Package 'SparseGrid'

July 21, 2025

Type Package

Title Sparse grid integration in R

Version 0.8.2

Date 2012-07-31

Author Jelmer Ypma

Maintainer Jelmer Ypma <uctpjyy@ucl.ac.uk>

Suggests testthat, statmod, mvtnorm

Description SparseGrid is a package to create sparse grids for numerical integration, based on code from www.sparse-grids.de

License GPL

NeedsCompilation no

Repository CRAN

Date/Publication 2013-07-31 21:18:24

Contents

`createIntegrationGrid` *Create integration grid with the least number of nodes, either using a sparse grid or a product rule grid.*

Description

This function creates nodes and weights that can be used for integration. It is a convenience function that calls createSparseGrid and createProductRuleGrid and returns the grid with the least number of nodes. Typically, a grid created by the product rule will only contain fewer nodes than a sparse grid for very low dimensions.

Usage

```
createIntegrationGrid(type, dimension, k, sym = FALSE)
```

Arguments

`type`	String or function for type of 1D integration rule, can take on values `"KPU"` Nested rule for unweighted integral over [0,1] `"KPN"` Nested rule for integral with Gaussian weight `"GQU"` Gaussian quadrature for unweighted integral over [0,1] (Gauss-Legendre) `"GQN"` Gaussian quadrature for integral with Gaussian weight (Gauss-Hermite) `func` any function. Function must accept level k and return a list with two elements `nodes` and `weights` for univariate quadrature rule with polynomial exactness 2k-1.
`dimension`	Dimension of the integration problem.
`k`	Accuracy level. The rule will be exact for polynomials up to total order 2k-1.
`sym`	(optional) only used for own 1D quadrature rule (type not "KPU",...). If sym is supplied and not FALSE, the code will run faster but will produce incorrect results if 1D quadrature rule is asymmetric.

Value

The return value contains a list with nodes and weights

`nodes`	matrix with a node in each row
`weights`	vector with corresponding weights

Author(s)

Jelmer Ypma

References

Florian Heiss, Viktor Winschel, Likelihood approximation by numerical integration on sparse grids, Journal of Econometrics, Volume 144, Issue 1, May 2008, Pages 62-80, http://www.sparse-grids.de

See Also

`createSparseGrid` `createProductRuleGrid` `createMonteCarloGrid` `integrate` `pmvnorm`

Examples

```
# load library
library('SparseGrid')

# create integration grid
int.grid <- createIntegrationGrid( 'GQU', dimension=3, k=5 )
```

`createMonteCarloGrid` *Create a multidimensional grid of nodes and weights for Monte Carlo integration*

Description

Simulate nodes using a random number generator supplied by the user, and combine these with a vector of equal weights into a list. Sparse grids can be created with the function createSparseGrid.

Usage

```
createMonteCarloGrid( rng, dimension, num.sim, ... )
```

Arguments

`rng`	function that generates random numbers. The first argument of this function should be called `n`. Examples are the R built-in functions `rnorm` and `runif` for random numbers from a standard normal or uniform distribution.
`dimension`	dimension of the integration problem.
`num.sim`	number of simulated integration nodes.
`...`	arguments that will be passed to the random number generator `rng`.

Value

The return value contains a list with nodes and weights

`nodes`	matrix with a node in each row
`weights`	vector with corresponding weights

Author(s)

Jelmer Ypma

See Also

`createSparseGrid` `createProductRuleGrid` `createIntegrationGrid` `integrate` `pmvnorm`

Examples

```
# load library
library('SparseGrid')

# set random seed
set.seed( 3141 )

# Create Monte Carlo integration grids
# 1. with draws from a uniform distribution
mc.grid <- createMonteCarloGrid( runif, dimension=2, num.sim=10 )
mc.grid

# 2. with draws from a standard normal distribution
mc.grid <- createMonteCarloGrid( rnorm, dimension=3, num.sim=1000 )

# 3. with draws from a normal distribution with mean=2 and sd=5
mc.grid <- createMonteCarloGrid( rnorm, dimension=3, num.sim=1000, mean=2, sd=5 )
```

---

createProductRuleGrid *Create a multidimensional grid of nodes and weights for integration*

Description

Creates nodes and weights according to the product rule, combining 1D nodes and weights. Sparse grids can be created with the function createSparseGrid.

Usage

```
createProductRuleGrid(type, dimension, k, sym = FALSE)
```

Arguments

type	String or function for type of 1D integration rule, can take on values "KPU" Nested rule for unweighted integral over [0,1] "KPN" Nested rule for integral with Gaussian weight "GQU" Gaussian quadrature for unweighted integral over [0,1] (Gauss-Legendre) "GQN" Gaussian quadrature for integral with Gaussian weight (Gauss-Hermite) func any function. Function must accept level k and return a list with two elements nodes and weights for univariate quadrature rule with polynomial exactness 2k-1.
dimension	dimension of the integration problem.
k	Accuracy level. The rule will be exact for polynomial up to total order 2k-1.
sym	(optional) only used for own 1D quadrature rule (type not "KPU",...). If sym is supplied and not FALSE, the code will run faster but will produce incorrect results if 1D quadrature rule is asymmetric.

Value

The return value contains a list with nodes and weights

nodes	matrix with a node in each row
weights	vector with corresponding weights

Author(s)

Jelmer Ypma

References

Florian Heiss, Viktor Winschel, Likelihood approximation by numerical integration on sparse grids, Journal of Econometrics, Volume 144, Issue 1, May 2008, Pages 62-80, http://www.sparse-grids.de

See Also

createSparseGrid createMonteCarloGrid createIntegrationGrid integrate pmvnorm

Examples

```
# load library
library('SparseGrid')

# define function to be integrated
# g(x) = x[1] * x[2] * ... * x[n]
g <- function( x ) {
return( prod( x ) )
}

#
# Create sparse integration grid to approximate integral of a function with uniform weights
#
sp.grid <- createSparseGrid( 'KPU', dimension=3, k=5 )

# number of nodes and weights
length( sp.grid$weights )

# evaluate function g in nodes
gx.sp <- apply( sp.grid$nodes, 1, g )

# take weighted sum to get approximation for the integral
val.sp <- gx.sp %*% sp.grid$weights

#
# Create integration grid to approximate integral of a function with uniform weights
#
pr.grid <- createProductRuleGrid( 'KPU', dimension=3, k=5 )

# number of nodes and weights
```

```
length( pr.grid$weights )

# evaluate function g in nodes
gx.pr <- apply( pr.grid$nodes, 1, g )

# take weighted sum to get approximation for the integral
val.pr <- gx.pr %*% pr.grid$weights

#
# Create integration grid to approximation integral using Monte Carlo simulation
#
set.seed( 3141 )
mc.grid <- createMonteCarloGrid( runif, dimension=3, num.sim=1000 )

# number of nodes and weights
length( mc.grid$weights )

# evaluate function g in MC nodes
gx.mc   <- apply( mc.grid$nodes, 1, g )

# take weighted sum to get approximation for the integral
# the weights are all equal to 1/1000 in this case
val.mc <- gx.mc %*% mc.grid$weights

val.sp
val.pr
val.mc
```

`createSparseGrid` *Create sparse grid*

Description

Creates nodes and weights that can be used for sparse grid integration. Based on Matlab code by Florian Heiss and Viktor Winschel, available from http://www.sparse-grids.de

Usage

```
createSparseGrid(type, dimension, k, sym = FALSE)
```

Arguments

`type`	String or function for type of 1D integration rule, can take on values
	`"KPU"` Nested rule for unweighted integral over [0,1]
	`"KPN"` Nested rule for integral with Gaussian weight
	`"GQU"` Gaussian quadrature for unweighted integral over [0,1] (Gauss-Legendre)
	`"GQN"` Gaussian quadrature for integral with Gaussian weight (Gauss-Hermite)

	`func` any function. Function must accept level k and return a list with two elements `nodes` and `weights` for univariate quadrature rule with polynomial exactness 2k-1.
`dimension`	dimension of the integration problem.
`k`	Accuracy level. The rule will be exact for polynomial up to total order 2k-1.
`sym`	(optional) only used for own 1D quadrature rule (type not "KPU",...). If sym is supplied and not FALSE, the code will run faster but will produce incorrect results if 1D quadrature rule is asymmetric.

Value

The return value contains a list with nodes and weights

`nodes`	matrix with a node in each row
`weights`	vector with corresponding weights

Author(s)

Jelmer Ypma

References

Florian Heiss, Viktor Winschel, Likelihood approximation by numerical integration on sparse grids, Journal of Econometrics, Volume 144, Issue 1, May 2008, Pages 62-80, http://www.sparse-grids.de

See Also

`createProductRuleGrid` `createMonteCarloGrid` `createIntegrationGrid` `integrate` `pmvnorm`

Examples

```
# load library
library('SparseGrid')

# define function to be integrated
# g(x) = x[1] * x[2] * ... * x[n]
g <- function( x ) {
return( prod( x ) )
}

#
# Create sparse integration grid to approximate integral of a function with uniform weights
#
sp.grid <- createSparseGrid( 'KPU', dimension=3, k=5 )

# number of nodes and weights
length( sp.grid$weights )

# evaluate function g in nodes
```

```
gx.sp <- apply( sp.grid$nodes, 1, g )

# take weighted sum to get approximation for the integral
val.sp <- gx.sp %*% sp.grid$weights

#
# Create integration grid to approximate integral of a function with uniform weights
#
pr.grid <- createProductRuleGrid( 'KPU', dimension=3, k=5 )

# number of nodes and weights
length( pr.grid$weights )

# evaluate function g in nodes
gx.pr <- apply( pr.grid$nodes, 1, g )

# take weighted sum to get approximation for the integral
val.pr <- gx.pr %*% pr.grid$weights

#
# Create integration grid to approximation integral using Monte Carlo simulation
#
set.seed( 3141 )
mc.grid <- createMonteCarloGrid( runif, dimension=3, num.sim=1000 )

# number of nodes and weights
length( mc.grid$weights )

# evaluate function g in MC nodes
gx.mc   <- apply( mc.grid$nodes, 1, g )

# take weighted sum to get approximation for the integral
# the weights are all equal to 1/1000 in this case
val.mc <- gx.mc %*% mc.grid$weights

val.sp
val.pr
val.mc
```

`readASCGrid` *Read integration grid from file*

Description

This function reads nodes and weights with the format of the `.asc` files available from http://www.sparse-grids.de

Usage

```
readASCGrid(filename, dimension)
```

Arguments

filename	name of the file that you want to read. The extension should be included.
dimension	dimension of the grid that you want to read.

Value

The return value contains a list with nodes and weights

nodes	matrix with a node in each row
weights	vector with corresponding weights

Author(s)

Jelmer Ypma

References

Florian Heiss, Viktor Winschel, Likelihood approximation by numerical integration on sparse grids, Journal of Econometrics, Volume 144, Issue 1, May 2008, Pages 62-80, http://www.sparse-grids.de

See Also

createSparseGrid createProductRuleGrid createIntegrationGrid integrate pmvnorm

Examples

```
# load library
library('SparseGrid')

## Not run:
# read file (e.g. after downloading from www.sparse-grids.de)
ReadASCFile(filename='GQU_d3_l5.asc', dimension=3)

## End(Not run)
```

Index