).

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

Examples

```
prspctv(surface = exampleSurfaceR2.2pks)
prspctv(surface = exampleSurfaceR2.2pks, theta = 35, phi = 25,
        expand = 0.75, xlab = 'Temperature (K)', ylab = 'pH',
        zlab = 'Yield (%)')
```

simplexExport *Exports the information contained in an object of class* smplx.

Description

Creates a text file with extension .smplx that contains the complete information contained in a simplex (an object with class smplx, see labsimplex). This file allows the continuation of an optimization proccess when the experiments take too long and multiple R sessions are required. The file produced is also useful to share the information of the optimization process. The exported simplex can be later imported with simplexImport.

Usage

```
simplexExport(simplex, filename = NULL, direc = NULL)
```

Arguments

simplex	object of class smplx containing the simplex to be exported
filename	string with the name (without extention) of the file that will be created. If not provided, the name of the simplex object is used.
direc	directory in which the file will be saved. If not provided, the current working directory is used.

Value

Generates a .smplx file containing all the information required to continue with the optimization process after the experiments have been carried.

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

See Also

simplexImport

Examples

```
simplex <- labsimplex(n = 5)
simplexExport(simplex = simplex)
```

simplexImport *Imports the information contained in a* .smplx *file.*

Description

The function reads and (optionally) loads into the environment the simplex (object of class smplx) contained in a .smplx file that was previously created using simplexExport.

Usage

```
simplexImport(filename, aut.load = TRUE, name = NULL)
```

Arguments

filename	string with the name of the file (without extension) to be imported. This file must be generated using simplexExport. The path must be included if the file is not in the current working directory.
aut.load	logical. Should the imported simplex object be directly loaded on the Environment? Default to TRUE.
name	name for the simplex object to be created if auth.ld = FALSE. When not provided, the name of the file is used.

Value

A smplx class object with the complete information of the simplex

Author(s)

Cristhian Paredes, <craparedesca@unal.edu.co>

Jesús Ágreda, <jagreda@unal.edu.co>

See Also

simplexImport

Examples

```
simplexR2 <- exampleOptimization(surface = exampleSurfaceR2)
simplexExport(simplex = simplexR2)
rm(simplexR2)
simplexImport(filename = "simplexR2")
```

Index