# Package 'SCOR'

July 21, 2025

**Type** Package

**Title** Spherically Constrained Optimization Routine

**Version** 1.1.2

**Depends** R (>= 3.5.0)

**Imports** doParallel, foreach, iterators, parallel

**Collate** 'imports.R' 'biomarker.R' 'SHUM.R' 'EHUM.R' 'ULBA.R' 'SCOR.R' 'SCOR-package.R' 'optimized_HUM.R' 'youden_points.R' 'YoudenBoxPlot.R'

**Maintainer** Debsurya De <debsurya001@gmail.com>

**Description** A non convex optimization package that optimizes any function under the criterion, combination of variables are on the surface of a unit sphere, as described in the paper : Das et al. (2019) <doi:10.48550/arXiv.1909.04024> .

**License** GPL-3

**URL** https://github.com/synx21/SCOR

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.0

**NeedsCompilation** no

**Author** Debsurya De [cre, aut],
Priyam Das [aut]

**Repository** CRAN

**Date/Publication** 2023-06-19 07:50:02 UTC

## Contents

---

AL *Alzheimer's disease neuropsychometric marker dataset*

---

## Description

The dataset is a subset of the longitudinal cohort of Washington University (WU) Alzheimer's Disease Research Center (ADRC). In the AL dataset, measurements of 12 neuropsychological markers were collected on 108 independent individuals of age 75. The individuals were classified into 3 groups based on published clinical demential rating (CDR).

## Usage

```
data(AL)
```

## Format

A data frame with 108 observations on the following 12 variables.

## Details

- ktemp. a numeric vector, measurements on the neuropsychometric test for "temporal factor".
- kpar. a numeric vector, measurements on the neuropsychometric test for "parietal factor".
- kfront. a numeric vector, measurements on the neuropsychometric test for "frontal factor".
- zpsy005. a numeric vector, measurements on the neuropsychometric test for "digital span forward".
- zpsy006. a numeric vector, measurements on the neuropsychometric test for "digital span backward".
- zinfo. a numeric vector, measurements on the neuropsychometric test for "information".
- zbentc. a numeric vector, measurements on the neuropsychometric test for "visual retention (10s)".
- zbentd. a numeric vector, measurements on the neuropsychometric test for "visual retention (copy)".
- zboston. a numeric vector,a numeric vector, measurements on the neuropsychometric test for "boston naming".
- zmentcon. a numeric vector,measurements on the neuropsychometric test for "mental control".
- zworflu. a numeric vector,measurements on the neuropsychometric test for "word fluency".
- zassc. a numeric vector,measurements on the neuropsychometric test for "associate learning".

## estimate_EHUM *Empirical Hyper Volume Under Manifolds*

### Description

An estimator of Hyper Volume Under Manifolds

### Usage

```
estimate_EHUM(beta, labels, x_mat)
```

### Arguments

| | |
|---|---|
| beta | The parameter we measure EHUM based on. |
| labels | The labels of the Columns of the data matrix. |
| x_mat | The Data Matrix |

### Value

Empirical Hyper-volume Under Maniforlds Estimate

### Examples

```
estimate_EHUM(rep(1, 12), colnames(AL), AL)


estimate_EHUM(1:10 , sample(c( rep("lab1", 10), rep("lab2", 10), rep("lab3", 10))),
matrix(rnorm(300), nrow = 10))
```

## estimate_SHUM *Smooth Approximations Of Empirical Hyper Volume Under Manifolds*

### Description

'SHUM' is a class of smoothed estimates of EHUM.

### Usage

```
estimate_SHUM(beta, labels, x_mat, p = 0)
```

### Arguments

| | |
|---|---|
| beta | The parameter we measure SHUM based on. |
| labels | The labels of the Columns of the data matrix. |
| x_mat | The Data Matrix |
| p | p decides whether to use $s_n(x)$ or $\phi_n(x)$. p = 1 stands for $\phi_n(x)$ and p = 0 stands for $s_n(x)$ |

### Value

Smooth approximation of the empirical Hyper-volume Under Manifolds Estimate

### References


- Maiti, Raju and Li, Jialiang and Das, Priyam and Feng, Lei and Hausenloy, Derek and Chakraborty, Bibhas
"A distribution-free smoothed combination method of biomarkers to improve diagnostic accuracy in multi-category classification"
(available at 'arXiv https://arxiv.org/abs/1904.10046).


### Examples

```
estimate_SHUM(rep(1, 12), colnames(AL), AL)
estimate_SHUM(rep(1, 12), colnames(AL), AL, p = 1)


estimate_SHUM(1:10 , sample(c( rep("lab1", 10), rep("lab2", 10), rep("lab3", 10))),
matrix(rnorm(300), nrow = 10))
```

---

## estimate_ULBA *Upper And Lower Bound Approach*

---

### Description

'ULBA' is an another approach to Hyper Volume Under Manifold Problem

### Usage

```
estimate_ULBA(beta, labels, x_mat)
```

### Arguments

| | |
|---|---|
| beta | The parameter we measure ULBA based on. |
| labels | The labels of the Columns of the data matrix. |
| x_mat | The Data Matrix |

## Value

Upper and Lower Bound Approach on empirical Hyper-volume Under Manifolds Estimate

## Examples

```
estimate_ULBA(rep(1, 12), colnames(AL), AL)


estimate_ULBA(1:10 , sample(c( rep("lab1", 10), rep("lab2", 10), rep("lab3", 10))),
matrix(rnorm(300), nrow = 10))
```

---

`optimized_HUM` *Optimizing Different Estimators Of Hyper Volume Under Manifold*

---

## Description

As we know 'SCOptim' is efficient in estimating maximizing Hyper Volume Under Manifolds Estimators, we made some pre-functions that optimizes specific Problems of EHUM,SHUM and ULBA.

## Usage

```
optimized_EHUM(
  beta_start,
  labels,
  x_mat,
  rho = 2,
  phi = 0.001,
  max_iter = 50000,
  s_init = 2,
  tol_fun = 1e-06,
  tol_fun_2 = 1e-06,
  minimize = FALSE,
  time = 36000,
  print = FALSE,
  lambda = 0.001,
  parallel = TRUE
)

optimized_SHUM(
  beta_start,
  labels,
  x_mat,
  p = 0,
  rho = 2,
```

```
  phi = 0.001,
  max_iter = 50000,
  s_init = 2,
  tol_fun = 1e-06,
  tol_fun_2 = 1e-06,
  minimize = FALSE,
  time = 36000,
  print = FALSE,
  lambda = 0.001,
  parallel = TRUE
)

optimized_ULBA(
  beta_start,
  labels,
  x_mat,
  rho = 2,
  phi = 0.001,
  max_iter = 50000,
  s_init = 2,
  tol_fun = 1e-06,
  tol_fun_2 = 1e-06,
  minimize = FALSE,
  time = 36000,
  print = FALSE,
  lambda = 0.001,
  parallel = TRUE
)
```

## Arguments

| | |
|---|---|
| beta_start | The initial guess for optimum $\beta$ by user |
| labels | Sample Sizes vector of that has number of elements in each category. It works like the labels of data matrix. |
| x_mat | The Data Matrix |
| rho | Step Decay Rate with default value 2 |
| phi | Lower Bound Of Global Step Size. Default value is $10^{-6}$ |
| max_iter | Max Number Of Iterations In each Run. Default Value is 50,000. |
| s_init | Initial Global Step Size. Default Value is 2. |
| tol_fun | Termination Tolerance on the function value. Default Value is $10^{-6}$ |
| tol_fun_2 | Termination Tolerance on the difference of solutions in two consecutive runs. Default Value is $10^{-6}$ |
| minimize | Binary Command to set SCOptim on minimization or maximization. FALSE is for minimization which is set default. |
| time | Time Allotted for execution of SCOptim |

| | |
|---|---|
| print | Binary Command to print optimized value of objective function after each iteration. FALSE is set fault |
| lambda | Sparsity Threshold. Default value is $10^{-3}$ |
| parallel | Binary Command to ask SCOptim to perform parallel computing. Default is set at TRUE. |
| p | This parameter exists for the case of optimized_SHUM only.p decides whether to use $s_n(x)$ or $\phi_n(x)$. p = 1 stands for $\phi_n(x)$ and p = 0 stands for $s_n(x)$ |

## Details

Optimization of EHUM, SHUM and ULBA using SCOptim.

## Value

Optimum Values Of HUM Estimates

## Examples

```
R <- optimized_SHUM(rep(1, 12), colnames(AL), AL, parallel = FALSE)
estimate_SHUM(R, colnames(AL), AL)
# This run will take about 10 mins on average based on computational capacity of the system
# Optimum value of HUM estimate noticed for this case : 0.8440681


R <- optimized_EHUM(rep(1, 12), colnames(AL), AL, parallel = FALSE)
estimate_EHUM(R, colnames(AL), AL)
# Optimum value of HUM estimate noticed for this case : 0.8403805

R <- optimized_ULBA(rep(1, 12), colnames(AL), AL, parallel = FALSE)
estimate_ULBA(R, colnames(AL), AL)
# Optimum value of HUM estimate noticed for this case : 0.9201903
```

---

# SCOptim *Spherically Constrained Optimization*

---

## Description

SCOptim runs our optimization algorithm, efficient in estimating maximizing Hyper Volume Under Manifolds Estimators.

## Usage

```
SCOptim(
  x0,
  func,
  rho = 2,
```

```
  phi = 0.001,
  max_iter = 50000,
  s_init = 2,
  tol_fun = 1e-06,
  tol_fun_2 = 1e-06,
  minimize = TRUE,
  time = 36000,
  print = FALSE,
  lambda = 0.001,
  parallel = FALSE
)
```

## Arguments

| | |
|---|---|
| x0 | The initial guess by user |
| func | The function to be optimized |
| rho | Step Decay Rate with default value 2 |
| phi | Lower Bound Of Global Step Size. Default value is $10^{-6}$ |
| max_iter | Max Number Of Iterations In each Run. Default Value is 50,000. |
| s_init | Initial Global Step Size. Default Value is 2. |
| tol_fun | Termination Tolerance on the function value. Default Value is $10^{-6}$ |
| tol_fun_2 | Termination Tolerance on the difference of solutions in two consecutive runs. Default Value is $10^{-6}$ |
| minimize | Binary Command to set SCOptim on minimization or maximization. TRUE is for minimization which is set default. |
| time | Time Allotted for execution of SCOptim |
| print | Binary Command to print optimized value of objective function after each iteration. FALSE is set fault |
| lambda | Sparsity Threshold. Default value is $10^{-3}$ |
| parallel | Binary Command to ask SCOptim to perform parallel computing. Default is set at FALSE. |

## Details

SCOptim is the modified version of RMPS, Recursive Modified Pattern Search. This is a blackbox algorithm efficient in optimizing non-differentiable functions. It works great in the shown cases of SHUM, EHUM and ULBA.

## Value

The point where the value Of the Function is maximized under a sphere.

## References


- Das, Priyam and De, Debsurya and Maiti, Raju and Chakraborty, Bibhas and Peterson, Christine B
"Estimating the Optimal Linear Combination of Biomarkers using Spherically Constrained Optimization"
(available at 'arXiv https://arxiv.org/abs/1909.04024).


## Examples

```
f <- function(x)
return(x[2]^2 + x[3]^3 +x[4]^4)

SCOptim(rep(1,10), f)

SCOptim(c(2,4,6,2,1), f, minimize = FALSE, print = TRUE)
#Will Print the List and Find the Maximum

SCOptim(c(1,2,3,4), f, time = 10, lambda = 1e-2)
#Will perform no iterations after 10 secs, Sparsity Threshold is 0.01
```

---

# youden_points *Finding Youden Indices*

---

## Description

A function to find Youden Indices and Cutpoints for number of categories less than equal to 3.

## Usage

```
youden_points(beta, labels, x_mat, grid_size = 100)
```

## Arguments

| | |
|---|---|
| beta | The parameter we do HUM based on |
| labels | The labels of the Columns of the data matrix. |
| x_mat | The Data Matrix |
| grid_size | The size of increment in the grid we check cutpoints against. Default value is 100. |

## Value

Youden Indices and Cut Points

## Examples

```
beta <- c(-0.399,-0.155,-0.265,-0.184,
     -0.267,0.666,-0.187,0.273,0.0463,0.167,0.163,0.178)

youden_points(beta, colnames(AL), AL)
```

---

`YoupointsBoxPlot` *Visualization Based On Youden Indices.*

---

## Description

A Box Plot Visualization Based On Youden Indices for less than equal to 3 categories.

## Usage

```
YoupointsBoxPlot(beta, labels, x_mat, cat_names = NULL, grid_size = 100)
```

## Arguments

| | |
|---|---|
| `beta` | The parameter we do HUM based on |
| `labels` | The labels of the Columns of the data matrix |
| `x_mat` | The Data Matrix |
| `cat_names` | The vector of strings containing category names. |
| `grid_size` | The size of increment in the grid we check cutpoints against. Default value is 100. |

## Value

Box Plot Visualization Based On Youden Indices

## Examples

```
beta <- c(-0.399,-0.155,-0.265,-0.184,
     -0.267,0.666,-0.187,0.273,0.0463,0.167,0.163,0.178)

YoupointsBoxPlot(beta, colnames(AL), AL, cat_names = c("Healthy", "MCI", "AD"))
```

# Index