Package 'sparseinv'

January 8, 2018

Type Package

Title Computation of the Sparse Inverse Subset

Version 0.1

Date 2017-07-17

Maintainer Andrew Zammit-Mangion <andrewzm@gmail.com>

Imports Matrix, spam

Description Creates a wrapper for the SuiteSparse routines that compute the Takahashi equations. These equations compute the elements of the inverse of a sparse matrix at locations where the (permuted) Cholesky factor is non-zero. The resulting matrix is known as a sparse inverse subset. Some helper functions (like the permuted Cholesky factorisation) are also implemented. Support for spam matrices is currently limited and will be implemented in the future.

BugReports http://github.com/andrewzm/sparseinv/issues

Depends R (>= 3.1)

License GPL (>= 2)

NeedsCompilation yes

LazyData true

RoxygenNote 6.0.1

Author Andrew Zammit-Mangion [aut, cre],
Timothy Davis [ctb],
Patrick Amestoy [ctb],
Iain Duff [ctb],
John K. Reid [ctb]

Archs i386, x64

R topics documented:

`sparseinv-package` *sparseinv*

Description

This package creates a wrapper for the SuiteSparse routines in C that us the Takahashi equations to compute the elements of the inverse of a sparse matrix at locations where the (permuted) Cholesky factor is non-zero. The resulting matrix is known as a sparse inverse subset. Some helper functions (like the permuted Cholesky factorisation) are also implemented. Support for spam matrices is currently limited and will be implemented in the future.

`cholPermute` *Sparse Cholesky Factorisation with fill-in reducing permutations*

Description

This function is similar to chol(A,pivot=T) when A is a sparse matrix. The fill-in reduction permutation is the approximate minimum degree permutation of Davis' SuiteSparse package configured to be slightly more aggressive than that in the Matrix package. If the Cholesky factor fails, the matrix is coerced to be symmetric.

Usage

```
cholPermute(Q, method = NULL)
```

Arguments

`Q`	matrix (sparse or dense), the Cholesky factor of which needs to be found
`method`	If "amd", Timothy Davis SuiteSparse algorithm is used, if not that in the R Matrix package is employed

Value

A list with two elements, Qpermchol (the permuted Cholesky factor) and P (the pivoting order matrix)

References

Havard Rue and Leonhard Held (2005). Gaussian Markov Random Fields: Theory and Applications. Chapman & Hall/CRC Press

Examples

```
require(Matrix)
cholPermute(sparseMatrix(i=c(1,1,2,2),j=c(1,2,1,2),x=c(0.1,0.2,0.2,1)))
```

`cholsolve` *Solve the equation Qx = y*

Description

This function is similar to `solve(Q,y)` but with the added benefit that it allows for permuted matrices. This function does the job in order to minimise user error when attempting to re-permute the matrices prior or after solving. The user also has an option for the permuted Cholesky factorisation of Q to be carried out internally.

Usage

```
cholsolve(Q, y, perm = F, cholQ = matrix(1, 0, 0), cholQp = matrix(1, 0,
  0), P = NA)
```

Arguments

`Q`	matrix (sparse or dense), the Cholesky factor of which needs to be found
`y`	matrix with the same number of rows as Q
`perm`	if F no permutation is carried out, if T permuted Cholesky factors are used
`cholQ`	the Cholesky factor of Q (if known already)
`cholQp`	the permuted Cholesky factor of Q (if known already)
`P`	the pivot matrix (if known already)

Value

x solution to Qx = y

References

Havard Rue and Leonhard Held (2005). Gaussian Markov Random Fields: Theory and Applications. Chapman & Hall/CRC Press

Examples

```
require(Matrix)
Q = sparseMatrix(i=c(1,1,2,2),j=c(1,2,1,2),x=c(0.1,0.2,0.2,1))
y = matrix(c(1,2),2,1)
cholsolve(Q,y)
```

cholsolveAQinvAT *Solve the equation X = AQ^-1t(A) under permutations*

Description

This function is a wrapper of solve() for finding `X = AQ^{-1}t(A)` when the permuted Cholesky factor of Q is known. #'

Usage

```
cholsolveAQinvAT(Q, A, Lp, P)
```

Arguments

`Q`	ignored (deprecated)
`A`	matrix
`Lp`	Permuted Cholesky factor of Q
`P`	the pivot matrix

Value

x solution to `X = AQ^{-1}t(A)`

References

Havard Rue and Leonhard Held (2005). Gaussian Markov Random Fields: Theory and Applications. Chapman & Hall/CRC Press

Examples

```
require(Matrix)
Q <- sparseMatrix(i=c(1,1,2,2),j=c(1,2,1,2),x=c(0.1,0.2,0.2,1))
X <- cholPermute(Q)
y <- matrix(c(1,2),2,1)
A <- y %*% t(y)
cholsolveAQinvAT(Q,A,X$Qpermchol,X$P)
```

densify *Densify with explicit zeroes*

Description

This function takes two sparse matrices and returns the first matrix padded with explicit zeros so that it is at least dense (probably denser) than the second matrix. This function only works with matrices of class Matrix #'

Usage

```
densify(A, B)
```

Arguments

A	object of class Matrix
B	object of class Matrix

Value

object of class Matrix

Examples

```
require(Matrix)
Q1 <- sparseMatrix(i=c(1,2,2),j=c(1,1,2),x=c(0.1,0.2,1))
Q2 <- sparseMatrix(i=c(1,1,2,2),j=c(1,2,1,2),x=c(0.1,0.3,0.2,1))
Q1dens <- densify(Q1,Q2)
Q1
Q1dens
```

---

symb *Return the symbolic representation of a Matrix*

Description

This function takes an object of class Matrix and returns the same Matrix with all elements replaced with 1 #'

Usage

```
symb(A)
```

Arguments

A	object of class Matrix

Value

object of class Matrix

Examples

```
require(Matrix)
Q <- sparseMatrix(i=c(1,2,2),j=c(1,1,2),x=c(0.1,0.2,1))
Qsymb <- symb(Q)
Qsymb
```

Takahashi_Davis *Takahashi equations*

Description

Computes the sparse inverse of a sparse matrix Q of using the Takahashi equations.

Usage

```
Takahashi_Davis(Q, return_perm_chol = 0, cholQp = matrix(0, 0, 0), P = 0,
  gc = 0)
```

Arguments

Q	precision matrix (sparse or dense)
return_perm_chol	if 1 returns the permuted Cholesky factor (not advisable for large systems)
cholQp	the permuted Cholesky factor of Q (if known already)
P	the pivot matrix (if known already)
gc	do garbage collection throughout (takes some time but useful for small memory machines)

Details

This function first computes the Cholesky factor of Q. The fill-in reduction permutation is the approximate minimum degree permutation (amd) of Timothy Davis' SuiteSparse package configured to be slightly more aggressive than that in the Matrix package. If the Cholesky factor fails, the matrix is coerced to be symmetric. The function then uses the Takahashi equations to compute the variances at the non-zero locations of the Cholesky factor from the factor itself. The equations themselves are implemented in C using the SparseSuite package of Timothy Davis.

Value

if return_perm_chol == 0, returns the partial matrix inverse of Q, where the non-zero elements correspond to those in the Cholesky factor. If !(return_perm_chol == 0), returns a list with three elements, S (the partial matrix inverse), Lp (the Cholesky factor of the permuted matrix) and P (the permutation matrix)

Note

This package is a wrapper for C functions implemented by Timothy Davis in SuiteSparse. The author of this package has done no work on the sparse inverse routines themselves and any acknowledgment should include one to SuiteSparse (see below for reference). The author of this package was made aware of this methodology by Botond Cseke.

References

Takahashi, K., Fagan, J., Chin, M.-S., 1973. Formation of a sparse bus impedance matrix and its application to short circuit study. 8th PICA Conf. Proc. June 4–6, Minneapolis, Minn.

Davis, T., 2011. SPARSEINV: A MATLAB toolbox for computing the sparse inverse subset using the Takahashi equations. URL http://faculty.cse.tamu.edu/davis/suitesparse.html

Examples

```
require(Matrix)
Q = sparseMatrix(i=c(1,1,2,2),j=c(1,2,1,2),x=c(0.1,0.2,0.2,1))
X <- cholPermute(Q)
S_partial = Takahashi_Davis(Q,cholQp = X$Qpermchol,P=X$P)
```

Index