Package 'volesti'

August 10, 2018

Type Package

License GPL (>= 2)

Title Volume approximation using VolEsti and CV algorithms.

Description Package provides C++ code and a Rcpp interface for volume approxiation. The main function takes as input a H-polytope or a V-polytope and apply VolEsti or CV algorithm.

Maintainer Fisikopoulos Vissarion <vissarion.fysikopoulos@oracle.com>, Chalkis Apostolos <tolis.chal@gmail.com>

Version 1.0

Date 2018-08-08

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

SystemRequirements C++11

Depends Rcpp (>= 0.12.17), RcppEigen, lpSolveAPI, BH

Imports Rcpp (>= 0.12.17)

LinkingTo Rcpp, RcppEigen, BH

RoxygenNote 6.0.1

Author Fisikopoulos Vissarion [cph, cre, aut], Chalkis Apostolos [cph, ctb, aut] (Contribution and development, as part of Google Summer of Code 2018 program)

RemoteType local

RemoteUrl /home/tolis/volume_approximation/R-proj

RemoteSha 1.0

R topics documented:

`CheBall` *Compute the Chebychev ball of a H-polytope.*

Description

For a H-polytope described by a $m \times d$ matrix A and a d-dimensional vector b, s.t.: $Ax \leq b$, this function computes the largest inscribed ball of that polytope by solving the corresponding linear program.

Usage

```
CheBall(A, b)
```

Arguments

`A`	the matrix of the H-polytope.
`b`	The d-dimensional vector b that containes the constants of the facets.

Value

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

Examples

```
#compute the Chebychev ball of a 2d unit simplex
A = matrix(c(-1,0,0,-1,1,1), ncol=2, nrow=3, byrow=TRUE)
b = c(0,0,1)
ball_vec = CheBall(A,b)
```

demoSampling *Run some sampling experiments.*

Description

Use uniform or spherical gaussian to sample from some convex H-polytopes, i.e. cubes, simplices, skinny_cubes, cross polytopes, birkhoff polytopes. We use the default values, i.e. $walklength = \lfloor 10 + dimension/10 \rfloor$, $N = 100$, Cordinate Directions HnR, $variance = 1$.

Usage

```
demoSampling(distribution)
```

Arguments

uniform	The string "uniform" to choose uniform as the target distribution.
gaussian	The string "gaussian" to choose spherical gaussian as the target distribution.

Value

Print the computed volumes and the error. If the test fails a message is printed.

Examples

```
#choose uniform distribution
demoSampling("uniform")
#choose spherical gaussian distribution
demoSampling("gaussian")
```

demoVolume *Run some volume approxiamtion experiments.*

Description

Run volesti or CV algorithm to approximate the volume of some cubes, simplices, skinny_cubes, cross polytopes, birkhoff polytopes. We run 10 experiments for volesti and 20 for CV. We demand $error = 0.1$. For all the other parameters use the default values for both algorithms.

Usage

```
demoVolume(algo)
```

Arguments

CV	The string "CV" to choose CV algorithm.
volesti	The string "volesti" to choose volesti algorithm.

Value

Print the computed volumes and the error. If the test fails a message is printed.

Examples

```
#test volesti
demoVolume("volesti")
#test CV
demoVolume("CV")
```

`ineToMatrix` *function to get a ine file and returns a numerical matrix A.*

Description

This function takes an ine file as a string (using read.csv()) and returns a numerical matrix A in ine format for function volume (see $volume$ function examples).

Usage

```
ineToMatrix(P)
```

Arguments

`P`	It is in format, read.cs('path/to/file.ine'). The ine file desrcibes a H-polytope.

Value

The numerical matrix in ine format.

Examples

```
#give the path to cube40.ine
A = ineToMatrix(read.csv('path/to/data/cube40.ine'))
```

`modifyMat` *Takes a numerical matrix in ine format and returns the matrix A and the vector b.*

Description

This function can be used to extract from a numerical matrix in ine format (see example), that describes a H-polytope, the $m \times d$ matrix A and the d-dimensional vector b, s.t.: $Ax \leq b$.

Usage

```
modifyMat(A)
```

Arguments

`A`	The numerical matrix in ine format (see example) of the H-polytope.

Value

A list that contains the numerical $m \times d$ matrix A and the numerical d-dimensional vector b, defining H-polytope P, s.t.: $Ax \leq b$.

Examples

```
# a 2d unit simplex in H-representation using numerical matrix in ine format
A = matrix(c(3,3,0,0,-1,0,0,0,-1,1,1,1), ncol=3, nrow=4, byrow=TRUE)
list_of_matrix_and_vector = modifyMat(A)
```

`rand_rotate` *Apply a random rotation to a convex H or V-polytope.*

Description

Give a convex H or V polytope as input and then a random rotation is computed and be applied to the polytope.

Usage

```
rand_rotate(Inputs)
```

Arguments

`list("argument"=value)`	A list that includes parameters for the chosen algorithm.
`path`	The path to an ine or ext file that describes the H or V polytope respectively. If path is given then "matrix" and "vector" inputs are not needed.
`matrix`	The matrix of the H polytope or the matrix that contains all the vertices of a V polytope row-wise. If the matrix is in ine file, for H-polytopes only (see examples), then the "vector" input is not needed.
`vector`	Only for H-polytopes. The d-dimensional vector b that containes the constants of the facets.
`vpoly`	A boolean parameter, has to be true when a V-polytope is given as input. Default value is false.
`verbose`	Optional. A boolean parameter for printing. Default is false.

Value

A H or V-polytope which is a random rotation of the polytope that is given as an input. The output for a H-polytope is a list that containes elements "matrix" and "vector". For a V-polytope the output is a $k \times d$ matrix that contains the k vertices of the V polytope row-wise.

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)
listHpoly = rand_rotate(list("matrix"=A, "vector"=b))

#rotate a V-polytope (3d cube)
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)
matVpoly = rand_rotate(list("matrix"=V, "Vpoly"=TRUE))
```

---

round_polytope *Apply rounding to a convex H or V-polytope.*

Description

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

Usage

```
round_polytope(Inputs)
```

Arguments

list("argument"=value)	A list that includes parameters for the rounding.
path	The path to an ine or ext file that describes the H or V polytope respectively. If path is given then "matrix" and "vector" inputs are not needed.
matrix	The matrix of the H polytope or the matrix that contains all the vertices of a V polytope row-wise. If the matrix is in ine file, for H-polytopes only (see examples), then the "vector" input is not needed.
vector	Only for H-polytopes. The d-dimensional vector b that containes the constants of the facets.
vpoly	A boolean parameter, has to be true when a V-polytope is given as input. Default value is false.
ball_walk	Optional. Boolean parameter to use ball walk, only for CV algorithm .Default value is false.
delta	Optional. The radius for the ball walk.
coordinate	Optional. A boolean parameter for the hit-and-run. True for Coordinate Directions HnR, false for Random Directions HnR. Default value is true.
verbose	Optional. A boolean parameter for printing. Default is false.

Value

A H or V-polytope which is a random rotation of the polytope that is given as an input. The output for a H-polytope is a list that containes elements "matrix" and "vector". For a V-polytope the output is a $k \times d$ matrix that contains the k vertices of the V polytope row-wise.

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)
listHpoly = round_polytope(list("matrix"=A, "vector"=b))

#rotate a V-polytope (3d cube) using Random Directions HnR
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)
matVpoly = round_polytope(list("matrix"=V, "Vpoly"=TRUE, "coordinate"=FALSE))
```

---

sample_points *Sample points from a convex Polytope*

Description

Sample N points from a H or a V-polytope with uniform or spherical gaussian target distribution.

Usage

```
sample_points(Inputs)
```

Arguments

list("argument"=value)	A list that includes parameters for the chosen target distribution and the random walk algorithm.
path	The path to an ine or ext file that describes the H or V polytope respectively. If path is given then "matrix" and "vector" inputs are not needed.
matrix	The matrix A of a H-polytope or the matrix V that contains all the vertices of a V polytope row-wise. If it is in ine format, only for H-polytopes, then the input "vector" is not needed.
vector	Only for H-polytopes. The d-dimensional vector b that containes the constants of the facets.
walk_length	Optional. The number of the steps for the random walk, default is $\lfloor 10 + d/10 \rfloor$.
internal_point	Optional. A d-dimensional vector that containes an internal point of the polytope.
gaussian	Optional. A boolean parameter to sample with gaussian target distribution. Default value is false.
variance	Optional. The variance for the spherical gaussian. Default value is 1.
N	The number of points that the function is going to sample from the convex polytope. Default value is 100.
ball_walk	Optional. Boolean parameter to use ball walk for the sampling. Default value is false.
delta	Optional. The radius for the ball walk.
verbose	Optional. A boolean parameter for printing. Default is false.
vpoly	A boolean parameter, has to be true when a V-polytope is given as input. Default value is false.
coordinate	Optional. A boolean parameter for the hit-and-run. True for Coordinate Directions HnR, false for Random Directions HnR. Default value is true.

Value

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

Examples

```
#uniform distribution from a 3d cube described by a set of vertices
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)
points = sample_points(list("matrix"=V, "Vpoly"=TRUE, "N"=1000))

#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)
points = sample_points(list("matrix"=A, "vector"=b, "gaussian"=TRUE, "variance"=2))
```

volume — *The main R function for volume approximation of a convex H or V Polytope*

Description

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

Usage

```
volume(Inputs)
```

Arguments

`list("argument"=value)`	A list that includes parameters for the chosen algorithm.
`path`	The path to an ine or ext file that describes the H or V polytope respectively. If path is given then "matrix" and "vector" inputs are not needed.
`matrix`	The matrix of the H polytope or the matrix that contains all the vertices of a V polytope row-wise. If the matrix is in ine file, for H-polytopes only (see examples), then the "vector" input is not needed.
`vector`	Only for H-polytopes. The d-dimensional vector b that containes the constants of the facets.
`walk_length`	Optional. The number of the steps for the random walk, default is $\lfloor 10 + d/10 \rfloor$.
`error`	Optional. Declare the goal for the approximation error. Default is 1 for volesti and 0.2 for CV.
`Chebychev`	Optional. A d+1 vector that containes the chebychev center. The first d coordinates corresponds to the center and the last one to the radius of the chebychev ball.
`annealing`	Optional. A boolean parameter to use CV algorithm. Default value is false.
`win_len`	Optional. The size of the window for the ratios' approximation in CV algorithm. Default value is $4\ dimension^2 + 500$.
`C`	Optional. a constant for the lower boud of $variance/mean^2$ in schedule annealing.
`N`	optional. The number of points we sample in each step of schedule annealing in CV algorithm. Default value is $500C + dimension^2/2$.
`ratio`	Optional. parameter of schedule annealing, larger ratio means larger steps in schedule annealing. Default value is $1 - 1/dimension$.
`frac`	Optional. the fraction of the total error to spend in the first gaussian. Default value is 0.1.

ball_walk	Optional. Boolean parameter to use ball walk, only for CV algorithm .Default value is false.
delta	Optional. The radius for the ball walk.
verbose	Optional. A boolean parameter for printing. Default is false.
vpoly	A boolean parameter, has to be true when a V-polytope is given as input. Default value is false.
coordinate	Optional. A boolean parameter for the hit-and-run. True for Coordinate Directions HnR, false for Random Directions HnR. Default value is true.
rounding	Optional. A boolean parameter to activate the rounding option. Default value is false.

Value

The approximation of the volume of a convex H or V 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 volesti algorithm for 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)
vol = volume(list("matrix"=A, "vector"=b))

# calling CV algorithm for a V-polytope (3d cube)
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)
vol = volume(list("matrix"=V, "annealing"=TRUE, "Vpoly"=TRUE))

# a 2d unit simplex in H-representation using ine format matrix, calling volesti algorithm
A = matrix(c(3,3,0,0,-1,0,0,0,-1,1,1,1), ncol=3, nrow=4, byrow=TRUE)
vol = volume(list("matrix"=A))
```

Index