Package: expQ2 (via r-universe)

September 6, 2024

Type Package

Title Non-negative vector * exp of rate matrix

Version 1.3

Date 2023-05-04

Author Chris Sherlock

Maintainer Devin Johnson <devin.johnson@noaa.gov>

Description Calculate `v'exp(Q)` by uniformisation or scaling and squaring, or exp(Q) by scaling and squaring. This is a duplicate of the `expQ` package written by Chris Sherlock hosted at ``https://github.com/ChrisGSherlock/expQ" with changes made to pass R CRAN checks.

License GPL (>= 2)

Imports Rcpp (>= 1.0.3), Matrix

LinkingTo Rcpp, RcppArmadillo, BH

RoxygenNote 7.2.3

Encoding UTF-8

NeedsCompilation yes

Repository https://dsjohnson.r-universe.dev

RemoteUrl https://github.com/dsjohnson/expQ2

RemoteRef HEAD

RemoteSha 53548786b618c0eb6f0580ac431311b1ff451bf4

Contents

expQ2-package *Non-negative vector * exp of rate matrix*

Description

v exp(Q) by uniformisation or scaling and squaring, or exp(Q) by scaling and squaring.

Package:	expQ2
Type:	Package
Version:	1.3
Date:	May 4, 2023
License:	CC0
LazyLoad:	yes

Note

This software package is based on the R package expQ developed by Chris Sherlock (https://github.com/ChrisGSherlock/expQ
This alternate version is maintained by scientists at the NOAA Fisheries Pacific Islands Fisheries Science Center and should be considered a fundamental research communication. The recommendations and conclusions presented here are those of the authors and this software should not be construed as official communication by NMFS, NOAA, or the U.S. Dept. of Commerce. In addition, reference to trade names does not imply endorsement by the National Marine Fisheries Service, NOAA. While the best efforts have been made to insure the highest quality, tools such as this are under constant development and are subject to change.

Author(s)

Chris Sherlock and Devin Johnson

Maintainer: Devin S. Johnson <devin.johnson@noaa.gov>

References

Sherlock, C. (2021). Direct statistical inference for finite Markov jump processes via the matrix exponential. Computational Statistics, 36(4), 2863-2887.

SS_exp_Q *Exponentiate a rate matrix*

Description

Exponentiates a whole rate matrix by making all elements non-negative, then scaling and squaring.

Usage

```
SS_exp_Q(Q, prec, renorm = TRUE)
```

Arguments

Q	Rate matrix (sparse or dense).
prec	Required precision - missing mass in the Poisson distribution.
renorm	Force elements of each row to sum to 1? Defaults to TRUE.

Value

exp(Q) A dense matrix.

Author(s)

Chris Sherlock

Examples

```
Qd <- matrix(nrow=2,ncol=2,data=c(-1,1,2,-2),byrow=TRUE); SS_exp_Q(Qd,1e-10)

library("Matrix")
d<-5; f<-0.3; ones<-rep(1,d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0
Qsum <- as.vector(Qs%*%ones)
diag(Qs) <- -Qsum
SS_exp_Q(Qs,1e-15)

## Not run:
M <- matrix(nrow=2,ncol=2,data=c(1,1,2,2),byrow=TRUE); SS_exp_Q(M,1e-10)
M <- matrix(nrow=2,ncol=2,data=c(1,-1,-2,2),byrow=TRUE); SS_exp_Q(M,1e-10)
SS_exp_Q(Qs,1.5)
SS_exp_Q(Qs,-2.0)

## End(Not run)
```

SS_v_exp_Q *Product of horizontal vector and exponential of a rate matrix*

Description

Evaluates v Exp(Q) by making all elements of Q non-negative, then scaling down by a power of two, and finally using a mixture of squaring and vector-matrix products.

Usage

```
SS_v_exp_Q(v, Q, prec, renorm = TRUE, checks = TRUE)
```

Arguments

v	Non-negative horizontal vector (dense).
Q	Rate matrix (sparse or dense).
prec	Required precision - missing mass in the Poisson distribution.
renorm	Force elements of each row to sum to 1? Defaults to TRUE.
checks	Perform sanity checks on the arguments? Defaults to TRUE.

Value

v exp(Q) A dense horizontal vector.

Author(s)

Chris Sherlock

Examples

```
v<-runif(2); v<-v/sum(v)
Qd <- matrix(nrow=2,ncol=2,data=c(-1,1,2,-2),byrow=TRUE); SS_v_exp_Q(t(v),Qd,1e-10)

library("Matrix")
d<-5; f<-0.3; ones<-rep(1,d); v<-runif(d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0
Qsum <- as.vector(Qs%*%ones)
diag(Qs) <- -Qsum
SS_v_exp_Q(t(v),Qs,1e-15)

## Not run:
v <- runif(2);
M <- matrix(nrow=2,ncol=2,data=c(1,1,2,2),byrow=TRUE); SS_v_exp_Q(t(v),M,1e-10)
M <- matrix(nrow=2,ncol=2,data=c(1,-1,-2,2),byrow=TRUE); SS_v_exp_Q(t(v),M,1e-10)
d<-5; f<-0.3; ones<-rep(1,d); v<-runif(d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0; Qsum <- as.vector(Qs%*%ones); diag(Qs) <- -Qsum
SS_v_exp_Q(v,Qs,1e-15)
SS_v_exp_Q(t(v),Qs,1.5)
SS_v_exp_Q(t(v),Qs,-2.0)
v=-runif(d)
SS_v_exp_Q(t(v),Qs,1e-15)

## End(Not run)
```

Unif_v_exp_Q *Product of horizontal vector and exponential of rate matrix*

Description

Evaluates v exp(Q) by making all elements of Q non-negative, then using the uniformisation method.

Usage

```
Unif_v_exp_Q(v, Q, prec, renorm = TRUE, t2 = TRUE, checks = TRUE)
```

Arguments

v	Non-negative horizontal vector (dense).
Q	Rate matrix (sparse or dense).
prec	Required precision - missing mass in the Poisson distribution.
renorm	Force elements of each row to sum to 1? Defaults to TRUE.
t2	Perform two-tailed truncation? Defaults to TRUE.
checks	Perform sanity checks on the arguments? Defaults to TRUE.

Value

v exp(Q) Dense horizontal vector.

Author(s)

Chris Sherlock

Examples

```
v <- runif(2); v <- v/sum(v)
Qd <- matrix(nrow=2,ncol=2,data=c(-1,1,2,-2),byrow=TRUE); Unif_v_exp_Q(t(v),Qd,1e-10)

library("Matrix")
d <- 5; f <- 0.3; ones <- rep(1,d); v <- runif(d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0
Qsum <- as.vector(Qs%*%ones)
diag(Qs) <- -Qsum
Unif_v_exp_Q(t(v),Qs,1e-15)

## Not run:
v <- runif(2);
M <- matrix(nrow=2,ncol=2,data=c(1,1,2,2),byrow=TRUE); Unif_v_exp_Q(t(v),M,1e-10)
M <- matrix(nrow=2,ncol=2,data=c(1,-1,-2,2),byrow=TRUE); Unif_v_exp_Q(t(v),M,1e-10)
d <- 5; f <- 0.3; ones <- rep(1,d); v <- runif(d)
Qs <- abs(rsparsematrix(d,d,f))
```

```
diag(Qs) <- 0; Qsum <- as.vector(Qs%*%ones); diag(Qs) <- -Qsum
Unif_v_exp_Q(v,Qs,1e-15)
Unif_v_exp_Q(t(v),Qs,1.5)
Unif_v_exp_Q(t(v),Qs,-2.0)
v <- -runif(d)
Unif_v_exp_Q(t(v),Qs,1e-15)

## End(Not run)
```

---

vT_exp_Q *Vector transpose * exponential of a rate matrix, all transposed.*

Description

Evaluates [v' exp(Q)]'; automatically chooses between scaling and squaring or uniformisation.

Usage

```
vT_exp_Q(v, Q, prec, renorm = TRUE, t2 = TRUE, checks = TRUE)
```

Arguments

v	Non-negative vertical vector (dense).
Q	Rate matrix (sparse or dense).
prec	Required precision - missing mass in the Poisson distribution.
renorm	Force elements of each row to sum to 1? Defaults to TRUE.
t2	Perform two-tailed truncation? Defaults to TRUE.
checks	Perform sanity checks on the arguments? Defaults to TRUE.

Value

[v' exp(Q)]' Dense vertical vector.

Author(s)

Chris Sherlock

Examples

```
v <- runif(2); v <- v/sum(v)
Qd <- matrix(nrow=2,ncol=2,data=c(-1,1,2,-2),byrow=TRUE); vT_exp_Q(v,Qd,1e-10)

library("Matrix")
d <- 5; f <- 0.3; ones <- rep(1,d); v <- runif(d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0; Qsum <- as.vector(Qs%*%ones); diag(Qs) <- -Qsum
vT_exp_Q(v,Qs,1e-15)
```

```
## Not run:
v <- runif(2);
M <- matrix(nrow=2,ncol=2,data=c(1,1,2,2),byrow=TRUE); vT_exp_Q(v,M,1e-10)
M <- matrix(nrow=2,ncol=2,data=c(1,-1,-2,2),byrow=TRUE); vT_exp_Q(v,M,1e-10)
d <- 5; f <- 0.3; ones <- rep(1,d); v <- runif(d)
Qs=abs(rsparsematrix(d,d,f))
diag(Qs) <- 0; Qsum <- as.vector(Qs%*%ones); diag(Qs) <- -Qsum
vT_exp_Q(t(v),Qs,1e-15)
vT_exp_Q(v,Qs,1.5)
vT_exp_Q(v,Qs,-2.0)
v=-runif(d)
vT_exp_Q(v,Qs,1e-15)

## End(Not run)
```

---

v_exp_Q *Product of horizontal vector and exponential of rate matrix*

Description

Evaluates v exp(Q); automatically chooses between scaling and squaring or uniformisation.

Usage

```
v_exp_Q(v, Q, prec, renorm = TRUE, t2 = TRUE, checks = TRUE)
```

Arguments

v	Non-negative horizontal vector (dense).
Q	Rate matrix (sparse or dense).
prec	Required precision - missing mass in the Poisson distribution.
renorm	Force elements of each row to sum to 1? Defaults to TRUE.
t2	Perform two-tailed truncation? Defaults to TRUE.
checks	Perform sanity checks on the arguments? Defaults to TRUE.

Value

v exp(Q) Dense horizontal vector.

Author(s)

Chris Sherlock

Examples

```
v <- runif(2); v <- v/sum(v)
Qd <- matrix(nrow=2,ncol=2,data=c(-1,1,2,-2),byrow=TRUE); v_exp_Q(t(v),Qd,1e-10)

library("Matrix")
d <- 5; f=0.3; ones <- rep(1,d); v <- runif(d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0
Qsum <- as.vector(Qs%*%ones)
diag(Qs) <- -Qsum
v_exp_Q(t(v),Qs,1e-15)

## Not run:
v <- runif(2);
M <- matrix(nrow=2,ncol=2,data=c(1,1,2,2),byrow=TRUE); v_exp_Q(t(v),M,1e-10)
M <- matrix(nrow=2,ncol=2,data=c(1,-1,-2,2),byrow=TRUE); v_exp_Q(t(v),M,1e-10)
d <- 5; f=0.3; ones <- rep(1,d); v <- runif(d)
Qs <- abs(rsparsematrix(d,d,f))
diag(Qs) <- 0; Qsum <- as.vector(Qs%*%ones); diag(Qs) <- -Qsum
v_exp_Q(v,Qs,1e-15)
v_exp_Q(t(v),Qs,1.5)
v_exp_Q(t(v),Qs,-2.0)
v <- -runif(d)
v_exp_Q(t(v),Qs,1e-15)

## End(Not run)
```

Index