Package 'volesti'

February 13, 2019

Type Package

License LGPL-3

Title Volume Approximation and Sampling of Convex Polytopes

Description Package provides an R interface for volesti C++ package. Volesti computes approximations of volume of polytopes given as a set of points or linear inequalities or Minkowski sum of segments (zonotopes). There are two algorithms for volume approximation as well as algorithms for sampling, rounding and rotating polytopes.

Version 0.0.0

Date 2018-09-05

BugReports https://github.com/vissarion/volume_approximation/issues

Depends Rcpp (>= 0.12.17)

Imports methods

LinkingTo Rcpp, RcppEigen, BH

Suggests testthat

Encoding UTF-8

RoxygenNote 6.1.1

R topics documented:

`copula1` *Construct a copula using uniform sampling from the unit simplex*

Description

Given two families of parallel hyperplanes (or a family of parallel hyperplanes and a family of concentric ellispoids centered at the origin) intersecting the canonical simplex, this function samples from the canonical simplex and construct an approximation of the bivariate probability distribution, called copula.

Usage

```
copula1(h1, h2, numSlices, N)
```

Arguments

`h1`	A d-dimensional vector that describes the direction of the first family of parallel hyperplanes.
`h2`	A d-dimensional vector that describes the direction of the second family of parallel hyperplanes.
`numSlices`	The number of the slices for the copula. Default value is 100.
`N`	The number of points to sample. Default value is $4 \cdot 10^6$.

Value

A $numSlices \times numSlices$ copula.

References

L. Cales, A. Chalkis, I.Z. Emiris, V. Fisikopoulos, "Practical volume computation of structured convex bodies, and an application to modeling portfolio dependencies and financial crises," Proc. of Symposium on Computational Geometry, Budapest, Hungary, *2018.*

Examples

```
# compute a copula for two families of parallel hyperplanes
h1 = runif(n = 10, min = 1, max = 1000)
h1 = h1 / 1000
h2=runif(n = 10, min = 1, max = 1000)
h2 = h2 / 1000
cop = copula1(h1=h1, h2=h2, numSlices = 10, N = 100000)
```

`copula2` *Construct a copula using uniform sampling from the unit simplex*

Description

Given two families of parallel hyperplanes (or a family of parallel hyperplanes and a family of concentric ellispoids centered at the origin) intersecting the canonical simplex, this function samples from the canonical simplex and construct an approximation of the bivariate probability distribution, called copula.

Usage

```
copula2(h, E, numSlices, N)
```

Arguments

`h`	A d-dimensional vector that describes the direction of the first family of parallel hyperplanes.
`E`	The $d \times d$ symmetric positive semidefine matrix that describes the family of concentric ellipsoids centered at the origin.
`numSlices`	The number of the slices for the copula. Default value is 100.
`N`	The number of points to sample. Default value is $4 \cdot 10^6$.

Value

A $numSlices \times numSlices$ copula.

References

L. Cales, A. Chalkis, I.Z. Emiris, V. Fisikopoulos, "Practical volume computation of structured convex bodies, and an application to modeling portfolio dependencies and financial crises," Proc. of Symposium on Computational Geometry, Budapest, Hungary, *2018.*

Examples

```
# compute a copula for a family of parallel hyperplanes and a family of conentric ellipsoids
h = runif(n = 10, min = 1, max = 1000)
h = h / 1000
E = replicate(10, rnorm(20))
E = cov(E)
cop = copula2(h=h, E=E, numSlices=10, N=100000)
```

exact_vol *Compute the exact volume of (a) a zonotope (b) an arbitrary simplex (c) a unit simplex (d) a cross polytope (e) a hypercube*

Description

Given a zonotope (as an object of class Zonotope), this function computes the sum of the absolute values of the determinants of all the $d \times d$ submatrices of the $m \times d$ matrix G that contains row-wise the segments that define the zonotope. For an arbitrary simplex that is given in V-representation this function computes the absolute value of the determinant formed by the simplex's points assuming it is shifted to the origin. For a d-dimensional unit simplex, hypercube or cross polytope this function computes the exact well known formulas.

Usage

```
exact_vol(P, body = NULL, Parameters = NULL)
```

Arguments

P	A zonotope or a simplex in V-representation.
body	A string that declares the type of the body for the exact sampling: a) 'simplex' for the unit simplex, b) 'cross' for the cross polytope, c) 'hypersphere' for the hypersphere, d) 'cube' for the unit cube.
Parameters	A list for the parameters of the methods: • dimension An integer that declares the dimension when exact sampling is enabled for a simplex or a hypersphere. • radius The radius of the d-dimensional hypersphere. Default value is 1.

Value

The exact volume of the zonotope

Examples

```
# compute the exact volume of a 5-dimensional zonotope defined by the Minkowski sum of 10 segments
Z = GenZonotope(5, 10)
vol = exact_vol(Z)

# compute the exact volume of a 2-d arbitrary simplex
V = matrix(c(2,3,-1,7,0,0),ncol = 2, nrow = 3, byrow = TRUE)
P = Vpolytope$new(V)
vol = exact_vol(P)

# compute the exact volume the 10-dimensional cross polytope
vol = exact_vol(body = "cross", Parameters = list("dimension" = 10))
```

fileToMatrix *function to get a ine file and returns a numerical matrix A*

Description

This function takes the path for an ine or an ext file and returns the corresponding numerical matrix and vector that are compatible with volesti package's functions.

Usage

```
fileToMatrix(path, zonotope)
```

Arguments

path	A string that containes the path to an ine or a ext file. The ine file desrcibes a H-polytope and ext file describes a V-polytope or a zonotope.
zonotope	A boolean parameter. It has to be TRUE when the path leads to an .ext file that describes a zonotope.

Value

If the path corresponds to an ine file then the return value is a list that containes elements "A" and "b", i.e. the numerical $m \times d$ matrix A and the numerical m-dimensional vector b, defining H-polytope P, s.t.: $Ax \leq b$. If it corresponds to an ext file (V-polytopes or zonotopes) then the return value is a $m \times d$ matrix that containes row-wise the vertices or the segments respectively.

Examples

```
# give the path to birk4.ine
path = system.file('extdata', package = 'volesti')
ListPoly = fileToMatrix(paste0(path,'/birk4.ine'))
```

GenCross *Generator function for cross polytopes*

Description

This function can be used to generate a d-dimensional cross polytope in H or V representation.

Usage

```
GenCross(dimension, repr)
```

Arguments

dimension	The dimension of the cross polytope.
repr	A string to declare the representation. It has to be 'H' for H-representation or 'V' for V-representation.

Value

A cross polytope in H or V-representation. For an H polytope the return value is a list with two elements: the "matrix" containing a $2^d \times d$ matrix A and the "vector" containing a 2^d -dimensional vector b, s.t. $Ax \leq b$. When the V-representation is chosen the return value is a $2d \times d$ matrix that containes the vertices row-wise.

Examples

```
# generate a 10-dimensional cross polytope in H-representation
PolyList = GenCross(10, 'H')

# generate a 15-dimension cross polytope in V-representation
PolyList = GenCross(15, 'V')
```

GenCube *Generator function for hypercubes*

Description

This function can be used to generate a d-dimensional Hypercube $[-1, 1]^d$ in H or V representation.

Usage

```
GenCube(dimension, repr)
```

Arguments

dimension	The dimension of the hypercube
repr	A string to declare the representation. It has to be 'H' for H-representation or 'V' for V-representation.

Value

A hypercube in H or V-representation. For an H polytope the return value is a list with two elements: the "matrix" containing a $2d \times d$ matrix A and the "vector" containing a $2d$-dimensional vector b, s.t. $Ax \leq b$. When the V-representation is chosen the return value is a $2^d \times d$ matrix that containes the vertices row-wise.

Examples

```
# generate a 10-dimensional hypercube in H-representation
PolyList = GenCube(10, 'H')

# generate a 15-dimension hypercube in V-representation
PolyList = GenCube(15, 'V')
```

GenProdSimplex *Generator function for product of simplices*

Description

This function can be used to generate a $2d$-dimensional polytope that is defined as the product of two d-dimensional unit simplices in H-representation.

Usage

```
GenProdSimplex(dimension)
```

Arguments

dimension	The dimension of the simplices.

Value

A polytope defined as the product of two unit simplices in H-representation. The retutn value is a list with two elements: the "matrix" containing a $(2d+1) \times 2d$ matrix A and the "vector" containing a $(2d + 1)$-dimensional vector b, s.t. $Ax \leq b$.

Examples

```
# generate a product of two 5-dimensional simplices.
PolyList = GenProdSimplex(5)
```

GenSimplex *Generator function for simplices*

Description

This function can be used to generate a d-dimensional unit simplex in H or V representation.

Usage

```
GenSimplex(dimension, repr)
```

Arguments

dimension	The dimension of the simplex.
repr	A string to declare the representation. It has to be 'H' for H-representation or 'V' for V-representation.

Value

A simplex in H or V-representation. For an H polytope the return value is a list with two elements: the "matrix" containing a $(d+1) \times d$ matrix A and the "vector" containing a $(d+1)$-dimensional vector b, s.t. $Ax \leq b$. When the V-representation is chosen the return value is a $(d+1) \times d$ matrix that containes the vertices row-wise.

Examples

```
# generate a 10-dimensional simplex in H-representation
PolyList = GenSimplex(10, 'H')

# generate a 20-dimensional simplex in V-representation
PolyList = GenSimplex(20, 'V')
```

GenSkinnyCube *Generator function for skinny hypercubes*

Description

This function can be used to generate a d-dimensional skinny hypercube only in H-representation.

Usage

```
GenSkinnyCube(dimension)
```

Arguments

dimension	The dimension of the skinny hypercube.

Value

A d-dimensional skinny hypercube in H-representation. The retutn value is a list with two elements: the "matrix" containing a $2d \times d$ matrix A and the "vector" containing a $2d$-dimensional vector b, s.t. $Ax \leq b$.

Examples

```
# generate a 10-dimensional skinny hypercube.
PolyList = GenSkinnyCube(10)
```

GenZonotope *Generator function for zonotopes*

Description

This function can be used to generate a d-dimensional zonotope defined by the Minkowski sum of m segments. We consider the $e_1, \ldots, e_d$ generators and $m-d$ random generators. Then we shift the zonotope in order to contain the origin. The origin is the center of symmetry as well. It might needs rounding before the volume approximation using SequenceOfBalls or CoolingGaussian algorithms.

Usage

```
GenZonotope(dimension, NumGen)
```

Arguments

dimension	The dimension of the zonotope.
NumGen	The number of segments that generate the zonotope.

Value

A $m \times d$ matrix that containes the m d-dimensional segments.

Examples

```
# generate a 10-dimensional zonotope defined by the Minkowski sum of 20 segments
zonotope = GenZonotope(10, 20)
```

InnerBall *Compute an inscribed ball of a convex polytope*

Description

For a H-polytope described by a $m \times d$ matrix A and a m-dimensional vector b, s.t.: $Ax \leq b$, this function computes the largest inscribed ball (Chebychev ball) by solving the corresponding linear program. For a V-polytope $d+1$ vertices that define a full dimensional simplex picked at random and the largest inscribed ball of the simplex is computed. For a zonotope P we compute the minimum r s.t.: $re_i \in P$ for all $i = 1, \ldots, d$. Then the ball centered at the origin with radius $r/\sqrt{d}$ is an inscribed ball.

Usage

```
InnerBall(P)
```

Arguments

P	A convex polytope. It is an object from class (a) HPolytope or (b) VPolytope or (c) Zonotope.

Value

A $d+1$-dimensional vector that describes the inscribed ball. The first d coordinates corresponds to the center of the ball and the last one to the radius.

Examples

```
# compute the Chebychev ball of a 2d unit simplex
P = GenSimplex(2,'H')
ball_vec = InnerBall(P)

# compute the Chebychev ball of 3-dimensional cube in V-representation
P = GenCube(3, 'V')
ball_vec = InnerBall(P)
```

Polytopes *Classes to construct convex polytopes (H, V or zonotopes)*

Description

C++ classes exposed in R by rcpp_module

Usage

```
Hpolytope$new(A,b) or Vpolytope$new(V) or Zonotope$new(G)
```

Examples

```
# Create a 2-d unit simplex in H-representation
A = matrix(c(-1,0,0,-1,1,1), ncol=2, nrow=3, byrow=TRUE)
b = c(0,0,1)
P = Hpolytope$new(A,b)

# Create a 3-d cube in V-representation
V = matrix(c(-1,1,-1,-1,-1,1,-1,1,1,-1,-1,-1,1,1,-1,1,-1,1,1,1,1,1,-1,-1), ncol=3, nrow=8, byrow=TRUE)
P = Vpolytope$new(V)

# Create a 2-d zonotope with 4 generators
G = matrix(c(1,0,0,1,-0.73,0.67,-0.25,0.96), ncol = 2, nrow = 4, byrow = TRUE)
P = Zonotope$new(G)
```

poly_gen *An internal Rccp function as a polytope generator*

Description

An internal Rccp function as a polytope generator

Usage

```
poly_gen(kind_gen, Vpoly_gen, dim_gen, m_gen)
```

Arguments

kind_gen	An integer to declare the type of the polytope.
Vpoly_gen	A boolean parameter to declare if the requested polytope has to be in V-representation.
dim_gen	An integer to declare the dimension of the requested polytope.
m_gen	An integer to declare the number of generators for the requested random zonotope

Value

A numerical matrix describing the requested polytope

rand_rotate *Apply a random rotation to a convex polytope (H-polytope, V-polytope or a zonotope)*

Description

Given a convex H or V polytope or a zonotope as input this function applies a random rotation.

Usage

```
rand_rotate(P)
```

Arguments

P	A convex polytope. It is an object from class (a) HPolytope or (b) VPolytope or (c) Zonotope.

Value

A random rotation of the polytope that is given as an input. The return class is the same as the input class.

Examples

```
# rotate a H-polytope (2d unit simplex)
P = GenSimplex(2,'H')
listHpoly = rand_rotate(P)

# rotate a V-polytope (3d cube)
P = GenCube(3, 'V')
matVpoly = rand_rotate(P)

# rotate a 5-dimensional zonotope defined by the Minkowski sum of 15 segments
Z = GenZonotope(3,6)
MatZono = rand_rotate(Z)
```

---

rotating *An internal Rccp function for the random rotation of a convex polytope*

Description

An internal Rccp function for the random rotation of a convex polytope

Usage

```
rotating(P)
```

Arguments

P	A convex polytope (H-, V-polytope or a zonotope).

Value

A matrix that describes the rotated polytope

---

rounding *Internal rcpp function for the rounding of a convex polytope.*

Description

Internal rcpp function for the rounding of a convex polytope.

Usage

```
rounding(P, WalkType = NULL, walk_length = NULL, radius = NULL)
```

Arguments

WalkType	Optional. A string that declares the random walk.
walk_length	Optional. The number of the steps for the random walk.
radius	Optional. The radius for the ball walk.

Value

A Matrix that describes the rounded polytope and contains the round value.

---

round_polytope *Apply rounding to a convex polytope (H-polytope, V-polytope or a zonotope)*

Description

Given a convex H or V polytope or a zonotope as input this function computes a rounding based on minimum volume enclosing ellipsoid of a pointset.

Usage

```
round_polytope(P, WalkType = NULL, walk_length = NULL, radius = NULL)
```

Arguments

P	A convex polytope. It is an object from class (a) Hpolytope or (b) Vpolytope or (c) Zonotope.
WalkType	Optional. A string that declares the random walk method: a) 'CDHR' for Coordinate Directions Hit-and-Run, b) 'RDHR' for Random Directions Hit-and-Run or c) 'BW' for Ball Walk. The default walk is 'CDHR'.
walk_length	Optional. The number of the steps for the random walk. The default value is $\lfloor 10 + d/10 \rfloor$.
radius	Optional. The radius for the ball walk.

Value

A list with 2 elements: (a) a polytope of the same class as the input polytope class and (b) the element "round_value" which is the determinant of the square matrix of the linear transformation that was applied on the polytope that is given as input.

Examples

```
# rotate a H-polytope (2d unit simplex)
A = matrix(c(-1,0,0,-1,1,1), ncol=2, nrow=3, byrow=TRUE)
b = c(0,0,1)
P = HPolytope$new(A, b)
listHpoly = round_polytope(P)

# rotate a V-polytope (3d cube) using Random Directions HnR with step equal to 50
P = GenCube(3, 'V')
ListVpoly = round_polytope(P, WalkType = 'RDHR', walk_length = 50)

# rotate a 4-dimensional zonotope defined by the Minkowski sum of 8 segments using ball walk with a radius equal t
Z = GenZonotope(4,8)
ListZono = round_polytope(Z, WalkType = 'BW', radius = 0.02)
```

`sample_points` *Sample many points from a convex Polytope (H-polytope, V-polytope or a zonotope) or use direct methods for uniform sampling from unit simplex and hypersphere*

Description

Sample N points from a H or a V-polytope or a zonotope with uniform or spherical gaussian - centered in an internal point- target distribution. The d-dimensional unit simplex is the set of points $\vec{x} \in \mathsf{R}^d$, s.t.: $\sum_i x_i \leq 1, x_i \geq 0$. The d-dimensional canonical simplex is the set of points $\vec{x} \in \mathsf{R}^d$, s.t.: $\sum_i x_i = 1, x_i \geq 0$.

Usage

```
sample_points(P = NULL, N = NULL, distribution = NULL,
  WalkType = NULL, walk_length = NULL, exact = NULL, body = NULL,
  Parameters = NULL, InnerPoint = NULL)
```

Arguments

`P`	A convex polytope. It is an object from class (a) Hpolytope or (b) Vpolytope or (c) Zonotope.
`N`	The number of points that the function is going to sample from the convex polytope. The default value is 100.
`distribution`	Optional. A string that declares the target distribution: a) 'uniform' for uniform distribution or b) 'gaussian' for spherical multidimensional distribution. The default target distribution is uniform.
`WalkType`	Optional. A string that declares the random walk method: a) 'CDHR' for Coordinate Directions Hit-and-Run, b) 'RDHR' for Random Directions Hit-and-Run or c) 'BW' for Ball Walk. The default walk is 'CDHR'.
`walk_length`	Optional. The number of the steps for the random walk. The default value is $\lfloor 10 + d/10 \rfloor$.

`InnerPoint` A d-dimensional numerical vector that defines a point in the interior of polytope P.

`exact.` A boolean parameter. It should be used for uniform sampling from the boundary or the interior of a hypersphere centered at the origin or from a unit or an arbitrary simplex. The arbitrary simplex has to be given as a V-polytope. For the rest well known convex bodies it has to be declared the dimension and the type of body (simplex, sphere, ball) as well as the radius of the hypersphere.

`body.` A string that declares the type of the body for the exact sampling: a) 'unit simplex' for the unit simplex, b) 'canonical simplex' for the canonical simplex, c) 'hypersphere' for the boundary of a hypersphere centered at the origin, d) 'ball' for the interior of a hypersphere centered at the origin.

`Parameters.` A list for the parameters of the methods:

- variance The variance of the spherical multidimensional gaussian. The default value is 1.
- dimension An integer that declares the dimension when exact sampling is enabled for a simplex or a hypersphere.
- radius The radius of the d-dimensional hypersphere. Default value is 1.
- BW_rad The radius for the ball walk.

Value

A $d \times N$ matrix that contains, column-wise, the sampled points from the convex polytope.

References

R.Y. Rubinstein and B. Melamed, "Modern simulation and modeling" Wiley Series in Probability and Statistics, *1998.*

A Smith, Noah and W Tromble, Roy, "Sampling Uniformly from the Unit Simplex," Center for Language and Speech Processing Johns Hopkins University, *2004.*

Art B. Owen, "Monte Carlo theory, methods and examples," Copyright Art Owen, *2009-2013.*

Examples

```
# uniform distribution from a 3d cube in V-representation using ball walk
P = GenCube(3, 'V')
points = sample_points(P, WalkType = "BW", walk_length = 5)

# gaussian distribution from a 2d unit simplex in H-representation with variance = 2
A = matrix(c(-1,0,0,-1,1,1), ncol=2, nrow=3, byrow=TRUE)
b = c(0,0,1)
P = Hpolytope$new(A,b)
points = sample_points(P, distribution = "gaussian", Parameters = list("variance" = 2))

# uniform points from the boundary of a 10-dimensional hypersphere with radius 5
points = sample_points(exact = TRUE, body = "hypersphere", Parameters = list("dimension" = 10, "radius" = 5))

# 10000 uniform points from a 2-d arbitrary simplex
V = matrix(c(2,3,-1,7,0,0),ncol = 2, nrow = 3, byrow = TRUE)
```

```
P = Vpolytope$new(V)
points = sample_points(P, N = 10000, exact = TRUE)
```

`SliceOfSimplex` *Compute the percentage of the volume of the unit simplex that is contained in the intersection of a half-space and the unit simplex*

Description

When a half-space H is given as a pair of a vector $a \in R^d$ and a scalar $z0 \in R$ s.t.: $a^T x \leq z0$ this function calls the Ali's version of the Varsi formula.

Usage

```
SliceOfSimplex(a, z0)
```

Arguments

`a`	A d-dimensional vector that defines the direction of the hyperplane.
`z0`	The scalar that defines the half-space.

Value

The percentage of the volume of the unit simplex that is contained in the intersection of the given half-space and the unit simplex.

References

Varsi, Giulio, "The multidimensional content of the frustum of the simplex," Pacific J. Math. 46, no. 1, 303–314, *1973.*

Ali, Mir M., "Content of the frustum of a simplex," Pacific J. Math. 48, no. 2, 313–322, *1973.*

Examples

```
# compute the frustum of H: -x1+x2<=0
a=c(-1,1)
z0=0
frustum = sliceOfSimplex(a, z0)
```

volume — *The main function for volume approximation of a convex Polytope (H-polytope, V-polytope or a zonotope)*

Description

For the volume approximation can be used two algorithms. Either SequenceOfBalls or CoolingGaussian. A H-polytope with m facets is described by a $m \times d$ matrix A and a m-dimensional vector b, s.t.: $Ax \leq b$. A V-polytope is described as a set of d-dimensional points. A zonotope is desrcibed by the Minkowski sum of d-dimensional segments.

Usage

```
volume(P, walk_length = NULL, error = NULL, InnerBall = NULL,
  Algo = NULL, WalkType = NULL, rounding = NULL, Parameters = NULL)
```

Arguments

P	A convex polytope. It is an object from class (a) Hpolytope or (b) Vpolytope or (c) Zonotope.
walk_length	Optional. The number of the steps for the random walk. Default value is $\lfloor 10 + d/10 \rfloor$ for SequenceOfBalls and 1 for CoolingGaussian.
error	Optional. Declare the upper bound for the approximation error. Default value is 1 for SequenceOfBalls and 0.1 for CoolingGaussian.
InnerBall	Optional. A $d + 1$ vector that containes an inner ball. The first d coordinates corresponds to the center and the last one to the radius of the ball. If it is not given then for H-polytopes the Chebychev ball is computed, for V-polytopes $d+1$ vertices are picked randomly and the Chebychev ball of the defined simplex is computed. For a zonotope that is defined by the Minkowski sum of m segments we compute the maximal r s.t.: $re_i \in Z$ for all $i = 1, \ldots, d$, then the ball centered at the origin with radius $r/\sqrt{d}$ is an inscribed ball.
Algo	Optional. A string that declares which algorithm to use: a) 'SoB' for SequenceOfBalls or b) 'CG' for CoolingGaussian.
WalkType	Optional. A string that declares the random walk method: a) 'CDHR' for Coordinate Directions Hit-and-Run, b) 'RDHR' for Random Directions Hit-and-Run or c) 'BW' for Ball Walk. The default walk is 'CDHR'.
rounding	Optional. A boolean parameter for rounding. Default value is FALSE.
Parameters	Optional. A list for the parameters of the algorithms: • Window The length of the sliding window for CG algorithm. The default value is $500 + 4dimension^2$. • C A constant for the lower bound of $variance/mean^2$ in schedule annealing of CG algorithm. The default value is 2. • N The number of points we sample in each step of schedule annealing in CG algorithm. The default value is $500C + dimension^2/2$.

- ratio Parameter of schedule annealing of CG algorithm, larger ratio means larger steps in schedule annealing. The default value is $1 - 1/dimension$.
- frac The fraction of the total error to spend in the first gaussian in CG algorithm. The default value is 0.1.
- BW_radThe radius for the ball walk. The default value is $4r/dimension$, where r is the radius of the inscribed ball of the polytope.

Value

The approximation of the volume of a convex polytope.

References

I.Z.Emiris and V. Fisikopoulos, "Practical polytope volume approximation," ACM Trans. Math. Soft., *2014.*,

B. Cousins and S. Vempala, "A practical volume algorithm," Springer-Verlag Berlin Heidelberg and The Mathematical Programming Society, *2015.*

Examples

```
# calling SOB algorithm for a H-polytope (2d unit simplex)
P = GenSimplex(2,'H')
vol = volume(P)

# calling CG algorithm for a V-polytope (3d cube)
P = GenSimplex(2,'V')
vol = volume(P, Algo = list("CG"=TRUE))

# calling CG algorithm for a 5-dimensional zonotope defined as the Minkowski sum of 10 segments
Z = GenZonotope(2, 4)
vol = volume(Z, WalkType = list("method"="hnr", "coordinate"=FALSE, "W"=5), rounding=TRUE)
```

Index