Package: MixOptim (via r-universe)

September 7, 2024

Type Package

Title Mixture Optimization Algorithm

Version 0.1.2

Author Cassandro Davi Emer

Maintainer Cassandro Davi Emer <cde@actisys.org>

Description Simple tools to perform mixture optimization based on the 'desirability' package by Max Kuhn. It also provides a plot routine using 'ggplot2' and 'patchwork'.

License GPL-2

Encoding UTF-8

LazyData true

Depends ggplot2, patchwork, desirability

RoxygenNote 7.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

Imports rlang

NeedsCompilation no

Repository CRAN

Date/Publication 2020-07-01 09:40:11 UTC

Contents

desirabilityPlot *Plot desirability profile*

Description

This function creates a graphical representation of the desirability profiles within the data range. It requires a data frame object generated from an optimization function. You can use a simpler data frame to plot the lines and present a more accurate data from another optimization call.

Usage

```
desirabilityPlot(functions, plotData, bestValues, desirab, types)
```

Arguments

functions	An array of functions
plotData	A data frame generated from an optimization function
bestValues	The optimal mixture composition to be presented
desirab	An array of desirability functions
types	An array of strings containing the attributes to be shown for each desirability function. Currently only accepts "max" or "min"

Value

A ggplot composite object from patchwork

mixtureFineOptim *Performs a restrict interval optimization*

Description

This function performs an optimization testing within an interval defined by the user using starting points and an alpha value. Since it is designed for more accurate searching, it does not allow the generation of the data frame for plotting.

Usage

```
mixtureFineOptim(
  functions,
  desirabilityModel,
  startPoint,
  step = 0.001,
  alpha = 0.02,
  verbose = TRUE
)
```

Arguments

functions	An array of functions
desirabilityModel	A desirability overallD model
startPoint	An array with the references (mid-points) for the optimization
step	The ammount of each increment in the optimization
alpha	Defines the range of the seach, as startPoint +- alpha for each x value
verbose	Defines if the user should be updated with the processing status (percentages)

Value

A list containg the data regarding the maximum desirability found

Examples

```
library(MixOptim)
dados <- read.table(header = TRUE, dec = ",", sep = "\t", text = "
x1 x2 x3 R1 R2 R3
1 0 0 0,76 8 5
1 0 0 0,75 8 5
0,5 0,5 0 1,4 7 7,5
0,5 0 0,5 0,55 8 10
0 1 0 4,1 4 10
0 1 0 4,4 4 10
0 0,5 0,5 0,9 7 12,5
0 0 1 0,42 9 15
0 0 1 0,4 10 15
0,6667 0,1667 0,1667 0,8 7 7,5
0,1667 0,6667 0,1667 1,7 7 10
0,1667 0,1667 0,6667 0,55 8 12,5
0,3333 0,3333 0,3333 0,8 8 10")

lm1 <- lm(data = dados, R1 ~ -1 + x1 + x2 + x3 + x1:x2 + x1:x3 + x2:x3)
summary(lm1)
flm1 <- function(x) 0.7678*x[1] + 4.2083*x[2] + 0.4274*x[3] - 4.3273*x[1]*x[2] +
      0.3070*x[1]*x[3] - 5.6101*x[2]*x[3]
lm2 <- lm(data = dados, R2 ~ -1 + x1 + x2 + x3)
summary(lm2)
flm2 <- function(x) 7.9742*x[1] + 4.5742*x[2] + 9.3742*x[3]
lm3 <- lm(data = dados, R3 ~ -1 + x1 + x2 + x3)
summary(lm3)
flm3 <- function(x) 4.9998461*x[1] + 9.9998461*x[2] + 14.9998461*x[3]

funcoes2 <- c(flm1, flm2, flm3)
des1<-dTarget(0.5, 0.6, 0.7)
des2<-dMax(8, max(dados$R2))
des3<-dMin(5, 10)
finalD<-dOverall(des1, des2, des3)

# code commented due to process time requirement
```

```
#teste <- mixtureOptim(funcoes2, finalD, 3, step = 0.01, plot = TRUE)
#desirabilityPlot(funcoes2, teste$plotData, teste$bestComposition,
#   list(des1, des2, des3), c("max", "max", "min"))

#teste2 <- mixtureFineOptim(funcoes2, finalD, teste$bestComposition, step = 0.0001)
#desirabilityPlot(funcoes2, teste$plotData, teste2$bestComposition,
#   list(des1, des2, des3), c("max", "max", "min"))
```

`mixtureOptim` *Performs a full interval optimization*

Description

This function performs a full interval optimization (0-1 for each x variable). It allows the creation of the data frame used for plotting.

Usage

```
mixtureOptim(
  functions,
  desirabilityModel,
  xCount,
  step = 0.01,
  plot = TRUE,
  verbose = TRUE
)
```

Arguments

`functions`	An array of functions
`desirabilityModel`	A desirability `overallD` model
`xCount`	The amount of `x` variables used in the functions
`step`	The ammount of each increment in the optimization
`plot`	Define is the data frame that can be used for the `desirabilityPlot` function will be create. Strongly affects performance
`verbose`	Defines if the user should be updated with the processing status (percentages)

Value

A list containg the data regarding the maximum desirability found

Examples

```
library(MixOptim)
saPred<-function(x) 17.3359 * x[1] + 30.7765 * x[2] + 20.0501 * x[3] + 0.7506 * x[1] * x[2] +
  (-6.3443 * x[1] * x[3]) + (-5.9291 * x[2] * x[3]) +(-25.3093 * x[1] * x[2] * x[3])
pvPred<-function(x)  0.884640 * x[1] + 0.789863 * x[2] + 0.825016 * x[3] +
  (-0.108964 * x[1] * x[2]) + 0.107167 * x[1] * x[3] + (-0.005220 * x[2] * x[3]) +
  1.625246 * x[1] * x[2] * x[3]

funcoes <- c(saPred, pvPred)
saD<-dMin(16.8293856, 31.41170555) #minimun value
pvD<-dMax(0.7796004, 0.9019796) #maximum value
overallD<-dOverall(saD, pvD)

a <- mixtureOptim(funcoes, overallD, 3, step = 0.05, plot = TRUE)
a$bestComposition
# next line will generate a ggplot/patchwork grid
#desirabilityPlot(funcoes, a$plotData, a$bestComposition, list(saD, pvD), c("min", "max"))
```

`mixtureRangeOptim` *Performs a specific range optimization*

Description

This function performs an optimization testing within an interval defined by the user using alpha values for each middle point provided. It allows the generation of the data frame required for plotting.

Usage

```
mixtureRangeOptim(
  functions,
  desirabilityModel,
  midPoints,
  alpha,
  step = 0.01,
  plot = TRUE,
  verbose = TRUE
)
```

Arguments

`functions`	An array of functions
`desirabilityModel`	A desirability `overallD` model
`midPoints`	An array with the references (mid-points) for the optimization
`alpha`	Defines the range of the seach, as `startPoint` +- `alpha` for each `x` value
`step`	The ammount of each increment in the optimization

plot	Define is the data frame that can be used for the desirabilityPlot function will be create. Strongly affects performance
verbose	Defines if the user should be updated with the processing status (percentages)

Value

A list containg the data regarding the maximum desirability found

Examples

```
library(MixOptim)
dados <- read.table(header = TRUE, sep = "\t", text = "
ID TiO2 Vehicle Extender A Extender B Hiding Scrub
1 0.05 0.20 0.30 0.45 7.8953 533.67
2 0.45 0.20 0.30 0.05 32.862 749
3 0.05 0.60 0.30 0.05 3.721 39.5
4 0.05 0.20 0.70 0.05 9.2751 203.25
5 0.25 0.20 0.30 0.25 20.132 555.25
6 0.05 0.40 0.30 0.25 4.7137 51.75
7 0.05 0.20 0.50 0.25 8.3829 342.75
8 0.25 0.40 0.30 0.05 16.245 84.75
9 0.25 0.20 0.50 0.05 22.639 360.75
10 0.05 0.40 0.50 0.05 5.4645 48
11 0.05 0.33 0.43 0.18 5.8882 76
12 0.18 0.20 0.43 0.18 17.256 386.25
13 0.18 0.33 0.30 0.18 12.351 136
14 0.18 0.33 0.43 0.05 14.499 75.5
15 0.10 0.25 0.35 0.30 10.548 325.75
16 0.30 0.25 0.35 0.10 22.096 359
17 0.10 0.45 0.35 0.10 6.2888 40.75
18 0.10 0.25 0.55 0.10 10.629 136.67
19 0.15 0.30 0.40 0.15 11.777 114")

hiding<-function(x) 67.748*x[1] + 7.291*x[2] + 11.419*x[3] + 14.578*x[4] -
   64.32*x[1]*x[2] + 35.878*x[1]*x[3] - 15.696*x[1]*x[4] - 31.006*x[2]*x[3] -
   38.668*x[2]*x[4] - 6.59*x[3]*x[4]
scrub<-function(x) 3937.5*x[1] + 899.3*x[2] + 502*x[3] + 2354.8*x[4] -
   8227.2*x[1]*x[2] - 3227.4*x[1]*x[3] - 2447.7*x[1]*x[4] - 2435.3*x[2]*x[3] -
   6325.1*x[2]*x[4] - 1050.3*x[3]*x[4]

funcoes2 <- c(hiding, scrub)
des1<-dMax(min(dados$Hiding), max(dados$Hiding))
des2<-dMin(min(dados$Scrub), max(dados$Scrub))
finalD<-dOverall(des1, des2)

# code commented due to process time requirement
#teste <- mixtureRangeOptim(funcoes2, finalD, midPoints = c(0.25, 0.4, 0.5, 0.25),
#    alpha = c(0.2, 0.2, 0.2, 0.2), step = 0.01, plot = TRUE)
#desirabilityPlot(funcoes2, teste$plotData, teste$bestComposition, list(des1, des2),
#    c("max", "min"))
#teste2 <- mixtureRangeOptim(funcoes2, finalD, midPoints = teste$bestComposition,
#    alpha = c(0.01, 0.01, 0.01, 0), step = 0.001, plot = FALSE)
```

```
#teste2
#desirabilityPlot(funcoes2, teste$plotData, teste2$bestComposition, list(des1, des2),
#    c("max", "min"))
```

Index