

# Package ‘emoa’

July 2, 2014

**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

**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** 2012-09-25 11:35:13

**NeedsCompilation** yes

## R topics documented:

emoa-package . . . . .	2
cec2007 . . . . .	3
coalesce . . . . .	4
crowding_distance . . . . .	4
dominated_hypervolume . . . . .	5
emoa_console_logger . . . . .	6
emoa_control . . . . .	7

emoa_logger . . . . .	7
emoa_null_logger . . . . .	8
hypervolume_indicator . . . . .	9
inbounds . . . . .	10
is_dominated . . . . .	10
nds_hv_selection . . . . .	11
nds_rank . . . . .	12
nondominated_points . . . . .	12
normalize_points . . . . .	13
pm_control . . . . .	13
pm_operator . . . . .	14
sbx_control . . . . .	15
sbx_operator . . . . .	15
steady_state_emoa_control . . . . .	16
sympart . . . . .	17
UF1 . . . . .	17
unary_r2_indicator . . . . .	18
which_points_on_edge . . . . .	19

<b>Index</b>	<b>20</b>
--------------	-----------

---

emoa-package

*The EMOA package*

---

## Description

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.

## Details

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 Nagarathnam Suganthany
- Santosh Tiwar

- Qingfu Zhang
- Aimin Zhou
- Shizheng Zhaoy

**Author(s)**

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

---

cec2007

*CEC 2007 multiobjective optimization competition results*

---

**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**

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

## End(Not run)
```

---

coalesce	<i>Return first non null argument.</i>
----------	--

---

**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	<i>Crowding Distance</i>
-------------------	--------------------------

---

**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 *Dominated Hypervolume calculation*

---

**Description**

dominated\_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

Carlos M. Fonseca, Luis Paquete, and Manuel Lopez-Ibanez. *An improved dimension-sweep algorithm for the hypervolume indicator*. In IEEE Congress on Evolutionary Computation, pages 1157-1163, Vancouver, Canada, July 2006.

## See Also

[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

```
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

```
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.  
...                passed to the parent logger factory.

**Value**

An emoa\_logger object.

**See Also**

[emoa\\_console\\_logger](#) and [emoa\\_null\\_logger](#) for convinience wrappers around emoa\_logger providing useful defaults.

---

emoa\_null\_logger            *null logger*

---

**Description**

Logger object that discards all log events.

**Usage**

```
emoa_null_logger(...)
```

**Arguments**

...                ignored.

**Value**

An emoa\_logger object.



---

hypervolume\_indicator *Binary quality indicators*

---

**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**

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

**Value**

Value of the quality indicator.

**Author(s)**

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

**References**

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., and Grunert da Fonseca, V (2003): Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2), 117-132.

---

inbounds	<i>Clip value to a given range</i>
----------	------------------------------------

---

**Description**

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

**Usage**

```
inbounds(x, l, 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	<i>Pareto dominance checks.</i>
--------------	---------------------------------

---

**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**

```
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**

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

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

**Arguments**

<code>values</code>	Matrix of function values.
<code>n</code>	Number of individuals to select for replacement.
<code>...</code>	Optional parameters passed to <a href="#">hypervolume_contribution</a> .

**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**

```
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>

---

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)
```

**Arguments**

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

**Value**

Function which implements the specified mutation operator.

**Author(s)**

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

---

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 [sbx\\_operator](#)):

**sbx.n** Nu parameter of SBX.

**sbx.p** \$p\$ parameter of SBX.

### Usage

```
sbx_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>

---

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>

---

UF1

*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                    Parmater vector.

**Value**

Function value.

**Author(s)**

Olaf Mersmann &lt;olafm@statistik.tu-dortmund.de&gt;

---

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 indices 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, 11
- \*Topic **optimize**
  - dominated\_hypervolume, 5
  - is\_dominated, 10
  - nds\_hv\_selection, 11
  - nds\_rank, 12
  - nondominated\_points, 12
  - sympart, 17
  - UF1, 17
- \*Topic **package**
  - emoa-package, 2

cec2007, 3  
coalesce, 4  
crowding\_distance, 4

dominated\_hypervolume, 5

emoa-package, 2  
emoa\_console\_logger, 6, 8  
emoa\_control, 7  
emoa\_logger, 6, 7  
emoa\_null\_logger, 8, 8  
epsilon\_indicator  
    (hypervolume\_indicator), 9

hypervolume\_contribution, 11  
hypervolume\_contribution  
    (dominated\_hypervolume), 5  
hypervolume\_indicator, 9

inbounds, 10  
is\_dominated, 10  
is\_maximally\_dominated (is\_dominated),  
    10

nds\_cd\_selection (nds\_hv\_selection), 11  
nds\_hv\_selection, 6, 11

nds\_rank, 12  
nondominated\_ordering (nds\_rank), 12  
nondominated\_points, 6, 12  
normalize\_points, 13

pm\_control, 13  
pm\_operator, 14, 16

r1\_indicator (hypervolume\_indicator), 9  
r2\_indicator (hypervolume\_indicator), 9  
r3\_indicator (hypervolume\_indicator), 9

sbx\_control, 15  
sbx\_operator, 15, 15  
steady\_state\_emoa\_control, 16  
sympart, 17

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, 19