Package ‘emoa’

March 6, 2020

Title Evolutionary Multiobjective Optimization Algorithms
Description Collection of building blocks for the design and analysis
of evolutionary multiobjective optimization algorithms.
Author Olaf Mersmann <olafm@statistik.tu-dortmund.de>
Maintainer Olaf Mersmann <olafm@statistik.tu-dortmund.de>
License GPL-2
URL http://www.statistik.tu-dortmund.de/~olafm/software/emoa/
LazyData yes
Version 0.5-0.1
Suggests RUnit

Collate 'cec2009.r' 'control.R' 'crowding_distance.r' 'dominance.r'
'emoa.r' 'front_edge.R' 'hypervolume.r' 'indicators.r'
'logger.R' 'poly_mutation.r' 'sb_crossover.r' 'selection.r'
'sympart.r' 'utilities.r'

Date
Repository CRAN
Date/Publication 2020-03-06 17:31:53 UTC
NeedsCompilation yes

R topics documented:

emoa-package ......................................................... 2
cec2007 ............................................................ 3
coalesce .............................................................. 4
crowding_distance .................................................. 4
dominated_hypervolume ............................................ 5
emoa_console_logger ............................................... 6
emoa_control ....................................................... 6
emoa_logger ........................................................... 7
emoa_null_logger .................................................. 8
hypervolume_indicator ............................................. 8
This package provides functions to construct evolutionary multiobjective optimization algorithms (EMOA). The long term goal is to also provide standard implementations of the most common EMOA in use today.

Without the hard work of many researchers who have published their source code under a liberal license, this package would not have been possible. In alphabetical order they are

- Michael H. Buselli
- Wessel Dankers
- Carlos Fonseca
- Joshua Knowles
- Huang Ling
- Wudong Liu
- Manuel Lopez-Ibanez
- Luis Paquete
- Ponnuthurai Nagaratnam Suganthany
- Santosh Tiwar
- Qingfu Zhang
- Aimin Zhou
- Shizheng Zhaoy
Description

This data set contains the hypervolume and R2 indicator results of the 8 different algorithms that took part in the CEC 2007 multiobjective optimization benchmark.

Usage

data(cec2007)

Format

A data frame with 456 observations of the following 9 variables.

- **algo**  Abbreviated name of algorithm
- **fun**   Name of benchmark function
- **d**     Dimension of objective space
- **n**     Number of function evaluations
- **metric** Name of quality metric
- **pdef**  Unique id for each combination of fun, d, n and metric
- **best**  Largest value of metric
- **median** Median value of metric
- **worst** Smallest value of metric
- **mean**  Average value of metric
- **std**   Standard deviation of metric

Source

http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/CEC2007-final-pdfs.zip

Examples

```r
## Not run:
data(cec2007)
require(lattice)
print(dotplot(algo ~ median | fun + metric, cec2007, groups=cec2007$n))

## End(Not run)
```
### coalesce

*Return first non null argument.*

#### Description
This function is useful when processing complex arguments with multiple possible defaults based on other arguments that may or may not have been provided.

#### Usage
```
coalesce(...)  
```

#### Arguments
- `...` List of values.

#### Value
First non null element in `...`.

#### Author(s)
Olaf Mersmann <olafm@statistik.tu-dortmund.de>

---

### crowding_distance

*Crowding Distance*

#### Description
Calculate crowding distances.

#### Usage
```
crowding_distance(front)  
```

#### Arguments
- `front` matrix of function values.

#### Value
crowding distance for each function value.

#### Author(s)
Olaf Mersmann <olafm@statistik.tu-dortmund.de>
dominated_hypervolume  Domated Hypervolume calculation

Description

donated_hypervolume calculates the dominated hypervolume of the points in points.

Usage

dominated_hypervolume(points, ref)

hypervolume_contribution(points, ref)

Arguments

points  Matrix containing the points one per column.
ref  Optional reference point. If not provided the maximum in each dimension is used.

Details

hypervolume_contribution calculates the hypervolume contribution of each point.
If no reference point ref is given, one is automatically calculated by determining the maximum in each coordinate.
Currently only one general algorithm is implemented due to Fonseca et.al. but work is underway to include others such as the Beume & Rudolph approach as well as the approach by Bradstreet et.al.
The 1D and 2D cases are handled separately by efficient algorithms. Calculates the exact dominated hypervolume of the points given in x subject to the reference point ref.

Value

For dominated_hypervolume the dominated hypervolume by the points in points with respect to the reference point ref. For hypervolume_contribution a vector giving the hypervolume solely dominated by that point.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

References

This code uses version 1.3 of the hypervolume code available from http://iridia.ulb.ac.be/~manuel/hypervolume. For a description of the algorithm see
emoa_control

**See Also**

See [nondominated_points](#) to extract the pareto front approximation from a given set of points and [nds_hv_selection](#) for a selection strategy based on the hypervolume contribution of each point.

---

**emoa_console_logger**

*console logger*

---

**Description**

Logger object that outputs log messages to the console

**Usage**

```r
emoa_console_logger(...)```

**Arguments**

... passed to `emoa_logger`.

**Details**

This is a wrapper that calls `emoa_logger(output=output,...)` internally and returns that logger.

**Value**

An `emoa_logger` object.

---

**emoa_control**

*Basic EMOA control parameters.*

---

**Description**

The following control parameters are recognized by `emoa_control`:

- **logger** `emoa_logger` object used to log events.
- **n** Number of parameters, defaults to the length of the longer of `upper` or `lower`.
- **d** Number of dimensions.

**Usage**

```r
emoa_control(f, upper, lower, ..., control, default)```
Arguments

- `f` Multiobjective optimization function.
- `upper` Upper bounds of parameter space.
- `lower` Lower bounds of parameter space.
- `...` Further arguments passed to `f`.
- `control` List of control parameters.
- `default` List of default control parameters.

Value

The control list with suitably adjusted arguments. Missing control parameters are taken from `default` or, if not present there, from an internal default.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

---

**emoa_logger** | *generic logger factory*

Description

Basic logger object with a flexible output routine.

Usage

```
emoa_logger(output, every = 10L)
```

Arguments

- `output` function used to display logging messages.
- `every` number of steps of the emoa between evaluations.

Value

An `emoa_logger` object.

See Also

`emoa_console_logger` and `emoa_null_logger` for convinience wrappers around `emoa_logger` providing useful defaults.
Description
Logger object that discards all log events.

Usage
emoa_null_logger(...)

Arguments
... ignored.

Value
An emoa_logger object.

Description
Calculates the quality indicator value of the set of points given in x with respect to the set given in o. As with all functions in emoa that deal with sets of objective values these are stored by column.

Usage
hypervolume_indicator(points, o, ref)
epsilon_indicator(points, o)
r1_indicator(points, o, ideal, nadir, lambda, utility = "Tchebycheff")
r2_indicator(points, o, ideal, nadir, lambda, utility = "Tchebycheff")
r3_indicator(points, o, ideal, nadir, lambda, utility = "Tchebycheff")
Arguments

points Matrix of points for which to calculate the indicator value stored one per column.
ref Reference point, if omitted, the nadir of the point sets is used.
ideal Ideal point of true Pareto front. If omitted the ideal of both point sets is used.
nadir Nadir of the true Pareto front. If omitted the nadir of both point sets is used.
lambda Number of weight vectors to use in estimating the utility.
utility Name of utility function.

Value

Value of the quality indicator.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

References


inbounds

Clip value to a given range

Description

Clip x to the interval [l, u]. This is useful to enforce box constraints.

Usage

inbounds(x, 1, u)

Arguments

x Value to clip.
l Lower limit.
u Upper limit.

Value

l if x < l, u if x > u else x.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>
### is_dominated

**Pareto dominance checks.**

**Description**

is_dominated returns which points from a set are dominated by another point in the set. %dominates% returns true if x Pareto dominates y and is_maximally_dominated returns TRUE for those points which do not dominate any other points.

**Usage**

```r
is_dominated(points)

is_maximally_dominated(points)
```

**Arguments**

- **points**
  
  Matrix containing points one per column.

**Value**

For is_dominated and is_maximally_dominated a boolean vector and for %dominates% a single boolean.

**Author(s)**

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

### nds_hv_selection

**Selection strategies**

**Description**

Selection strategies for EMOA.

**Usage**

```r
nds_hv_selection(values, n = 1, ...)

nds_cd_selection(values, n = 1, ...)
```

**Arguments**

- **values**
  
  Matrix of function values.

- **n**
  
  Number of individuals to select for replacement.

- **...**
  
  Optional parameters passed to hypervolume_contribution.
**nds_rank**

**Details**

The currently implemented strategies are nondominated sorting followed by either hypervolume contribution or crowding distance based ranking. Both of these implementations are currently limited to selecting a single individual for replacement.

**Author(s)**

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

---

**nds_rank**

* Nondominated sorting ranks

---

**Description**

Perform (partial) nondominated sort of the points in `points` and return the rank of each point.

**Usage**

```r
nds_rank(points, partial)
nondominated_ordering(points, partial)
```

**Arguments**

- `points` Matrix containing points one per column.
- `partial` Optional integer specifying the number of points for which the rank should be calculated. Defaults to all points.

**Value**

Vector containing the ranks of the first `partial` individuals or all individuals.

**Author(s)**

Olaf Mersmann <olafm@statistik.tu-dortmund.de>
normalize_points

nondominated_points  
Nondominated points

Description
Return those points which are not dominated by another point in points. This is the Pareto front approximation of the point set.

Usage
nondominated_points(points)

Arguments
points  
Matrix of points, one per column.

Value
Those points in points which are not dominated by another point.

Author(s)
Olaf Mersmann <olafm@statistik.tu-dortmund.de>

normalize_points

Scale point cloud

Description
Rescale all points to lie in the box bounded by minval and maxval.

Usage
normalize_points(points, minval, maxval)

Arguments
points  
Matrix containing points, one per column.
minval  
Optional lower limits for the new bounding box.
maxval  
Optional upper limits for the new bounding box.

Value
Scaled points.

Author(s)
Olaf Mersmann <olafm@statistik.tu-dortmund.de>
pm_control

Polynomial mutation (PM) control parameters

Description

Control parameters:

- **pm.n** Nu parameter of PM.
- **pm.p** p parameter of PM.

Usage

```
pm_control(f, upper, lower, ..., control, default = list())
```

Arguments

- **f** Multiobjective optimization function.
- **upper** Upper bounds of parameter space.
- **lower** Lower bounds of parameter space.
- **...** Further arguments passed to `f`.
- **control** List of control parameters.
- **default** List of default control parameters.

Value

The `control` list with suitably adjusted arguments. Missing control parameters are taken from `default` or, if not present there, from an internal default.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

pm_operator

Polynomial mutation operator

Description

Returns a polynomial mutation operator with the given parameters.

Usage

```
pm_operator(n, p, lower, upper)
```
sbx_control

Arguments

\begin{itemize}
\item \texttt{n} \hspace{1cm} Distance parameter mutation distribution (\(\eta\)).
\item \texttt{p} \hspace{1cm} Probability of one point mutation.
\item \texttt{lower} \hspace{1cm} Lower bounds of parameter space.
\item \texttt{upper} \hspace{1cm} Upper bounds of parameter space.
\end{itemize}

Value

Function which implements the specified mutation operator.

Author(s)

Olaf Mersmann \textless{}olafm@statistik.tu-dortmund.de\textgreater{}

\begin{verbatim}
 sbx_control
 Simulated binary crossover (SBX) control parameters

Description

sbx_control interprets the following parameters used to control the behaviour of the simulated binary crossover operator (see \texttt{sbx_operator}):

\begin{itemize}
\item \texttt{sbx.n} Nu parameter of SBX.
\item \texttt{sbx.p} $p$ parameter of SBX.
\end{itemize}

Usage

\begin{verbatim}
 sbx_control(f, upper, lower, ..., control, default = list())
\end{verbatim}

Arguments

\begin{itemize}
\item \texttt{f} \hspace{1cm} Multiobjective optimization function.
\item \texttt{upper} \hspace{1cm} Upper bounds of parameter space.
\item \texttt{lower} \hspace{1cm} Lower bounds of parameter space.
\item \texttt{...} \hspace{1cm} Further arguments passed to \texttt{f}.
\item \texttt{control} \hspace{1cm} List of control parameters.
\item \texttt{default} \hspace{1cm} List of default control parameters.
\end{itemize}

Value

The \texttt{control} list with suitably adjusted arguments. Missing control parameters are taken from \texttt{default} or, if not present there, from an internal default.

Author(s)

Olaf Mersmann \textless{}olafm@statistik.tu-dortmund.de\textgreater{}}
sbx_operator

Simulated binary crossover operator

Description

Returns a simulated binary crossover operator with the given parameters.

Usage

sbx_operator(n, p, lower, upper)

Arguments

- **n**: Distance parameter of crossover distribution ($\eta$).
- **p**: Probability of one point crossover.
- **lower**: Lower bounds of parameter space.
- **upper**: Upper bounds of parameter space.

Value

Function with one parameter $x$ which takes a matrix containing two sets of parameters and returns a matrix of two sets of parameters which resulted from the crossover operation. As with all emoa functions, the parameter sets are stored in the columns of $x$. $x$ should therefore always have two columns and a warning will be given if it has more than two columns.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

See Also

- pm_operator

steady_state_emoa_control

Steady state EMOA parameters

Description

steady_state_emoa_control interprets the following control parameters:

- **mu**: Population size.
- **maxeval**: Maximum number of function evaluations to use.
Usage

steady_state_emoa_control(f, upper, lower, ..., control, default = list())

Arguments

- `f`: Multiobjective optimization function.
- `upper`: Upper bounds of parameter space.
- `lower`: Lower bounds of parameter space.
- `...`: Further arguments passed to `f`.
- `control`: List of control parameters.
- `default`: List of default control parameters.

Value

The `control` list with suitably adjusted arguments. Missing control parameters are taken from `default` or, if not present there, from an internal default.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>

---

sympart

*Functions from the CEC 2007 EMOA competition.*

Description

Functions from the CEC 2007 EMOA competition.

Usage

sympart(x)

Arguments

- `x`: Parameter vector.

Value

Function value.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>
Functions from the CEC 2009 EMOA competition.

Description

Functions from the CEC 2009 EMOA competition.

Usage

UF1(x)
UF2(x)
UF3(x)
UF4(x)
UF5(x)
UF6(x)
UF7(x)
UF8(x)
UF9(x)
UF10(x)

Arguments

x Parameter vector.

Value

Function value.

Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>
**unary_r2_indicator**

*Unary R2 indicator*

**Description**

Unary R2 indicator

**Usage**

unary_r2_indicator(points, weights, ideal)

**Arguments**

- **points**
  Matrix of points for which to calculate the indicator value stored one per column.
- **weights**
  Matrix of weight vectors stored one per column.
- **ideal**
  Ideal point of true Pareto front. If omitted the ideal of points is used.

**Value**

Value of unary R2 indicator.

**Author(s)**

Olaf Mersmann <olafm@p-value.net>

**which_points_on_edge**

*Determine which points are on the edge of a Pareto-front approximation.*

**Description**

Determine which points are on the edge of a Pareto-front approximation.

**Usage**

which_points_on_edge(front)

**Arguments**

- **front**
  Pareto-front approximation.

**Value**

An integer vector containing the indicies of the points (columns) of front which are on the edge of the Pareto-front approximation.
Author(s)

Olaf Mersmann <olafm@statistik.tu-dortmund.de>
Index

*Topic datasets
  cec2007, 3
*Topic nonlinear
  nds_hv_selection, 10
*Topic optimize
  dominated_hypervolume, 5
  is_dominated, 10
  nds_hv_selection, 10
  nds_rank, 11
  nondominated_points, 12
  sympart, 16
  UF1, 17
*Topic package
  emoa-package, 2
  cec2007, 3
  coalesce, 4
  crowding_distance, 4
  dominated_hypervolume, 5
  emoa_package, 2
  emoa_console_logger, 6, 7
  emoa_control, 6
  emoa_logger, 6, 7
  emoa_null_logger, 7, 8
  epsilon_indicator
    (hypervolume_indicator), 8
  hypervolume_contribution, 10
  hypervolume_contribution
    (dominated_hypervolume), 5
  hypervolume_indicator, 8
  inbounds, 9
  is_dominated, 10
  is_maximally_dominated(is_dominated), 10
  nds_cd_selection (nds_hv_selection), 10
  nds_hv_selection, 6, 10
  nds_rank, 11
  nondominated_ordering(nds_rank), 11
  nondominated_points, 6, 12
  normalize_points, 12
  pm_control, 13
  pm_operator, 13, 15
  r1_indicator (hypervolume_indicator), 8
  r2_indicator (hypervolume_indicator), 8
  r3_indicator (hypervolume_indicator), 8
  sbx_control, 14
  sbx_operator, 14, 15
  steady_state_emoa_control, 15
  sympart, 16
  UF1, 17
  UF10 (UF1), 17
  UF2 (UF1), 17
  UF3 (UF1), 17
  UF4 (UF1), 17
  UF5 (UF1), 17
  UF6 (UF1), 17
  UF7 (UF1), 17
  UF8 (UF1), 17
  UF9 (UF1), 17
  unary_r2_indicator, 18
  which_points_on_edge, 18