# Package: smaa (via r-universe)

October 20, 2026

**Version** 0.3-4

**Date** 2026-08-01

**Title** Stochastic Multi-Criteria Acceptability Analysis

**Depends** R (>= 3.5.0)

**Suggests** hitandrun

**Imports** graphics

**Description** Implementation of the Stochastic Multi-Criteria Acceptability Analysis (SMAA) family of Multiple Criteria Decision Analysis (MCDA) methods. Tervonen, T. and Figueira, J. R. (2008) <doi:10.1002/mcda.407>.

**URL** https://github.com/gertvv/rsmaa

**License** GPL-3

**NeedsCompilation** yes

**Author** Gert van Valkenhoef [aut, cre, cph]

**Maintainer** Gert van Valkenhoef <gert@gertvv.nl>

**Repository** https://cran.r-universe.dev

**Date/Publication** 2026-08-21 06:40:02 UTC

**RemoteUrl** https://github.com/cran/smaa

**RemoteRef** HEAD

**RemoteSha** 6d7d92b8d314943063c7b0a2a7df62b34f8f437f

## Contents

smaa-package . . . . . 2
smaa . . . . . 3
smaa.cf . . . . . 4
smaa.cw . . . . . 5
smaa.entropy . . . . . 6
smaa.pvf . . . . . 8
smaa.pwi . . . . . 9

smaa.ra . . . . . 10
smaa.ranks . . . . . 11
smaa.values . . . . . 12

**Index** **13**

---

smaa-package *SMAA: Stochastic Multi-criteria Acceptability Analysis*

---

## Description

This R package implements the Stochastic Multi-criteria Acceptability Analysis (SMAA) family of methods for stochastic Multiple Criteria Decision Analysis (MCDA). In SMAA methods, uncertainty in criteria measurements and incomplete information on the weights are accounted for by Monte Carlo integration of probability distributions.

## Details

The `smaa` function implements the core Monte Carlo integration method. It calculates the SMAA decision metrics (rank acceptabilities and central weights) in one go.

Alternatively, the `smaa.values`, `smaa.ranks`, `smaa.ra`, and `smaa.cw` perform the individual steps. Note that `smaa` is slightly more efficient because it does not store the alternatives' values or rankings.

The hitandrun-package is complementary to this package in that it provides methods for sampling weights when incomplete preference information is available in the form of linear constraints on the weight vector.

## Author(s)

Gert van Valkenhoef

## References

T. Tervonen and J.R. Figueira (2008), *A survey on stochastic multicriteria acceptability analysis methods*, Journal of Multi-Criteria Decision Analysis 15(1-2):1-14. [doi:10.1002/mcda.407]

T. Tervonen, G. van Valkenhoef, N. Basturk, and D. Postmus (2012), *Hit-And-Run enables efficient weight generation for simulation-based multiple criteria decision analysis*, European Journal of Operational Research 224(3):552-559. [doi:10.1016/j.ejor.2012.08.026]

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate SMAA metrics (one-stage)
result <- smaa(meas, pref)
print(result)
```

```
# Calculate SMAA metrics (multi-stage)
values <- smaa.values(meas, pref)
summary(values)
ranks <- smaa.ranks(values)
smaa.ra(ranks)
smaa.entropy.ranking(ranks)
smaa.cw(ranks, pref)

# Calculate confidence factors
smaa.cf(meas, result$cw)
```

---

`smaa` *One-stage SMAA analysis*

---

## Description

Calculate SMAA decision indices based on a set of samples from the criteria values distribution and a set of samples from the feasible weight space.

## Usage

```
smaa(meas, pref)
```

## Arguments

| | |
|---|---|
| `meas` | Criteria measurements. An $N \times m \times n$ array, where `meas[i,,]` is a matrix where the m alternatives are the rows and the n criteria the columns. The values must be standardized measurements (i.e. after application of the partial value function). `smaa.pvf` provides a convenience method to standardize partial values. |
| `pref` | Weights. An $N \times n$ array, where `pref[i,]` is a normalized weight vector. |

## Details

The one-stage method does not store the alternatives' values or the raw rankings. Instead, only standard summary metrics are provided.

## Value

| | |
|---|---|
| `ra` | Rank acceptabilities (see `smaa.ra`). |
| `cw` | Central weights (see `smaa.cw`). |

## Author(s)

Gert van Valkenhoef

## See Also

`smaa.pvf`

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))

# Read weights from file
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))
# Alternatively, sample weights using hitandrun:
# library(hitandrun)
# pref <- simplex.sample(n, N)$samples

# Calculate SMAA metrics
result <- smaa(meas, pref)
print(result)
plot(result)

result <- smaa(meas, c(0.5, 0.2, 0.3))
print(result)
```

---

# smaa.cf *SMAA confidence factors*

---

## Description

Calculate SMAA confidence factors of the central weights.

## Usage

```
smaa.cf(meas, cw)
```

## Arguments

| | |
|---|---|
| meas | Criteria measurements. An $N \times m \times n$ array, where `meas[i,,]` is a matrix where the m alternatives are the rows and the n criteria the columns. The values must be standardized measurements (i.e. after application of the partial value function). |
| cw | An $m \times n$ matrix of central weights, where each row corresponds to an alternative and each column to a criterion. |

## Details

The confidence factor for an alternative is its first-rank acceptability under its central weight.

## Value

An object of class `smaa.cf`, with the following elements:

| | |
|---|---|
| cf | A vector of confidence factors, one for each alternative. |
| cw | The central weights (see `smaa.cw`). |

The number of SMAA iterations is stored in `attr(x, "smaa.N")`.

### Author(s)

Gert van Valkenhoef

### See Also

smaa.cw

### Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate central weights
values <- smaa.values(meas, pref)
ranks <- smaa.ranks(values)
cw <- smaa.cw(ranks, pref)
print(cw)
cf <- smaa.cf(meas, cw)
print(cf)
plot(cf)
```

---

## smaa.cw *SMAA central weights*

---

### Description

Calculate SMAA central weights from sampled rankings and the corresponding weights.

### Usage

```
smaa.cw(ranks, pref)
```

### Arguments

| | |
|---|---|
| ranks | An $N \times m$ array of sampled rankings, where $N$ is the number of SMAA iterations and $m$ is the number of alternatives. |
| pref | An $N \times n$ array of sampled rankings, where $N$ is the number of SMAA iterations and $n$ is the number of alternatives. |

### Value

An $m \times n$ matrix of central weights, where each row corresponds to an alternative and each column to a criterion. The number of SMAA iterations is stored in `attr(x, "smaa.N")`.

### Note

The value is given class `smaa.cw`, use `unclass(x)` to treat it as a regular matrix.

## Author(s)

Gert van Valkenhoef

## See Also

smaa.ranks

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate central weights
values <- smaa.values(meas, pref)
ranks <- smaa.ranks(values)
cw <- smaa.cw(ranks, pref)
print(cw)
plot(cw)
```

---

# smaa.entropy *Decision entropy*

---

## Description

Calculate decision entropy from the sampled SMAA rankings. For both ranking and choice problematics.

## Usage

```
smaa.entropy.ranking(ranks, p0 = 1)
smaa.entropy.choice(ra, p0 = 1)
```

## Arguments

| | |
|---|---|
| ranks | Object of class smaa.ranks containing sampled SMAA rankings. |
| ra | Object of class smaa.ra containing SMAA rank acceptabilities. Alternatively, an object of class smaa.ranks from which the rank acceptabilities will be calculated. |
| p0 | Baseline probability for the entropy calculation. |

## Details

Calculates the entropy for the given problematic, quantifying either the uncertainty in the ranking of the alternatives (where the outcome space $Y$ consists of the $m!$ possible rankings) or in the choice of the best alternative (where the outcome space $Y$ consists of the $m$ alternatives). The entropy is given by:

$$H(Y|W) = -\sum_{y \in Y} p_0 p(y|W) \log p_0 p(y|W)$$

where $W$ is the space of feasible weights. Since the SMAA analysis samples from the outcome space, the $p(y|W)$ can be estimated directly from the given sample.

## Value

The entropy (a single numeric value).

## Note

The number of samples needed to accurately estimate $H(Y|W)$ for the ranking problematic is currently unknown.

## Author(s)

Gert van Valkenhoef

## References

G. van Valkenhoef and T. Tervonen, *Optimal weight constraint elicitation for additive multi-attribute utility models*, presentation at EURO 2013, July 2013, Rome, Italy.

## See Also

smaa.ranks smaa.ra

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate ranks
values <- smaa.values(meas, pref)
ranks <- smaa.ranks(values)

# Calculate ranking entropy
smaa.entropy.ranking(ranks)

# Calculate choice entropy
# (equal to ranking entropy because there are only two alternatives)
smaa.entropy.choice(ranks)
smaa.entropy.choice(smaa.ra(ranks))
```

`smaa.pvf` *Compute piece-wise linear partial value functions*

## Description

Given a set of reference levels and their values, compute a linearly interpolated (piece-wise linear) partial value function.

## Usage

```
smaa.pvf(x, cutoffs, values, outOfBounds="error")
```

## Arguments

| | |
|---|---|
| `x` | Values to compute the PVF for, a numeric vector |
| `cutoffs` | Reference levels (ascending order) |
| `values` | Values of the reference levels |
| `outOfBounds` | What to do when some of the `x` are outside the range of the given `cutoffs`. When `"error"`, throws an error. When `"clip"`, clips to the value of the first or last cutoff. When `"interpolate"`, interpolates according to the closest interval. |

## Value

A numeric vector the same length as `x`.

The values are computed by linear interpolation between the values of the two closest reference levels. This has been implemented in C for a dramatic performance improvement.

## Author(s)

Gert van Valkenhoef

## Examples

```
x <- c(50, 90, 100, 10, 40, 101, 120)
values <- smaa.pvf(x,
  cutoffs=c(50, 75, 90, 100),
  values=c(1, 0.8, 0.5, 0),
  outOfBounds="clip")
stopifnot(all.equal(values, c(1.0, 0.5, 0.0, 1.0, 1.0, 0.0, 0.0)))
```

smaa.pwi *SMAA pair-wise winning indices*

## Description

Calculate SMAA pair-wise winning indices from sampled rankings.

## Usage

```
smaa.pwi(ranks)
```

## Arguments

| | |
|---|---|
| ranks | An $N \times m$ array of sampled rankings, where $N$ is the number of SMAA iterations and $m$ is the number of alternatives. |

## Value

An $m \times m$ matrix of pair-wise winning indices. The index at (i, j) describes the share of samples for which alternative i has a better (lower) rank than alternative j.

## Author(s)

Tommi Tervonen

## See Also

smaa.ranks

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate pair-wise winning indices
values <- smaa.values(meas, pref)
ranks <- smaa.ranks(values)
pwi <- smaa.pwi(ranks)
print(pwi)
```

`smaa.ra` *SMAA rank acceptabilities*

## Description

Calculate SMAA rank acceptabilities from sampled rankings.

## Usage

```
smaa.ra(ranks)
```

## Arguments

| | |
|---|---|
| `ranks` | An $N \times m$ array of sampled rankings, where $N$ is the number of SMAA iterations and $m$ is the number of alternatives. |

## Value

An $m \times m$ matrix of rank probabilities, where each row corresponds to an alternative. The number of SMAA iterations is stored in `attr(x, "smaa.N")`.

## Note

The value is given class `smaa.ra`, use `unclass(x)` to treat it as a regular matrix.

## Author(s)

Gert van Valkenhoef

## See Also

`smaa.ranks`

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate rank acceptabilities
values <- smaa.values(meas, pref)
ranks <- smaa.ranks(values)
ra <- smaa.ra(ranks)
print(ra)
plot(ra)
```

smaa.ranks *SMAA ranking*

## Description

Calculate SMAA ranks based on the sampled alternatives' values.

## Usage

```
smaa.ranks(values)
```

## Arguments

| | |
|---|---|
| values | An $N \times m$ array of sampled alternative values, where $N$ is the number of SMAA iterations and $m$ is the number of alternatives. |

## Value

An $N \times m$ array of ranks obtained by each alternative in each iteration.

## Note

The value is given class `smaa.ranks`, use `unclass(x)` to treat it as a regular matrix.

## Author(s)

Gert van Valkenhoef

## See Also

`smaa.values` `smaa.ra` `smaa.cw`

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))

# Calculate alternative ranks
values <- smaa.values(meas, pref)
ranks <- smaa.ranks(values)
summary(ranks)

ranks.expected <- readRDS(system.file("extdata/thrombo-ranks-nopref.rds", package="smaa"))
stopifnot(all.equal(ranks, ranks.expected))
```

`smaa.values` *SMAA alternative values*

## Description

Calculate the alternative values based on a set of samples from the criteria values distribution and a set of samples from the feasible weight space.

## Usage

```
smaa.values(meas, pref)
```

## Arguments

| | |
|---|---|
| `meas` | Criteria measurements. An $N \times m \times n$ array, where `meas[i,,]` is a matrix where the m alternatives are the rows and the n criteria the columns. The values must be standardized measurements (i.e. after application of the partial value function). `smaa.pvf` provides a convenience method to standardize partial values. |
| `pref` | Weights. An $N \times n$ array, where `pref[i,]` is a normalized weight vector. |

## Value

An $N \times m$ array of alternative values.

## Author(s)

Gert van Valkenhoef

## See Also

`smaa.pvf` `smaa.ranks`

## Examples

```
N <- 1E4; m <- 2; n <- 3
meas <- readRDS(system.file("extdata/thrombo-meas.rds", package="smaa"))

# Read weights from file
pref <- readRDS(system.file("extdata/thrombo-weights-nopref.rds", package="smaa"))
# Alternatively, sample weights using hitandrun:
# library(hitandrun)
# pref <- simplex.sample(n, N)$samples

# Calculate alternative values
values <- smaa.values(meas, pref)
summary(values)
plot(values)

values.expected <- readRDS(system.file("extdata/thrombo-values-nopref.rds", package="smaa"))
stopifnot(all.equal(values, values.expected))
```

# Index

∗ **SMAA**
smaa, 3

hitandrun-package, *2*

smaa, *2*, 3
smaa-package, 2
smaa.cf, 4
smaa.cw, *2–4*, 5, *5*, *11*
smaa.entropy, 6
smaa.pvf, *3*, 8, *12*
smaa.pwi, 9
smaa.ra, *2*, *3*, *7*, 10, *11*
smaa.ranks, *2*, *6*, *7*, *9*, *10*, 11, *12*
smaa.values, *2*, *11*, 12