Package ‘Rfmtool’

June 3, 2024

Type Package

Version 5.0.4

Date 2024-05-31

Title Fuzzy Measure Tools

Author Gleb Beliakov [aut, cre],
Quan Vu [ctb],
Andrei Kelarev [ctb],
Michel Berkelaar [ctb],
Kjell Eikland [ctb],
Samuel E. Buttrey [ctb],
Stefan I. Larimore [ctb],
Timothy A. Davis [ctb],
John Gilbert [ctb],
Esmond Ng [ctb],
Peter Notebaert [ctb],
Richard Stallman [ctb],
Jeroen Dirks [ctb],
Daniela L. Calderon [ctb]

Maintainer Gleb Beliakov <gleb@deakin.edu.au>

Depends R (>= 2.9.2), Rcpp

LinkingTo Rcpp

Suggests

Description Various tools for handling fuzzy measures, calculating Shapley value and interaction index, Choquet and Sugeno integrals, as well as fitting fuzzy measures to empirical data are provided. Construction of fuzzy measures from empirical data is done by solving a linear programming problem by using 'lpsolve' package, whose source in C adapted to the R environment is included. The description of the basic theory of fuzzy measures is in the manual in the Doc folder in this package. Please refer to the following:

[1] <https://personal-sites.deakin.edu.au/~gleb/fmtools.html>

[2] G. Beliakov, H. Bustince, T. Calvo, 'A Practical Guide to Averaging', Springer, (2016, ISBN: 978-3-319-24753-3).

[3] G. Beliakov, S. James, J-Z. Wu, 'Discrete Fuzzy Measures', Springer, (2020, ISBN: 978-3-030-15305-2).

License LGPL-3

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-06-02 23:20:20 UTC

R topics documented:

`fm` *Rfmtool package*

Description

This function shows a list of function included in this toolbox

Usage

```
fm()
```

Details

The following functions involve the parameters v (the array containing the fuzzy measure in standard representation) or Mob (in Mobius representation), n - the dimension and m = 2^n. The values of the fuzzy measure always obey the binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
fm()
```

`fm.add_pair_sparse` *Function for adding a pair to the sparse fuzzy measure*

Description

This is used for populating capacities which Add a pair v_ij to the structure, their Indices are 1-based.

Usage

```
fm.add_pair_sparse( i, j, v, envsp = NULL)
```

Arguments

`i`	One of the indices which are 1-based
`j`	One of the indices which are 1-based
`v`	The value to be added.
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n)

Value

`output` The output is an added pair v_ij to the structure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
  n <- 3
  tups<-vector()
  tupsidx<-vector()
  envsp <- fm.PrepareSparseFM(n, tups,tupsidx)
   envsp <-fm.add_pair_sparse(1,2, 0.4, envsp)
   envsp <-fm.add_pair_sparse(1,3, 0.3, envsp)
envsp
     envsp <- fm.FreeSparseFM(envsp)
```

`fm.add_singletons_sparse` *Function for adding singletons to the sparse fuzzy measure*

Description

This is used for adding singletons to the structure.

Usage

```
fm.add_singletons_sparse(v, envsp=NULL)
```

Arguments

`v`	The vector of singletons of size n.
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output` The output is added singletons to the structure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
tups<-vector()
tupsidx<-vector()
envsp <- fm.PrepareSparseFM(n, tups,tupsidx)

envsp <- fm.add_singletons_sparse(c(0, 0.3, 0.5),envsp)
```

---

fm.add_tuple_sparse *Function for adding singletons to the sparse fuzzy measure*

Description

This is used for populating capacities which Add a tuple of size tupsize to the structure whose Indices are 1-based in tuple.

For populating capacities, adds a whose 1-based indices are in tuple

Usage

```
fm.add_tuple_sparse( tuple, v, envsp=NULL)
```

Arguments

tuple	Collection of objects. It is a list of cardinalities of the nonzero tuples (cardinality, tuple composition)
v	The value of the tuple to be added
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is adding a tuple of size tupsize

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
  n <- 4
tups<-vector()
tupsidx<-vector()
envsp <- fm.PrepareSparseFM(n, tups,tupsidx)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.2,envsp)
envsp <- fm.add_tuple_sparse(c(1,3,4),0.3,envsp)
```

fm.Banzhaf — *Banzhaf value computation function*

Description

Calculates the Banzhaf indices of input criteria from general fuzzy measure.

Usage

```
fm.Banzhaf(v,env=NULL)
```

Arguments

`v`	Fuzzy measure in general representation.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is an array of size n, which contain Banzhaf indices of input criteria.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Banzhaf(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)

env<-fm.Free(env)
```

fm.Banzhaf2addMob — *Function for calculating Banzhaf values of 2-additive fuzzy measure in Mobius representation*

Description

Calculate the Banzhaf values of a 2-additive fuzzy measure for n inputs given in Mobius representation. The results are in arrays.

Usage

```
fm.Banzhaf2addMob(n, Mob)
```

Arguments

n	Number of inputs
Mob	Fuzzy measure value in Mobius representation

Value

output	The output is an array of size n, which contain Banzhaf indices of input criteria.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
Banzhaf  <- fm.Banzhaf2addMob(3, c(0.2, 0.3, 0.5, -0.2, 0.4, 0.1))
```

---

fm.BanzhafMob — *Banzhaf value computation function in Mobius representation*

Description

Calculates the Banzhaf indices of input criteria from general fuzzy measure in Mobius representation.

Usage

```
fm.BanzhafMob(Mob,env=NULL)
```

Arguments

Mob	Fuzzy measure in Mobius representation.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is an array of size n, which contain Banzhaf indices of input criteria.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.BanzhafMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
env<-fm.Free(env)
```

fm.BanzhafMob_sparse *Banzhaf values computation function in sparse representation*

Description

Calculates Banzhaf values vectors of size n of a sparse fuzzy measure

Usage

```
fm.BanzhafMob_sparse(n, envsp=NULL)
```

Arguments

n	The size of values vectors
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is Banzhaf values vectors of size n of a sparse fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
  n <- 3
tups<-vector()
tupsidx<-vector()
envsp <- fm.PrepareSparseFM(n, tups,tupsidx)
  envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
  envsp <- fm.add_pair_sparse(1,2,0.4,envsp);

  fm.BanzhafMob_sparse(3, envsp)

  envsp <- fm.FreeSparseFM(envsp)
```

fm.Bipartition — *Bipartition interaction index computation function*

Description

Calculates the Bipartition interaction indices of input criteria from general fuzzy measure.

Usage

```
fm.Bipartition(v,env=NULL)
```

Arguments

`v`	Fuzzy measure in general representation.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is an array of size 2^n, which contain bipartition interaction indices of input criteria coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Bipartition(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
env<-fm.Free(env)
```

fm.BipartitionBanzhaf — *Bipartition Banhzaf interaction index computation function*

Description

Calculates the Banzhaf Bipartition interaction indices of input criteria from general fuzzy measure.

Usage

```
fm.BipartitionBanzhaf(v,env=NULL)
```

Arguments

`v`	Fuzzy measure in general representation.
`env`	Environment variable obtained from fm.Init(n).

Value

output	The output is an array of size 2^n, which contain Banzhaf bipartition interaction indices of input criteria coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.BipartitionBanzhaf(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
env<-fm.Free(env)
```

---

fm.check_convexity_monotonicity_mob

Function for checking supermodularity of the set function v in Mobius representation

---

Description

Checks supermodularity of the set function v in Mobius representation using stan- dard check.

Usage

```
fm.check_convexity_monotonicity_mob(v, len, env=NULL)
```

Arguments

v	matrix v stores fuzzy measurements consecutively in cardinal order v.
len	this is the length of array Mob (this array is usually smaller than 2^n), and is computed by fm.fm_arraysize_kadd(N, Kadd).
env	Environment variable obtained from fm.Init(n).

Value

output	The output is 1 or 0 to check for monotonicity.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
step <- 0.001
Fn <- NULL
Option<- 3
fuzzymeasures <- fm.generate_fm_randomwalk(1, 3, 2, 1000, Option, step, Fn, env)
len <- fuzzymeasures$length
check <- fm.check_convexity_monotonicity_mob(fuzzymeasures$V, len, env)
```

---

fm.check_monotonicity *Function for checking monotonicity of the set function v*

Description

Checks monotonicity of the set function v in standard representation using insert sort.

Usage

```
fm.check_monotonicity(v, env=NULL)
```

Arguments

v	matrix v stores fuzzy measurements consecutively in cardinal order.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is 1 or 0 to check for monotonicity.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
v <- fm.generate_fm_sorting(1, 1000, 0, env)
monotonicity <- fm.check_monotonicity(v, env)
```

fm.check_monotonicity_mob

Function for checking monotonicity of the set function v in Mobius representation.

Description

Checks monotonicity of the set function v in Mobius representation using standard check.

Usage

```
fm.check_monotonicity_mob(v, len, env=NULL)
```

Arguments

`v`	matrix v stores fuzzy measurements consecutively in cardinal order.
`len`	this is the length of array Mob (this array is usually smaller than 2^n), and is computed by fm_arraysize_kadd(N, Kadd)
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 or 0 to check for monotonicity

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
step <- 100
Fn <- NULL
Option<- 3
fuzzymeasures <- fm.generate_fm_randomwalk(1, 3, 2, 1000, Option, step, Fn, env)
len <- fuzzymeasures$length
check <- fm.check_monotonicity_mob(fuzzymeasures$V, len, env)
check
```

fm.check_monotonicity_mob_2additive

Function for checking the monotonicity of the 2-additive set function v in Mobius representation.

Description

Check the monotonicity of the 2-additive set function v in Mobius representation using fast check.

Usage

```
fm.check_monotonicity_mob_2additive(v, n, temp=NULL)
```

Arguments

v	Random 2-additive fuzzy measure in Mobius representation.
n	Number of inputs
temp	Auxiliary array of length n^2 (e.g: array(0.0,n*n)). It may or may not be specified (if speed matters, then preallocate it).

Value

output	The output is 1 or 0 to check for monotonicity.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
v <- fm.generate_fm_2additive(1, 10)
n <- 10
v$len
v$V
check <- fm.check_monotonicity_mob_2additive(v$V, n)
check

temp <- array(0.0,10*10);
check <- fm.check_monotonicity_mob_2additive(v$V, n, temp)
check
```

fm.check_monotonicity_sort_insert

Function for checking monotonicity of the set function v

Description

Checks monotonicity of the set function v in standard representation using insert sort.

Usage

```
fm.check_monotonicity_sort_insert(v, indices, env=NULL)
```

Arguments

`v`	matrix v stores fuzzy measurements consecutively in cardinal order.
`indices`	The indices can be used at subsequent steps of monotonicity verification. This function is called after merge sort, so the indices are already precomputed.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is a list of components (True/False, indices, values). The indices and values can be used at subsequent steps of monotonicity verification (e.g., values slightly perturbed)

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
v <- fm.generate_fm_sorting(1, 1000, 0, env)
 out <- fm.check_monotonicity_sort_merge(v, NULL, env)
 out$V[1] = out$V[1] *1.1
 out<- fm.check_monotonicity_sort_insert(out$V, out$index, env)
 out$out
```

`fm.check_monotonicity_sort_merge`
Function for checking monotonicity of the set function v

Description

Checks monotonicity of the set function v in standard representation using merge sort.

Usage

```
fm.check_monotonicity_sort_merge(v, indices=NULL, env=NULL)
```

Arguments

`v`	matrix v stores fuzzy measurements consecutively in cardinal order.
`indices`	The indices can be used at subsequent steps of monotonicity verification. Initially indices need not be specified
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is a list of components (True/False, indices, values). The indices and values can be used at subsequent steps of monotonicity verification (e.g., values slightly perturbed)

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
v <- fm.generate_fm_sorting(1, 1000, 0, env)
v
measure <- fm.check_monotonicity_sort_merge(v,NULL, env)

print(measure$out)

  measure$V[1] = measure$V[1] *1.1
  measure <- fm.check_monotonicity_sort_merge(measure$V, measure$index, env)
```

fm.Choquet — *Choquet integral computation function*

Description

Calculates the value of a discrete Choquet integral of input x, with fuzzy measure in general representation.

Usage

```
fm.Choquet(x, v, env=NULL)
```

Arguments

x	Input vector of size n, containing utility value of input criteria. x is in [0,1].
v	The general fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The ouput is a single value of the computed Choquet integral.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Choquet(c(0.6, 0.3, 0.8), c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
env<-fm.Free(env)
```

fm.Choquet2addMob — *Function for calculating Choquet integral value for 2-additive fuzzy measure in Mobius representation*

Description

Calculates the Choquet integral value of a 2-additive fuzzy measure for n inputs given in Mobius representation.

Usage

```
fm.Choquet2addMob(n, x, Mob)
```

Arguments

`n`	Number of inputs
`x`	Input vector of size n, containing utility value of input criteria. x is in [0,1].
`Mob`	The Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.

Value

`output`	The output is the Choquet integral value in Mobius representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
Choquet <- fm.Choquet2addMob(3, c(0.2,0.5,0.4), c(0.2, 0.3, 0.5, -0.2, 0.4, 0.1))
```

`fm.ChoquetCoMobKInter` *function for calculating Choquet integral value with respect to dual k-interactive fuzzy measure in Mobius representation*

Description

Calculates the Choquet integral of x with respect to dual k-interactive fuzzy measure in Mobius representation.

Usage

```
fm.ChoquetCoMobKInter(x, Mob, kadd, env=NULL)
```

Arguments

`x`	Input vector of size n, containing utility value of input criteria. x is in [0,1].
`Mob`	The Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data
`kadd`	is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n. kadd is k in k-additive f-measure, 1 < kadd < n+1; if kdd=n - f.m. is unrestricted.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is the Choquet integral value in Mobius representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env <-fm.Init(4)
step <- 0.0001
Fn <- NULL
fuzzymeasures <- fm.generate_fm_kinteractivedualconvex(1, 4, 2, 1000, step, Fn, env)
fuzzymeasures
env

fm.ChoquetCoMobKInter(c(0.2,0.5,0.4,0.1), fuzzymeasures$V, 2, env)
env<-fm.Free(env)
```

---

fm.ChoquetKinter — *Choquet integral value computation function in standard representation wrt k-interactive fuzzy measure*

Description

This is an alternative calculation of the Choquet integral from the fuzzy measure in Mobius representation.

Usage

```
fm.ChoquetKinter(x, v, kint, env)
```

Arguments

x	Input vector of size n, containing utility value of input criteria. x is in [0,1].
v	The fuzzy measure of size less than m=2^n. Its values can be provided by users, or by estimating from empirical data.
kint	the k-interactivity parameter, must be smaller than n.
env	Environment variable obtained from fm.Init(n).

Value

output	The ouput is a single value of the computed Choquet integral.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.ChoquetKinter(c(0.6,0.3,0.8),c(0,0.3,0.5,0.6,0.4,0.8,0.7,1),2,env)
env<-fm.Free(env)
```

`fm.ChoquetMob` *Choquet integral value computation function in Mobius representation*

Description

This is an alternative calculation of the Choquet integral from the fuzzy measure in Mobius representation.

Usage

```
fm.ChoquetMob(x, Mob, env=NULL)
```

Arguments

x	Input vector of size n, containing utility value of input criteria. x is in [0,1].
Mob	The Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The ouput is a single value of the computed Choquet integral.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.ChoquetMob(c(0.2,0.5,0.4), c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.ChoquetMob_sparse *Choquet integral computation function in sparse representation*

Description

Calculates the Choquet integral in Mobius sparse representation.

Usage

```
fm.ChoquetMob_sparse(x, envsp=NULL)
```

Arguments

`x`	Input vector of size n, containing utility value of input criteria. x is in [0,1].
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output`	The output is the Choquet integral in Mobius sparse representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_pair_sparse(1,2,0.4,envsp);

 ChoquetMobsparse <- fm.ChoquetMob_sparse(c(0.1,0.05,0.2),envsp)
 ChoquetMobsparse
 envsp <- fm.FreeSparseFM(envsp)
```

fm.ConstructLambdaMeasure *Function for Constructing Lambda*

Description

Finds the value of lambda and calculates the rest of the values of the fuzzy measure, given its values at singletons; singletons is an array of size n. The outputs are lambda and v, v is in standard representation and binary ordering.

Usage

```
fm.ConstructLambdaMeasure(singletons,env)
```

Arguments

`singletons`	Singletons is an array of n.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The ouput is the list (lambda, measure), where measure is a fuzzy measure in standard representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
w <- fm.ConstructLambdaMeasure(c(0, 0.3, 0.5),env)
```

`fm.ConstructLambdaMeasureMob`

Function for Constructing Lambda in Mobius representation

Description

Finds the value of lambda and calculates the rest of the values of the fuzzy measure, given its values at singletons; singletons is an array of size n. The outputs are lambda and measure, measure is in Mobius representation.

Usage

```
fm.ConstructLambdaMeasureMob(singletons,env)
```

Arguments

`singletons`	Singletons is an array of n.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The ouput is the list (lambda, measure), where measure is a fuzzy measure in Mobius representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
w <- fm.ConstructLambdaMeasureMob(c(0, 0.3, 0.5),env)
w$measure
fm.Free(env)
```

---

fm.ConvertCoMob2Kinter

Function for dual k-interactive fuzzy measure from Mobius to standard representation

---

Description

Converts dual k-interactive fuzzy measure from Mobius to standard representation.

Usage

```
fm.ConvertCoMob2Kinter(Mob,kadd, fullmu, env=NULL)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from emperical data.
`kadd`	is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n. kadd is k in k-additive f-measure, 1 < kadd < n+1; if kdd=n - f.m. is unrestricted.
`fullmu`	Integer flag. is 1 then all 2n are allocated, otherwise a more compact representation fo rk-interactive fuzzy measures is used.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is k-interactive fuzzy measure standard representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env <-fm.Init(4)
fullmu <- 0
step<-0.001
Fn <- NULL
        fuzzymeasures <- fm.generate_fm_kinteractivedualconvex(1, 4, 2, 1000,
        step, Fn, env)

fm.ConvertCoMob2Kinter(fuzzymeasures$V, 2, fullmu, env )
```

`fm.dualm` *Function for calculating dual of fuzzy measure*

Description

Calculates the dual of fuzzy measure v, returns it as value of the function (array of size m).

Usage

```
fm.dualm(v, env=NULL)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from emperical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The ouput is an array of size m with the dual of fuzzy measure v.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
w <- fm.dualm(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.dualmMob *Dualm computation function in Mobius representation*

Description

Calculates the dual of fuzzy measure v, returns it as value of the function (array of size m).

Usage

```
fm.dualmMob(Mob,env=NULL)
```

Arguments

Mob	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from emperical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The ouput is an array of size m with the dual of fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
w <- fm.dualmMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.dualMobKadd *Function for calculating dual of k-additive fuzzy measure in Mobius representation*

Description

Calculates the dual of a k-additive fuzzy measures for n inputs.

Usage

```
fm.dualMobKadd(Mob, env = NULL, kadd = "NA")
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).
`kadd`	Value of k-interactivity, which is used for reducing the complexity of fuzzy measures. It is defined as an optional argument

Value

`output`	The output is the dual of a k-additive fuzzy measures for n inputs

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
  dualMob_Kadd <- fm.dualMobKadd(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1), env,2)
```

`fm.EntropyChoquet` *Entropy of fuzzy measure*

Description

Calculates entropy value of the Choquet integral for the fuzzy measure v in general representation

Usage

```
fm.EntropyChoquet(v,env)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The ouput is the entropy value of the Choquet integral for the fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.EntropyChoquet(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1))
```

`fm.EntropyChoquetMob` *Entropy Choquet computation function in Mobius representation*

Description

Calculates entropy value of the Choquet integral for the fuzzy measure v in Mobius representation

Usage

```
fm.EntropyChoquetMob(Mob,env)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The ouput is entropy value of the Choquet integral for the fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.EntropyChoquetMob(c(0.0,0.3,0.5,-0.2,0.4,0.1,-0.2,0.1),env)
```

fm.errorcheck *Basic error check*

Description

This function checks that the enviromnemt variable is internally consistent.

Usage

```
fm.errorcheck(env)
```

Arguments

`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The ouput is TRUE or FALSE.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.errorcheck(env)
```

fm.expand_2add_full *Function for exporting full representation of 2-additive capacity*

Description

From sparse to full representation of 2-additive capacity (singletons and pairs, augmented with 0s).

Usage

```
fm.expand_2add_full(n, envsp=NULL)
```

Arguments

`n`	Number of inputs
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is a sparse to full representation of 2-additive capacity (singletons and pairs, augmented with 0s)

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_pair_sparse(1,2,0.4,envsp);

cap2add <- fm.expand_2add_full(n,envsp)
cap2add

envsp <- fm.FreeSparseFM(envsp)
```

---

fm.expand_sparse_full *Function for exporting full capacity from sparse representation*

Description

Exports from sparse to full capacity.

Usage

```
fm.expand_sparse_full(n, envsp=NULL)
```

Arguments

n	Number of inputs.
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	Exports from sparse to full capacity.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n<-3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_pair_sparse(1,2,0.4,envsp);

cap <- fm.expand_sparse_full(n, envsp)
cap

envsp <- fm.FreeSparseFM(envsp)
```

---

fm.export_maximal_chains

Function for exporting maximal chains

Description

Returns in mc the arrays of maximal chains (there are n! such arrays) of a fuzzy measure v. Each maximal chain corresponds to the coefficients of a linea. function on the respective simplex

Usage

```
fm.export_maximal_chains(v, env = NULL)
```

Arguments

v	Fuzzy measure in general representation.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is mc the arrays of maximal chains

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
exportmaximalchains <- fm.export_maximal_chains(
 c(0, 0.00224, 0.0649, 0.510, 0.00965, 0.374,0.154, 1),env)
```

fm.fitting *Fuzzy Measure Fitting function*

Description

Estimate values of the fuzzy measures from empirical data.

Usage

```
fm.fitting(data, env=NULL, kadd="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in,y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
env	Environment variable obtained from fm.Init(n).
kadd	The value of k-additivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n.

Value

output	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
```

```
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
        nrow=20,
        ncol=4,byrow=TRUE);
fm.fitting(d,env)
```

`fm.fitting2additive` *Fuzzy Measure Fitting function*

Description

Estimate values of the fuzzy measures from empirical data tailored 2-additive standard fuzzy measure.

Usage

```
fm.fitting2additive(data, options=0, indexlow, indexhigh , option1=0, orness)
```

Arguments

`data`	is the empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector contains utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 store the observed aggregating value y.
`options`	options (default value is 0) 1 - lower bounds on Shapley values supplied in indexlow, 2 - upper bounds on Shapley values supplied in indexhigh, 3 - lower and upper bounds on Shapley values supplied in indexlow and indexhigh, 4 - lower bounds on all interaction indices supplied in indexlow, 5 - upper bounds on all interaction indices supplied in indexhigh, 6 - lower and upper bounds on all interaction indices supplied inindexlow and indexhigh. All these value will be treated as additional constraints in the LP.
`indexlow`	optional array of size n (options =1,2,3) or m (options=4,5,6) containing the lower bounds on the Shapley values or interaction indices
`indexhigh`	optional array of size n (options =1,2,3) or m (options=4,5,6) containing the upper bounds on the Shapley values or interaction indices
`option1`	if the value is 1, the interval of orness values will be fitted (and the desired low and high orness values should be provided). If 0, no additional orness constraints.
`orness`	optional array of size 2, for example c(0.1,1)

Value

output	The output is an array containing the values of a standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);



indexlow=c(0.1,0.1,0.2);
indexhigh=c(0.9,0.9,0.5);

fm.fitting2additive(d, options=3, indexlow, indexhigh, option1=0, orness=c(0.1,0.7))
```

---

fm.fittingKinteractive

Fuzzy Measure Fitting function

Description

Estimate values of the k-interacive fuzzy measures from empirical data.

Usage

```
fm.fittingKinteractive(data, env=NULL, kadd="NA", K="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
env	Environment variable obtained from fm.Init(n).
kadd	Value of k-interactivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = 2.
K	Value of FM value for sets of cardinality kadd+1, its default value is K = 0.5.

Value

output	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
```

```
fm.fittingKinteractive(d,env,2,0.8)
```

---

fm.fittingKinteractiveAuto *Fuzzy Measure Fitting function of the k-interactive*

Description

Estimate values of the k-interacive fuzzy measures from empirical data.

Usage

```
fm.fittingKinteractiveAuto(data, env=NULL, kadd="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
env	Environment variable obtained from fm.Init(n).
kadd	Value of k-interactivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = 2. The constant K the value of FM value for sets of cardinality kadd+1 is computed from data.

Value

output	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
```

```
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
fm.fittingKinteractiveAuto(d,env,2)
```

---

fm.fittingKinteractiveMarginal

Fuzzy Measure Fitting function of the k-interactive using marginal representation

---

Description

Estimate values of the k-interacive fuzzy measures from empirical data using marginal representation.

Usage

```
fm.fittingKinteractiveMarginal(data, env=NULL, kadd="NA", K="NA", submod ="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
env	Environment variable obtained from fm.Init(n).
kadd	Value of k-interactivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = 2.
K	The constant K, the value of FM value for sets of cardinality kadd+1 is computed from data, default 0.5.
submod	-1 indicates supermodular FM is needed, +1 indicates submodular, 0 otherwise. Should be consistent with K and n, see manual

Value

output — The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
fm.fittingKinteractiveMarginal(d,env,2,0.6, 0)
```

---

fm.fittingKinteractiveMarginalMC

Fuzzy Measure Fitting function of the k-interactive using marginal representation and maximal chains method

---

Description

Estimate values of the k-interacive fuzzy measures from empirical data using marginal representation and maximal chains method.

Usage

```
fm.fittingKinteractiveMarginalMC(data, env=NULL, kadd="NA", K="NA", submod ="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
env	Environment variable obtained from fm.Init(n).
kadd	Value of k-interactivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = 2.
K	The constant K the value of FM value for sets of cardinality kadd+1 is computed from data, default 0.5.
submod	-1 indicates supermodular FM is needed, +1 indicates submodular, 0 otherwise. Should be consistent with K and n, see manual

Value

output	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
```

```
fm.fittingKinteractiveMarginalMC(d,env,2,0.6,0)
```

---

fm.fittingKinteractiveMC   *Fuzzy Measure Fitting function of the k-interactive using maximal chains method*

Description

Estimate values of the k-interacive fuzzy measures from empirical data using maximal chains method.

Usage

```
fm.fittingKinteractiveMC(data, env=NULL, kadd="NA", K="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x(i2,...,x_in), y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
env	Environment variable obtained from fm.Init(n).
kadd	Value of k-interactivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = 2.
K	The constant K the value of FM value for sets of cardinality kadd+1 is computed from data, default 0.5.

Value

output	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
```

```
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
        nrow=20,
        ncol=4,byrow=TRUE);
fm.fittingKinteractiveMC(d,env,2,0.6)
```

`fm.fittingKmaxitive` *Fuzzy Measure Fitting function of the k-maxitive*

Description

Estimate values of the k-maxitive fuzzy measures from empirical data.

Usage

```
fm.fittingKmaxitive(data, env=NULL, kadd="NA")
```

Arguments

`data`	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
`env`	Environment variable obtained from fm.Init(n).
`kadd`	Value of k-maxitivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n.

Value

`output`	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
fm.fittingKmaxitive(d,env,2)
```

`fm.fittingKtolerant` *Fuzzy Measure Fitting function of the k-tolerant*

Description

Estimate values of the k-tolerant fuzzy measures from empirical data.

Usage

```
fm.fittingKtolerant(data, env=NULL, kadd="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.

env	Environment variable obtained from fm.Init(n).
kadd	Value of k-tolerance, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n.

Value

output	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
fm.fittingKtolerant(d,env,2)
```

---

fm.fittingMob — *Mobius Fuzzy Measure Fitting function*

Description

Estimate values of the Mobius fuzzy measures from empirical data.

Usage

```
fm.fittingMob(data, env=NULL ,kadd="NA")
```

Arguments

data	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 store the observed aggregating value y.
env	Environment variable obtained from fm.Init(n).
kadd	value of k-additivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its defaultvalue is kadd = n.

Value

output	The output is an array of size 2^n containing estimated Mobius fuzzy measure in binary ordering.

Note

The fit might not be perfect, and not all the constraints can be fully met.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
```

```
        nrow=20,
        ncol=4,byrow=TRUE);
env<-fm.Init(3)
fm.fittingMob(d,env)
```

`fm.fittingOWA` *Symmetric Fuzzy Measure Fitting function*

Description

Estimate values of the symmetric fuzzy measures from empirical data. The resulting Choquet integral is the OWA function.

Usage

```
fm.fittingOWA(data, env=NULL)
```

Arguments

`data`	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is an array of size n containing estimated OWA coefficients.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
```

```
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
fm.fittingOWA(d,env)
```

`fm.fittingWAM` *Additive Fuzzy Measure Fitting function*

Description

Estimate values of an additive fuzzy measure from empirical data. In this case the Choquet integral is the weighted arithmetic mean WAM.

Usage

```
fm.fittingWAM(data, env=NULL)
```

Arguments

`data`	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector containing utility values of n input criteria x_i1,x_i2,...,x_in,y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 stores the observed aggregated value y.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is an array of size n containing estimated weighting vector of WAM.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
fm.fittingWAM(d,env)
```

---

fm.fm_arraysize   *Function for returning the length of the array*

Description

Returns the length of the array of values of k-interactive fuzzy measures. Useful for reserving memory.

Usage

```
fm.fm_arraysize(env = NULL, kint = "NA")
```

Arguments

env	Environment variable obtained from fm.Init(n).
kint	Interactive fuzzy measure. 0 < kint <= n

Value

output	The outputs is the length of the array of values of k-interactive fuzzy measures

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
arraylength <- fm.fm_arraysize(env,1)
```

---

fm.Free *FreeSparseFM function*

Description

Frees the memory previously allocated in env.

Usage

```
fm.Free(env)
```

Arguments

env	Structure required for auxiliary data. It is obtained from fm.Init(n).

Value

output	Frees the memory previously allocated in envsp.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
 n<-3
env <- fm.Init(n)
env<-fm.Free(env)
env
```

fm.FreeSparseFM *FreeSparseFM function*

Description

Frees the memory previously allocated in envsp.

Usage

```
fm.FreeSparseFM(envsp)
```

Arguments

envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	Frees the memory previously allocated in envsp.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
 n<-3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.FreeSparseFM(envsp)
envsp <- fm.PrepareSparseFM(n, c(0.2,0.4,0.1), c(2,1,2,2,1,3,3,1,2,3))
 envsp <- fm.FreeSparseFM(envsp)
envsp
```

fm.FuzzyMeasureFitLP *Fuzzy Measure Fitting function.*

Description

Estimate values of the fuzzy measures from empirical data. The result is an array containing the values of a standard fuzzy measure in binary ordering. kadd defines the complexity of fuzzy measure. If kadd is not provided, its default value is equal to the number of inputs.

Usage

```
fm.FuzzyMeasureFitLP(data, env=NULL, kadd="NA",
       options=0, indexlow=(NULL), indexhigh=(NULL) , option1=0, orness=(NULL))
```

Arguments

`data`	Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector contains utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria,the column n + 1 store the observed aggregating value y.
`env`	Environment variable obtained from fm.Init(n).
`kadd`	Value of k-additivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n. kadd is k in k-additive f-measure, 1 < kadd < n+1; if kdd=n - f.m. is unrestricted
`options`	Options default value is 0. 1 - lower bounds on Shapley values supplied in indexlow, 2 - upper bounds on Shapley values supplied in indexhigh, 3 - lower and upper bounds on Shapley values supplied in indexlow and indexhigh, 4 - lower bounds on all interaction indices supplied in indexlow, 5 - upper bounds on all interaction indices supplied in indexhigh, 6 - lower and upper bounds on all interaction indices supplied inindexlow and indexhigh. All these value will be treated as additional constraints in the LP.
`indexlow`	Array of size n (options =1,2,3) or m (options=4,5,6) containing the lower bounds on the Shapley values or interaction indices
`indexhigh`	Array of size n (options =1,2,3) or m (options=4,5,6) containing the upper bounds on the Shapley values or interaction indices
`option1`	If the value is 1, the interval of orness values will be fitted (and the desired low and high orness values should be provided). If 0, no additional orness constraints.
`orness`	Array of size 2, for example c(0.1,1)

Value

`output`	The output is an array of size 2^n containing estimated standard fuzzy measure in binary ordering.

Note

The fit might not be perfect, and not all the constraints can be fully met.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
            0.808716, 0.584991, 0.479858, 0.544309,
            0.350281, 0.895935, 0.822815, 0.625868,
            0.746582, 0.174103, 0.858917, 0.480347,
            0.71048, 0.513519, 0.303986, 0.387631,
```

```
            0.0149841, 0.0914001, 0.364441, 0.134229,
            0.147308, 0.165894, 0.988495, 0.388044,
            0.445679, 0.11908, 0.00466919, 0.0897714,
            0.00891113, 0.377869, 0.531647, 0.258585,
            0.571167, 0.601746, 0.607147, 0.589803,
            0.166229, 0.663025, 0.450775, 0.357412,
            0.352112, 0.0570374, 0.607666, 0.270228,
            0.783295, 0.802582, 0.519867, 0.583348,
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
env<-fm.Init(3)
fm.FuzzyMeasureFitLP(d,env)
indexlow=c(0.1,0.1,0.2);
indexhigh=c(0.9,0.9,0.5);
fm.FuzzyMeasureFitLP(d,env, kadd=2, indexlow, indexhigh,
  options=3, option1=1, orness=c(0.1,0.7))
```

---

fm.FuzzyMeasureFitLPMob

Mobius Fuzzy Measure Fitting function, R wrapper for FuzzyMeasureFitLP() in fuzzymeasurefit.cpp

---

Description

Estimate values of the Mobius fuzzy measures from empirical data. The result is an array containing the values of the fuzzy measure in Mobius, ordered according to set cardinalities. kadd defines the complexity of fuzzy measure. if kadd is not provided, its default value is equal to the number of inputs.

Usage

```
fm.FuzzyMeasureFitLPMob(data, env=NULL, kadd="NA",
       options=0, indexlow=(NULL), indexhigh=(NULL) , option1=0, orness=(NULL))
```

Arguments

`data` Empirical data set in pairs (x_1,y_1),(x_2,y_2),...,(x_d,y_d) where x_i in [0,1]^n is a vector contains utility values of n input criteria x_i1,x_i2,...,x_in, y_i in [0,1] is a single aggregated value given by decision makers. The data is stored as a matrix of M by n+1 elements, where M is the number of data instances, and n is the number of input criteria, the column n + 1 store the observed aggregating value y.

env	Environment variable obtained from fm.Init(n).
kadd	Value of k-additivity, which is used for reducing the complexity of fuzzy measures. kadd is defined as an optional argument, its default value is kadd = n. kadd is k in k-additive f-measure, 1 < kadd < n+1; if kdd=n - f.m. is unrestricted
options	Options default value is 0. 1 - lower bounds on Shapley values supplied in indexlow, 2 - upper bounds on Shapley values supplied in indexhigh, 3 - lower and upper bounds on Shapley values supplied in indexlow and indexhigh, 4 - lower bounds on all interaction indices supplied in indexlow, 5 - upper bounds on all interaction indices supplied in indexhigh, 6 - lower and upper bounds on all interaction indices supplied inindexlow and indexhigh. All these value will be treated as additional constraints in the LP.
indexlow	Array of size n (options =1,2,3) or m (options=4,5,6) containing the lower bounds on the Shapley values or interaction indices
indexhigh	Array of size n (options =1,2,3) or m (options=4,5,6) containing the upper bounds on the Shapley values or interaction indices
option1	If the value is 1, the interval of orness values will be fitted (and the desired low and high orness values should be provided). If 0, no additional orness constraints.
orness	Array of size 2, for example c(0.1,1)

Value

output	The output is an array of size 2^n containing estimated Mobius fuzzy measure in binary ordering.

Note

The fit might not be perfect, and not all the constraints can be fully met.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
d <-  matrix( c( 0.00125122, 0.563568, 0.193298, 0.164338,
          0.808716, 0.584991, 0.479858, 0.544309,
          0.350281, 0.895935, 0.822815, 0.625868,
          0.746582, 0.174103, 0.858917, 0.480347,
          0.71048, 0.513519, 0.303986, 0.387631,
          0.0149841, 0.0914001, 0.364441, 0.134229,
          0.147308, 0.165894, 0.988495, 0.388044,
          0.445679, 0.11908, 0.00466919, 0.0897714,
          0.00891113, 0.377869, 0.531647, 0.258585,
          0.571167, 0.601746, 0.607147, 0.589803,
          0.166229, 0.663025, 0.450775, 0.357412,
          0.352112, 0.0570374, 0.607666, 0.270228,
          0.783295, 0.802582, 0.519867, 0.583348,
```

```
            0.301941, 0.875946, 0.726654, 0.562174,
            0.955872, 0.92569, 0.539337, 0.633631,
            0.142334, 0.462067, 0.235321, 0.228419,
            0.862213, 0.209595, 0.779633, 0.498077,
            0.843628, 0.996765, 0.999664, 0.930197,
            0.611481, 0.92426, 0.266205, 0.334666,
            0.297272, 0.840118, 0.0237427, 0.168081),
       nrow=20,
       ncol=4,byrow=TRUE);
env<-fm.Init(3)
fm.FuzzyMeasureFitLPMob(d,env)
indexlow=c(0.1,0.1,0.2);
indexhigh=c(0.9,0.9,0.5);
fm.FuzzyMeasureFitLPMob(d,env, kadd=2, indexlow, indexhigh,
  options=3, option1=1, orness=c(0.1,0.7))
```

`fm.generate_antibuoyant` *Function for generating one antibuoyant random fuzzy measure*

Description

Generates one antibuoyant random fuzzy measure in standard representation.

Usage

```
fm.generate_antibuoyant(env = NULL)
```

Arguments

`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is one antibuoyant random fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fuzzymeasures <- fm.generate_antibuoyant(env)
fuzzymeasures
```

`fm.generate_balanced` *Function for random generation of balanced fuzzy measures in standard representation*

Description

Generate several balanced random fuzzy measures in standard representation.

Usage

```
fm.generate_balanced(num, env=NULL)
```

Arguments

`num`	Generates num random fuzzy measures stored in an array v of length num * 2n.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output are several random fuzzy measures containing in an array v of length num * 2n

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fuzzymeasures  <- fm.generate_balanced(2, env)
fuzzymeasures
```

`fm.generate_belief` *Function for random generation of belief fuzzy measures in standard representation*

Description

Generate several random k-additive belief measures in Mobius representation.

Usage

```
fm.generate_belief(num, kadd, env=NULL)
```

Arguments

num	Generates num random belief measures stored in an array Mob of length num * fm_arraysize_kadd(n, kadd).
kadd	k-additivity
env	Environment variable obtained from fm.Init(n).

Value

output	The output are several random belief measures containing in an array v of length num * fm_a

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(5)
belief  <- fm.generate_belief(2, 3, env)
# 2 3-additive measures with n=5
belief
```

---

fm.generate_fmconvex_tsort

Function for generating convex fuzzy measures

Description

Generates num convex random fuzzy measures stored consecutively in cardinality ordering in the output array.

Usage

```
fm.generate_fmconvex_tsort(num, kint, markov, option, K, env = NULL)
```

Arguments

num	Several random fuzzy measures stored in cardinality ordering in the array v (num is their number)
kint	Interactive fuzzy measure. 0 < kint <= n
markov	Number of Markov steps to take, the randomness increases with that number
option	Option = 1 employs internal rejection method to improve uniformity, but for n > 5 is is not essential
K	K is the constant in k-interactive fuzzy measures
env	Environment variable obtained from fm.Init(n).

Value

`output` The output is the generation of num convex random fuzzy measures stored consecutively in cardinality ordering in the array v

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fuzzymeasures <- fm.generate_fmconvex_tsort(1,3,1000,0,1, env)
```

---

fm.generate_fm_2additive

Function for generating 2-additive fuzzy measures in Mobius representation

---

Description

Generates num random 2-additive fuzzy measures in Mobius representation.

Usage

```
fm.generate_fm_2additive(num, n)
```

Arguments

`num`	Generates num random fuzzy measures stored consecutively in cardinality ordering in the array v.
`n`	Number of inputs

Value

`output` The output are random fuzzy measures, it contains singletons and pairs but no emptyset

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
num <- 2
n <- 5
fuzzymeasures <- fm.generate_fm_2additive(num,n)
fuzzymeasures$V
fuzzymeasures$len
```

---

fm.generate_fm_2additive_concave

Function for generating 2additive concave fuzzy measures.

Description

Generates num 2-additive concave (supermodular) fuzzy measures for n inputs.

Usage

```
fm.generate_fm_2additive_concave(num, n)
```

Arguments

num	Generated num concave random fuzzy measures stored consecutively in cardinality ordering in the array v
n	Number of inputs

Value

output	The output is the length of the part of the array v allocated for each fuzzy measure, and the array with singletons and pairs in Mobius representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
fuzzymeasures <- fm.generate_fm_2additive_concave(10,20)
```

---

fm.generate_fm_2additive_convex
Function for generating 2-additive convex fuzzy measures

Description

Generates num 2-additive convex (supermodular) fuzzy measures for n inputs.

Usage

```
fm.generate_fm_2additive_convex(num, n)
```

Arguments

num	Generates num convex random fuzzy measures stored consecutively in cardinality ordering in the array v
n	Number of inputs

Value

output	The output is the length of the part of the array v allocated for each fuzzy measure, and the array with singletons and pairs in Mobius representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
addconvex <- fm.generate_fm_2additive_convex(5,20)
```

---

fm.generate_fm_2additive_convex_sparse
Function for generating 2additive convex numbers in sparse representation

---

Description

Generates a random 2-additive supermodular fuzzy measure in sparse representation.

Usage

```
fm.generate_fm_2additive_convex_sparse(n, envsp = NULL)
```

Arguments

n	Number of inputs
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output are 2-additive supermodular fuzzy measure in sparse representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 5
tups<-vector()
tupsidx<-vector()
envsp <- fm.PrepareSparseFM(n, tups,tupsidx)
envsp <- fm.generate_fm_2additive_convex_sparse(n, envsp)
envsp
   envsp <- fm.FreeSparseFM(envsp)
```

---

fm.generate_fm_2additive_convex_withsomeindependent

Function for generating 2additive convex fuzzy measures with some independent inputs

---

Description

Generates num 2-additive convex (supermodular) fuzzy measures for n inputs. Some of the interaction indices are set to 0 (independence).

Usage

```
fm.generate_fm_2additive_convex_withsomeindependent(num, n)
```

Arguments

num	Generates num convex random fuzzy measures stored consecutively in cardinality ordering in the array
n	Number of inputs

Value

output	The output is the length of the part of the array v allocated for each fuzzy measure, and the array with singletons and pairs in Mobius representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
addconvex <- fm.generate_fm_2additive_convex_withsomeindependent(5,20)
```

---

fm.generate_fm_2additive_randomwalk2

Function for generating random 2-additive fuzzy measures in Mobius representation by using random walk.

Description

Generate a random 2-additive fuzzy measures in Mobius representation by using random walk.

Usage

```
fm.generate_fm_2additive_randomwalk2(num, n, markov, option, step, Fn)
```

Arguments

`num`	Generated num random fuzzy measures stored consecutively in cardinality ordering in the array v.
`n`	Number of inputs.
`markov`	Number of Markov steps to take, the randomness increases with that number.
`option`	Not used, reserved for future use.
`step`	The maximum size of random steps (with respect to each value). The actual step is a random value up to Step.
`Fn`	The callback function to verify any additional conditions on generated FM. Provided by the user or NULL.

Value

`output`	The output are random 2-additive fuzzy measure, it contains singletons and pairs but no emptyset.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
num <- 10
n <- 5
fuzzymeasures <- fm.generate_fm_2additive_randomwalk2(num, n, 1000, 0, 0.001, NULL)
```

`fm.generate_fm_kadditive_convex_sparse`
Generate kadditive convex sparse fuzzy measures

Description

Generates a random k-additive Belief fuzzy measure in sparse representation

Usage

```
fm.generate_fm_kadditive_convex_sparse(n, kadd, nonzero, envsp = NULL)
```

Arguments

`n`	Inputs length. (n inputs)
`kadd`	kadd is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. default value is kadd = n. 1 < kadd < n+1; if kdd=n - f.m. is unrestricted
`nonzero`	Values stored and indexed in the respective arrays which are part of the structure
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output`	The output is k-additive Belief fuzzy measure in sparse representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
  n <- 5
tups<-vector()
tupsidx<-vector()
envsp <- fm.PrepareSparseFM(n, tups,tupsidx)
envsp <- fm.generate_fm_kadditive_convex_sparse(n,4,10, envsp)
envsp
   envsp <- fm.FreeSparseFM(envsp)
```

fm.generate_fm_kinteractivedualconcave

Function for generating k-interactive dual concave fuzzy measures in Mobius representation

Description

Generates num k-interactive dual concave fuzzy measures in Mobius representation using random walk of length markov of stepsize step

Usage

```
fm.generate_fm_kinteractivedualconcave(num, n, kadd, markov,  step,  Fn, env)
```

Arguments

`num`	Generated num random fuzzy measures stored consecutively in cardinality ordering in the array v.
`n`	Number of inputs.
`kadd`	kadd is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. default value is kadd = n. 1 < kadd < n+1; if kdd=n - f.m. is unrestricted.
`markov`	Number of Markov steps to take, the randomness increases with that number.
`step`	The maximum size of random steps (with respect to each value). The actual step is a random value up to Step.
`Fn`	The callback function to verify any additional conditions on generated FM. Provided by the user.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output are k-interactive dual concave fuzzy measures in Mobius representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(4)
step <- 0.001
Fn <- NULL

fuzzymeasures <- fm.generate_fm_kinteractivedualconcave(10, 4, 2, 1000, step, Fn, env)
fuzzymeasures
```

`fm.generate_fm_kinteractivedualconvex`
Function for generating several k-interactive dual convex fuzzy measures in Mobius representation

Description

Generates num k-interactive dual convex fuzzy measures in Mobius representation using random walk of length markov of stepsize step.

Usage

```
fm.generate_fm_kinteractivedualconvex(num, n, kadd, markov,  step, Fn, env)
```

Arguments

`num`	Generated num random fuzzy measures stored consecutively in cardinality ordering in the array v.
`n`	Number of inputs.
`kadd`	kadd is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. default value is kadd = n. 1 < kadd < n+1; if kdd=n - f.m. is unrestricted.
`markov`	Number of Markov steps to take, the randomness increases with that number.
`step`	The maximum size of random steps (with respect to each value). The actual step is a random value up to Step.
`Fn`	The callback function to verify any additional conditions on generated FM. Provided by the user.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output are several k-interactive dual convex fuzzy measures in Mobius representation

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(4)
step <- 0.0001
Fn <- NULL

fuzzymeasures <- fm.generate_fm_kinteractivedualconvex(10, 4, 2, 1000, step, Fn, env)
```

```
fuzzymeasures

env<-fm.Free(env)
```

---

fm.generate_fm_minplus
Generate randomly fuzzy measures

Description

Generate several random fuzzy measures (num is their number) stored in cardinality ordering in the array v using minimals_plus method.

Usage

```
fm.generate_fm_minplus(num, kint, markov, option, K, env = NULL)
```

Arguments

num	Generated num random fuzzy measures stored consecutively in cardinality ordering in the array
kint	Interactive fuzzy measure. 0 < kint <= n
markov	Number of Markov steps to take, the randomness increases with that number
option	Option = 1 employs internal rejection method to improve uniformity, but for n > 5 is is not essential
K	K is the constant in k-interactive fuzzy measures
env	Environment variable obtained from fm.Init(n).

Value

output	The output is generate several random fuzzy measures

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fuzzymeasures <- fm.generate_fm_minplus(10,3,1000,0,0.7, env)
fuzzymeasures
```

`fm.generate_fm_randomwalk`
Function for generating several k-additive fuzzy measure

Description

Generates num k-additive fuzzy measures in the standard or Mobius representation using random walk of length markov of stepsize step.

Usage

```
fm.generate_fm_randomwalk(num, n, kadd, markov, option, step, Fn, env)
```

Arguments

`num`	Generated num random fuzzy measures stored consecutively in standard or cardinality ordering in the array v.
`n`	Number of inputs
`kadd`	kadd is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. default value is kadd = n. 1 < kadd < n+1; if kdd=n - f.m. is unrestricted. The parameter kadd only matters for options 3 and 5
`markov`	Number of Markov steps to take, the randomness increases with that number.
`option`	Option = 0 - normal, 1 convex (supermodular), 2 antibuoyant, 3 kadditive , 4 belief measure, 5 kadditive convex. The measure generated is in standard representation fo all options except 3,5. The parameter kadd only matters for options 3 and 5. In that case the measure is in more compact Mobius representation.
`step`	The maximum size of random steps (with respect to each value). The actual step is a random value up to Step.
`Fn`	The callback function to verify any additional conditions on generated FM. Provided by the user. if not NULL, is a callback function to perform additional check at every Markov step of the current set function, i.e., any extra conditions
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is named list with the first element v being the fuzzy measure and the second being the length of the array containing it

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
 Fn <- function(n,v){
      out <- 0.0

      for(i in 1:n) out<- out+v[i];

      if(out>1) {
      return(0)
      } else
      return(1)
  }
  env<-fm.Init(3)
  step <- 0.0010
  Option <- 3
  n <- 3

fuzzymeasures <- fm.generate_fm_randomwalk(2, 3, 2, 1000, Option, step, Fn, env)
print(fuzzymeasures)
print(fuzzymeasures$length)
```

---

fm.generate_fm_sorting

Function for random generation of fuzzy measures in standard representation

---

Description

Generate several random fuzzy measures in standard representation

Usage

```
fm.generate_fm_sorting(num, markov, option, env = NULL)
```

Arguments

num	Generates num random fuzzy measures stored in an array v of length num * 2n.
markov	Number of Markov steps to take, the randomness increases with that number.
option	Option = 1 employs internal rejection method to improve uniformity, but for n > 5 is not essential.
env	Environment variable obtained from fm.Init(n).

Value

output	The output are several random fuzzy measures containing in an array v of length num * 2n

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
markovsteps <- 100
fuzzymeasures <- fm.generate_fm_sorting(5, markovsteps, 0, env)
fuzzymeasures
```

---

fm.generate_fm_tsort *Function for random generation of fuzzy measures*

Description

Generate several random fuzzy measures (num is their number) stored in cardinality ordering in the array v using topological sort.

Usage

```
fm.generate_fm_tsort(num, kint, markov, option, K, env = NULL)
```

Arguments

`num`	Generated num random fuzzy measures stored consecutively in cardinality ordering in the array
`kint`	Interactive fuzzy measure. 0 < kint <= n
`markov`	Number of Markov steps to take, the randomness increases with that number
`option`	Option = 1 employs internal rejection method to improve uniformity, but for n > 5 is is not essential
`K`	K is the constant in k-interactive fuzzy measures
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is generate several random fuzzy measures

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fuzzymeasures <- fm.generate_fm_tsort(10,3,1000,0,0.7, env)
fuzzymeasures
```

fm.get_num_tuples *Function for exporting number of tuples*

Description

Returns the number of tuples.

Usage

```
fm.get_num_tuples(envsp=NULL)
```

Arguments

envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is the number of tuples.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.4,envsp);
fm.get_num_tuples(envsp)

envsp <-fm.FreeSparseFM(envsp)
```

fm.get_sizearray_tuples *Function for exporting the size of the array of tuples*

Description

Returns the length of the array of tuples.

Usage

```
fm.get_sizearray_tuples(envsp=NULL)
```

Arguments

envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is the length of the array of tuples.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.4,envsp);
fm.get_sizearray_tuples(envsp)

envsp <- fm.FreeSparseFM(envsp)
```

---

fm.Init *Initialisation function*

Description

This function initialises the internal structures which makes computations faster. The structures are saved in the output environment variable, which should be subsequently passed to other functions. Several environment variables (for different dimensions) can be initialised at the same time.

Usage

```
fm.Init(n1)
```

Arguments

n1	The number of variables.

Value

output	The ouput is the enviromnet variable containing the internal structures.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
```

`fm.Interaction` *Interaction Index computation function*

Description

Calculates all the interaction indices of input criteria for standard fuzzy measure.

Usage

```
fm.Interaction(v,env)
```

Arguments

`v`	Fuzzy measure value in standard representation
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is a matrix, whose first column stores the interaction index values, and the second column stores the indices of criteria in coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Interaction(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.InteractionB *Banzhaf Interaction Index computation function*

Description

Calculates all the Banzhaf Interaction indices of input criteria for a standard fuzzy measure.

Usage

```
fm.InteractionB(v,env)
```

Arguments

v	Fuzzy measure value in standard representation
env	Environment variable obtained from fm.Init(n).

Value

output	The output is a matrix, whose first column stores the Banzhaf Interaction index values, and the second column stores the indices of criteria in coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.InteractionB(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.InteractionBMob *Banzhaf InteractionB Index computation function in Mobius representation*

Description

Calculates all the Banzhaf InteractionB indices of input criteria for a Mobius fuzzy measure.

Usage

```
fm.InteractionBMob(Mob,env)
```

Arguments

Mob	Fuzzy measure value in Mobius representation
env	Environment variable obtained from fm.Init(n).

Value

`output` The output is a matrix, whose first column stores the Banzhaf Interaction index values, and the second column stores the indices of criteria in coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.InteractionBMob(c( 0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

---

fm.InteractionMob *Interaction Index computation function for Mobius fuzzy measure*

Description

Calculates all the interaction indices of input criteria for a Mobius fuzzy measure.

Usage

```
fm.InteractionMob(Mob,env )
```

Arguments

`Mob`	Fuzzy measure value in Mobius representation
`env`	Environment variable obtained from fm.Init(n).

Value

`output` The output is a matrix, whose first column stores the interaction index values, and the second column stores the indices of criteria in coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.InteractionMob(c( 0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

`fm.IsMeasureAdditive` *IsMeasureAdditive function*

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureAdditive(v,env)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureAdditive(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

`fm.IsMeasureAdditiveMob` *IsMeasureAdditive function in Mobius representation*

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureAdditiveMob(Mob,env)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureAdditiveMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

`fm.IsMeasureBalanced` *IsMeasureBalanced function*

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureBalanced(v,env)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureBalanced(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

`fm.IsMeasureBalancedMob` *IsMeasureBalanced function in Mobius representation*

Description

Returns 1 if yes, 0 if no; Mob is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureBalancedMob(Mob,env)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from emperical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureBalancedMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

`fm.IsMeasureKmaxitive` *IsMeasureKmaxitive function*

Description

Returns k; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureKmaxitive(v,env=NULL)
```

Arguments

v	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is k. If k=n then not k-maxitive

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureKmaxitive(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

---

fm.IsMeasureKmaxitiveMob

IsMeasureKmaxitive function in Mobius representation

Description

Returns k; mob is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureKmaxitiveMob(Mob,env=NULL)
```

Arguments

Mob	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from emperical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is k. If k=n then not k-maxitive

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureKmaxitiveMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

---

fm.IsMeasureSelfdual *IsMeasureSelfdual function*

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureSelfdual(v,env)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSelfdual(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.IsMeasureSelfdualMob

IsMeasureSelfdual function in Mobius representation

Description

Returns 1 if yes, 0 if no; Mob is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureSelfdualMob(Mob,env)
```

Arguments

Mob	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSelfdualMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.IsMeasureSubadditive

IsMeasureSub additive function

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureSubadditive(v,env)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSubadditive(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

`fm.IsMeasureSubadditiveMob`

IsMeasureSub additive function in Mobius representation

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureSubadditiveMob(Mob,env)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSubadditiveMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

---

fm.IsMeasureSubmodular
IsMeasureSub modular function

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureSubmodular(v,env=NULL)
```

Arguments

v	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSubmodular(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.IsMeasureSubmodularMob

IsMeasureSubmodular function in Mobius representation

Description

Returns 1 if yes, 0 if no; Mob is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureSubmodularMob(Mob,env=NULL)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSubmodularMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.IsMeasureSuperadditive

IsMeasureSuperadditive function

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureSuperadditive(v,env=NULL)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSuperadditive(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

`fm.IsMeasureSuperadditiveMob` *IsMeasureSuperadditive function in Mobius representation*

Description

Returns 1 if yes, 0 if no; Mob is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureSuperadditiveMob(Mob,env=NULL)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSuperadditiveMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

`fm.IsMeasureSupermodular` *IsMeasureSupermodular function*

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureSupermodular(v,env=NULL)
```

Arguments

`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSupermodular(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.IsMeasureSupermodularMob *IsMeasureSupermodular function in Mobius representation*

Description

Returns 1 if yes, 0 if no; Mob is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureSupermodularMob(Mob,env=NULL)
```

Arguments

Mob	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSupermodularMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.IsMeasureSymmetric *IsMeasureSymmetric function*

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in standard representation.

Usage

```
fm.IsMeasureSymmetric(v,env=NULL)
```

Arguments

v	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSymmetric(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

---

fm.IsMeasureSymmetricMob

IsMeasureSymmetric function in Mobius representation

Description

Returns 1 if yes, 0 if no; v is a fuzzy measure in Mobius representation.

Usage

```
fm.IsMeasureSymmetricMob(Mob,env=NULL)
```

Arguments

`Mob`	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is 1 if yes, 0 if no.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.IsMeasureSymmetricMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.is_inset_sparse *Function for checking if i belongs to the tuple A*

Description

Checks if element i (1-based) belongs to the tuple indexed A (whose cardinality can be 1,2, other (automatically determined)).

Usage

```
fm.is_inset_sparse(A, card, i, envsp=NULL)
```

Arguments

A	Tuple indexed.
card	Whose cardinality can be 1,2, other (automatically determined)
i	Element (1-based)
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is a logical value.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.4,envsp);

fm.is_inset_sparse(0,3,1,envsp)
fm.is_inset_sparse(0,3,4,envsp)

envsp <- fm.FreeSparseFM(envsp)
```

fm.is_subset_sparse *Function for checking if tuple B is subset of tuple A*

Description

Checks if tuple B is a subset of tuple A, The cardinalities of both tuples need to be supplied.

Usage

```
fm.is_subset_sparse(A, cardA, B, cardB, envsp = NULL)
```

Arguments

A	Tuple
cardA	Whose cardinality can be 1,2, other (automatically determined)
B	Tuple, tup=0
cardB	Whose cardinality can be 1,2, other (automatically determined)
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is a logical value.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.4,envsp);
envsp <- fm.add_pair_sparse(1,2,0.2,envsp);
envsp <- fm.add_pair_sparse(1,3,0.3,envsp);

fm.is_subset_sparse(0,3,0,2,envsp) #is 0th pair a subset of the 0th tuple?
fm.is_subset_sparse(0,3,1,2,envsp) #is 1th pair a subset of the 0th tuple?

envsp<-fm.FreeSparseFM(envsp)
```

fm.max_subset_sparse *Maximun of x computation function in sparse representation*

Description

Calculates maximum of x with the indices belonging to tuple indexed as S

Usage

```
fm.max_subset_sparse(x, S, cardS, envsp=NULL)
```

Arguments

x	Input vector of size n, containing utility value of input criteria. x is in [0,1].
S	Indices belonging to tuple indexed
cardS	Cardinality cardS
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is the maximum of x with the indices belonging to tuple indexed as S

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.4,envsp);

fm.max_subset_sparse(c(0.1,0.05,0.2),0,3,envsp)
envsp <- fm.FreeSparseFM(envsp)
```

fm.min_subset_sparse *Minimun of x computation function in sparse representation*

Description

Calculates minimum of x with the indices belonging to tuple indexed as S

Usage

```
fm.min_subset_sparse(x, S, cardS, envsp=NULL)
```

Arguments

`x`	Input vector of size n, containing utility value of input criteria. x is in [0,1].
`S`	Indices belonging to tuple indexed
`cardS`	Cardinality cardS
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output`	The output is the minimum of x with the indices belonging to tuple indexed as S

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, vector(), vector())
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.4,envsp);

fm.min_subset_sparse(c(0.1,0.05,0.2),0,3,envsp)
 envsp <- fm.FreeSparseFM(envsp)
```

fm.Mobius *Mobius transform function*

Description

Calculates Mobius representation of general fuzzy measure, the input and output is an array of size 2^n=m in binary ordering.

Usage

```
fm.Mobius(v,env=NULL)
```

Arguments

v	Fuzzy measure value in standard representation.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is the fuzzy measure in Mobius representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Mobius(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.NonadditivityIndex *Nonadditivity index computation function*

Description

Calculate the nonadditivity indices of input criteria from general fuzzy measure.

Usage

```
fm.NonadditivityIndex(v,env=NULL)
```

Arguments

v	Fuzzy measure in general representation.
env	Environment variable obtained from fm.Init(n).

Value

output
: The output is an array of size 2^n, which contain nonadditivity indices of input criteria coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.NonadditivityIndex(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

---

fm.NonadditivityIndexMob
Nonadditivity index computation function in Mobius representation

---

Description

Calculate the nonadditivity indices of input criteria from general fuzzy measure in Mobius representation.

Usage

```
fm.NonadditivityIndexMob(Mob,env=NULL)
```

Arguments

Mob
: Fuzzy measure in Mobius representation.

env
: Environment variable obtained from fm.Init(n).

Value

output
: The output is an array of size 2^n, which contain nonadditivity indices of input criteria coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
NonadditivityIndex <- fm.NonadditivityIndexMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.NonmodularityIndex *Nonmodularity index computation function*

Description

Calculate all the m = 2^n nonmodularity indices of fuzzy measure v given in standard representation

Usage

```
fm.NonmodularityIndex(v, env = NULL)
```

Arguments

v	Fuzzy measure in general representation.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is an array of size m

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
Nonmodularityindex <- fm.NonmodularityIndex(c(0,0.3,0.5,0.6,0.4,0.8,0.7,1),env)
```

fm.NonmodularityIndexKinteractive *NonmodularityIndexKinteractive computation function*

Description

Calculate all the m = 2^n nonmodularity indices of k-interactive fuzzy measure v given in standard representation (in cardinality ordering)

Usage

```
fm.NonmodularityIndexKinteractive(v, env = NULL, kadd = "NA")
```

Arguments

v	Fuzzy measure in general representation.
env	Environment variable obtained from fm.Init(n).
kadd	kadd is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. default value is kadd = n. 1 < kadd < n+1; if kdd=n - f.m. is unrestricted

Value

output	The output is an array of size m.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.NonmodularityIndexKinteractive(c(0,0.3,0.5,0.6,0.4,0.8,0.7,1),env,2)
```

---

fm.NonmodularityIndexMob

Nonmodularityindex computation function in Mobius representation

---

Description

Calculates all the nonmodularity indices of fuzzy measure in Mobius representation representation

Usage

```
fm.NonmodularityIndexMob(Mob, env = NULL)
```

Arguments

Mob	Fuzzy measure in Mobius representation of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is an array of size m

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.NonmodularityIndexMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

---

fm.NonmodularityIndexMobkadditive
Function for calculating all Nonmodularity indices of k-additive in Mobius representation

---

Description

Calculate all the m = 2^n nonmodularity indices of k-additive in Mobius representation(in cardinality ordering)

Usage

```
fm.NonmodularityIndexMobkadditive(Mob, env = NULL, kadd = "NA")
```

Arguments

Mob	Fuzzy measure in Mobius representation of size m=2^n. Its values can be provided by users, or by estimating from empirical data
env	Environment variable obtained from fm.Init(n).
kadd	kadd is the value of k-additivity, which is used for reducing the complexity of fuzzy measures. default value is kadd = n. 1 < kadd < n+1; if kdd=n - f.m. is unrestricted.

Value

output	The output is an array of size m.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.NonmodularityIndexMobkadditive(c(0.0,0.3,0.5,-0.2,0.4,0.1,-0.2,0.1),env,2)
```

fm.NonmodularityIndex_sparse *Nonmodularity index computation function in sparse representation*

Description

Calculate all 2^n nonmodularity indices using Mobius transform of a fuzzy measure of lenght 2^n=m, using sparse representation

Usage

```
fm.NonmodularityIndex_sparse( n, envsp=NULL)
```

Arguments

n	Number of inputs
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is all 2^n nonmodularity indice.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
tups<-vector()
tupsidx<-vector()
envsp <- fm.PrepareSparseFM(n, tups,tupsidx)
envsp <- fm.add_singletons_sparse(c(0.2,0.1,0.2),envsp)
envsp <- fm.add_pair_sparse(1,2,0.4,envsp);

fm.NonmodularityIndex_sparse(3,envsp)
envsp <- fm.FreeSparseFM(envsp)
```

fm.OrnessChoquet *OrnessChoquet function*

Description

Calculate Orness value of the Choquet integral of the fuzzy measure, where v is a standard representation.

Usage

```
fm.OrnessChoquet(v,env=NULL)
```

Arguments

`v`	Standard fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is the Orness the Choquet integral for the fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.OrnessChoquet(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.OrnessChoquetMob *OrnessChoquet function in Mobius representation*

Description

Calculate Orness value of the Choquet integral of the fuzzy measure, where Mob is the Mobius representation.

Usage

```
fm.OrnessChoquetMob(Mob,env=NULL)
```

Arguments

Mob	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is the Orness the Choquet integral for the fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.OrnessChoquetMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

---

fm.populate_fm_2add_sparse

Function for populating 2-additive sparse capacity

Description

Populate 2-additive sparse capacity with nonzero values using the singletons and two arrays of indices (of size numpairs).

Usage

```
fm.populate_fm_2add_sparse(singletons, numpairs, pairs, indicesp1, indicesp2, envsp)
```

Arguments

singletons	Singletons 0-based.
numpairs	Size numpairs.
pairs	Array 0-based.
indicesp1	Array of indices of Size numpairs.need to be 1-based.
indicesp2	Array of indices of Size numpairs.need to be 1-based.
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is Populate 2-additive sparse capacity with nonzero values using the singletons and two arrays of indices (of size numpairs)

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
 n <- 3
 envsp <- fm.PrepareSparseFM(n, vector(), vector())

envsp <- fm.populate_fm_2add_sparse(c(0.1,0.2,0.3), 3,
           c(0.4,0.5,0.6), c(1,1,2), c(2,3,3), envsp)
 envsp
 envsp <- fm.FreeSparseFM(envsp)
```

---

fm.populate_fm_2add_sparse_from2add

Function for populating 2-additive sparse capacity from 2-additive capacity

---

Description

Given 2-additive capacity singletons+pairs in one array v , selects nonzero pairs and populates sparse capacity envsp

Usage

```
fm.populate_fm_2add_sparse_from2add(n, v, envsp=NULL)
```

Arguments

n	Number of inputs
v	Pairs in one array v
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is a nonzero pairs and populates sparse capacity envsp

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
        n <- 3
        envsp <- fm.PrepareSparseFM(n, vector(), vector())
        envsp <- fm.populate_fm_2add_sparse_from2add(3,c(0.4,0.5,0.6, 0, 0, 0.1),envsp)
envsp
     envsp <- fm.FreeSparseFM(envsp)
```

`fm.PrepareSparseFM` *PrepareSparseFM preparation function*

Description

This function initialises Sparse representation structure. It is used to allocate storage and later populate these values

Usage

```
fm.PrepareSparseFM(n, tups, tupsidx)
```

Arguments

`n`	Number of inputs
`tups`	Tuples to be added (can be null vector)
`tupsidx`	Cardinalities and indices (1-based) of the elements of tuples (can be null vector)

Value

`output`	The output allocate storage and later populate these values. envsp

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
   n<-3
    envsp <- fm.PrepareSparseFM(n, vector(), vector())
    envsp <- fm.FreeSparseFM(envsp)
    envsp <- fm.PrepareSparseFM(n, c(0.2,0.4,0.1), c(2,1,2,2,1,3,3,1,2,3))
envsp
     envsp <- fm.FreeSparseFM(envsp)
```

fm.Shapley *Shapley value computation function*

Description

Calculates the Shapley values of input criteria from general fuzzy measure,

Usage

```
fm.Shapley(v,env=NULL)
```

Arguments

`v`	Fuzzy measure in general representation.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is an array of size n, which contain Shapley values of input criteria.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Shapley(c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.Shapley2addMob *Function for calculating Shapley values of 2-additive fuzzy measure in Mobius representation*

Description

Calculate the Shapley values of a 2-additive fuzzy measure for n inputs given in Mobius representation. The results are in arrays.

Usage

```
fm.Shapley2addMob(n, Mob)
```

Arguments

`n`	Number of inputs
`Mob`	Fuzzy measure value in Mobius representation

Value

`output` The output is an array of size n, which contain Shapley indices of input criteria.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
Shapley <- fm.Shapley2addMob(3, c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1))
```

---

fm.ShapleyMob *Shapley value computation function in Mobius representation*

Description

Calculate the Shapley indices of input criteria from general fuzzy measure in Mobius representation.

Usage

```
fm.ShapleyMob(Mob,env=NULL)
```

Arguments

`Mob` Fuzzy measure in Mobius representation.

`env` Environment variable obtained from fm.Init(n).

Value

`output` The output is an array of size n, which contain Shapley values of input criteria.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.ShapleyMob(c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.ShapleyMob_sparse *Shapley values computation function in sparse representation*

Description

Calculate Shapley values vectors of size n of a sparse fuzzy measure

Usage

```
fm.ShapleyMob_sparse(n, envsp=NULL)
```

Arguments

`n`	Size of values vectors
`envsp`	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output`	The output is Shapley values vectors of size n of a sparse fuzzy measure.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
 n <- 3
 envsp <- fm.PrepareSparseFM(n, c(0.2,0.4,0.1), c(2,1,2,2,1,3,3,1,2,3))

fm.ShapleyMob_sparse(3, envsp)

envsp <- fm.FreeSparseFM(envsp)
```

fm.ShowCoalitions *Show Coalitions function*

Description

Return the decimal expression for the subsets A. In binary and in cardinality ordering respectively.

Usage

```
fm.ShowCoalitions(env = NULL)
```

Arguments

env	Environment variable obtained from fm.Init(n).

Value

output	is the array of integers which show the decimal expressions for all 2^n coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
ShowCoalitions <- fm.ShowCoalitions(env)
ShowCoalitions
```

---

fm.ShowCoalitionsCard *Show CoalitionsCard function*

Description

Return the decimal expression for the subsets A. In binary and in cardinality ordering respectively.

Usage

```
fm.ShowCoalitionsCard(env = NULL)
```

Arguments

env	Environment variable obtained from fm.Init(n).

Value

output	The output the decimal expression for the subsets A. It is the array of integers containing the decimal expressions for all 2^n coalitions.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
CoalitionsCard <- fm.ShowCoalitionsCard(env)
CoalitionsCard
```

`fm.sparse_get_pairs` *Get pairs computation function in sparse representation*

Description

Export the internal arrays of the sparse capacity as arrays of singletons, pairs and tuples.

Usage

```
fm.sparse_get_pairs( envsp=NULL)
```

Arguments

envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is the array of pairs and their number.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n)
envsp <-fm.add_pair_sparse(1,2, 0.4, envsp)
envsp <-fm.add_pair_sparse(1,3, 0.3, envsp)

pairs <- fm.sparse_get_pairs(envsp)
pairs
     envsp <- fm.FreeSparseFM(envsp)
```

`fm.sparse_get_singletons` *Get singletons of sparse fuzzy measure*

Description

Export the internal arrays of the sparse capacity as arrays of singletons, pairs and tuples.

Usage

```
fm.sparse_get_singletons(envsp=NULL)
```

Arguments

`envsp` Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output` The output is the numbers of pairs and tuples.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n)
envsp <- fm.add_singletons_sparse(c(0, 0.3, 0.5),envsp)
singletons <- fm.sparse_get_singletons(envsp)
singletons
     envsp <- fm.FreeSparseFM(envsp)
```

`fm.sparse_get_tuples` *Get tuples of a sparse fuzzy measure*

Description

Export the internal arrays of the sparse capacity as arrays of singletons, pairs and tuples.

Usage

```
fm.sparse_get_tuples(envsp=NULL)
```

Arguments

`envsp` Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

`output` The output is the numbers of pairs and tuples.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n)
envsp <- fm.add_tuple_sparse(c(1,2,3),0.2,envsp)
envsp <- fm.add_tuple_sparse(c(1,3,4),0.3,envsp)

tuples <- fm.sparse_get_tuples(envsp)
tuples
 envsp <- fm.FreeSparseFM(envsp)
```

`fm.Sugeno` *Sugeno computation function*

Description

Calculate the value of a Sugeno integral of input x, with fuzzy measure in standard representation

Usage

```
fm.Sugeno(x, v,env=NULL)
```

Arguments

`x`	Input vector of size n, containing utility value of input criteria. x is in [0,1].
`v`	General fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
`env`	Environment variable obtained from fm.Init(n).

Value

`output`	The output is a single value of the computed Sugeno integral.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Sugeno(c(0.6, 0.3, 0.8), c(0, 0.3, 0.5, 0.6, 0.4, 0.8, 0.7, 1),env)
```

fm.SugenoMob — *Sugeno function in Mobius representation*

Description

Calculate the value of a Sugeno integral of input x, with fuzzy measure in mobius representation

Usage

```
fm.SugenoMob(x, Mob,env=NULL)
```

Arguments

x	Input vector of size n, containing utility value of input criteria. x is in [0,1].
Mob	Mobius fuzzy measure of size m=2^n. Its values can be provided by users, or by estimating from empirical data.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is a single value of the computed Sugeno integral.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.SugenoMob(c(0.6, 0.3, 0.8), c(0.0, 0.3, 0.5, -0.2, 0.4, 0.1, -0.2, 0.1),env)
```

fm.test — *Test function*

Description

This function provide some examples of how fuzzy measure operation in this toolbox are used. It can be used to test if the toolbox has been installed successfully or not.

Usage

```
fm.test()
```

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
fm.test()
```

---

fm.tuple_cardinality_sparse *Tuple cardinality*

Description

Returns the cardinality of the tuple numbered i in the list of tuples.

Usage

```
fm.tuple_cardinality_sparse(i, envsp = NULL)
```

Arguments

i	In the list of tuples.
envsp	Structure required for sparse representation which stores the relevant values (k-tuples). It is obtained from fm.PrepareSparseFM(n).

Value

output	The output is the cardinality of the tuple numbered i in the list of tuple.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
n <- 3
envsp <- fm.PrepareSparseFM(n, c(0.2,0.4,0.1), c(2,1,2,2,1,3,3,1,2,3))
fm.tuple_cardinality_sparse(0,envsp)
 envsp <- fm.FreeSparseFM(envsp)
```

fm.Zeta *Zeta transform function*

Description

Calculate the general fuzzy measure from Mobius representation. The input and output is an array of size 2^n=m in binary ordering. This is the inverse of the Mobius function.

Usage

```
fm.Zeta(Mob,env)
```

Arguments

Mob	Fuzzy measure value in Mobius representation.
env	Environment variable obtained from fm.Init(n).

Value

output	The output is the fuzzy measure in general representation.

Author(s)

Gleb Beliakov, Andrei Kelarev, Quan Vu, Daniela L. Calderon, Deakin University

Examples

```
env<-fm.Init(3)
fm.Zeta(c(0.0,0.3,0.5,-0.2,0.4,0.1,-0.2,0.1),env)
```

Index