Matrix Reference Manual

0.8-7

Generated by Doxygen 1.3.6-20040222

Mon May 31 14:04:36 2004

Contents

1 Matrix File Index

1.1 Matrix File List

Here is a list of all files with brief descriptions:

2 Matrix File Documentation

2.1 cscMatrix.c File Reference

```
#include "cscMatrix.h"
```

Functions

- SEXP csc_validate (SEXP x)
- SEXP csc_crossprod (SEXP x)
- SEXP csc_matrix_crossprod (SEXP x, SEXP y)
- SEXP csc_to_triplet (SEXP x)
- SEXP csc_to_matrix (SEXP x)
- SEXP csc_to_geMatrix (SEXP x)
- SEXP csc_to_imagemat (SEXP x)
- SEXP matrix_to_csc (SEXP A)
- SEXP triplet_to_csc (SEXP triplet)
- SEXP csc_getDiag (SEXP x)
- SEXP csc_transpose (SEXP x)

2.1.1 Function Documentation

2.1.1.1 SEXP csc_crossprod (SEXP *x*)

2.1.1.2 SEXP csc_getDiag (SEXP *x*)

2.1.1.3 SEXP csc_matrix_crossprod (SEXP *x*, SEXP *y*)

2.1.1.4 SEXP csc_to_geMatrix (SEXP *x*)

2.1.1.5 SEXP csc_to_imagemat (SEXP *x*)

2.1.1.6 SEXP csc_to_matrix (SEXP *x*)

2.1.1.7 SEXP csc_to_triplet (SEXP *x*)

2.1.1.8 SEXP csc_transpose (SEXP *x*)

2.1.1.9 SEXP csc_validate (SEXP *x*)

2.1.1.10 SEXP matrix_to_csc (SEXP *A*)

2.1.1.11 SEXP triplet_to_csc (SEXP *triplet*)

2.2 cscMatrix.h File Reference

```
#include <Rdefines.h>
#include "Mutils.h"
#include "taucs/taucs.h"
```

Functions

- SEXP csc_crossprod (SEXP x)
- SEXP csc_matrix_crossprod (SEXP x, SEXP y)
- SEXP csc_validate (SEXP x)
- SEXP csc_to_triplet (SEXP x)
- SEXP csc_to_matrix (SEXP x)
- SEXP csc_to_geMatrix (SEXP x)
- SEXP csc_to_imagemat (SEXP x)
- SEXP matrix_to_csc (SEXP A)
- SEXP triplet_to_csc (SEXP triplet)
- SEXP csc_getDiag (SEXP x)
- SEXP csc_transpose (SEXP x)

2.2.1 Function Documentation

2.2.1.1 SEXP csc_crossprod (SEXP *x*)

2.2.1.2 SEXP csc_getDiag (SEXP *x*)

2.2.1.3 SEXP csc_matrix_crossprod (SEXP *x*, SEXP *y*)

2.2.1.4 SEXP csc_to_geMatrix (SEXP *x*)

2.2.1.5 SEXP csc_to_imagemat (SEXP *x*)

2.2.1.6 SEXP csc_to_matrix (SEXP *x*)

2.2.1.7 SEXP csc_to_triplet (SEXP *x*)

2.2.1.8 SEXP csc_transpose (SEXP *x*)

2.2.1.9 SEXP csc_validate (SEXP *x*)

2.2.1.10 SEXP matrix_to_csc (SEXP *A*)

2.2.1.11 SEXP triplet_to_csc (SEXP *triplet*)

2.3 dense.c File Reference

```
#include "dense.h"
```

Functions

- int left_cyclic (double x[ ], int ldx, int j, int k, double cosines[ ], double sines[ ])
- SEXP getGivens (double x[ ], int ldx, int jmin, int rank)
- SEXP checkGivens (SEXP X, SEXP jmin, SEXP rank)
- SEXP lsq_dense_Chol (SEXP X, SEXP y)
- SEXP lsq_dense_QR (SEXP X, SEXP y)
- SEXP lapack_qr (SEXP Xin, SEXP tl)

2.3.1 Function Documentation

2.3.1.1 SEXP checkGivens (SEXP *X*, SEXP *jmin*, SEXP *rank*)

2.3.1.2 SEXP getGivens (double *x*[ ], int *ldx*, int *jmin*, int *rank*) `[static]`

2.3.1.3 SEXP lapack_qr (SEXP *Xin*, SEXP *tl*)

2.3.1.4 int left_cyclic (double *x*[ ], int *ldx*, int *j*, int *k*, double *cosines*[ ], double *sines*[ ]) `[static]`

Perform a left cyclic shift of columns j to k in the upper triangular matrix x, then restore it to upper triangular form with Givens rotations. The algorithm is based on the Fortran routine DCHEX from Linpack.

The lower triangle of x is not modified.

Parameters:

x Matrix stored in column-major order

ldx leading dimension of x

j column number (0-based) that will be shifted to position k

k last column number (0-based) to be shifted

cosines cosines of the Givens rotations

sines sines of the Givens rotations

Returns:

0 for success

2.3.1.5 SEXP lsq_dense_Chol (SEXP *X*, SEXP *y*)

2.3.1.6 SEXP lsq_dense_QR (SEXP *X*, SEXP *y*)

2.4 dense.h File Reference

```
#include "Rdefines.h"
#include "R_ext/Lapack.h"
```

Functions

- SEXP lsq_dense_Chol (SEXP X, SEXP y)
- SEXP lsq_dense_QR (SEXP X, SEXP y)
- SEXP lapack_qr (SEXP Xin, SEXP tl)

2.4.1 Function Documentation

2.4.1.1 SEXP lapack_qr (SEXP *Xin*, SEXP *tl*)

2.4.1.2 SEXP lsq_dense_Chol (SEXP *X*, SEXP *y*)

2.4.1.3 SEXP lsq_dense_QR (SEXP *X*, SEXP *y*)

2.5 factorizations.c File Reference

```
#include "factorizations.h"
```

Functions

- SEXP LU_validate (SEXP obj)
- SEXP Cholesky_validate (SEXP obj)
- SEXP SVD_validate (SEXP obj)

2.5.1 Function Documentation

2.5.1.1 SEXP Cholesky_validate (SEXP *obj*)

2.5.1.2 SEXP LU_validate (SEXP *obj*)

2.5.1.3 SEXP SVD_validate (SEXP *obj*)

2.6 factorizations.h File Reference

```
#include "Mutils.h"
```

Functions

- SEXP LU_validate (SEXP obj)
- SEXP Cholesky_validate (SEXP obj)
- SEXP SVD_validate (SEXP obj)

2.6.1 Function Documentation

2.6.1.1 SEXP Cholesky_validate (SEXP *obj*)

2.6.1.2 SEXP LU_validate (SEXP *obj*)

2.6.1.3 SEXP SVD_validate (SEXP *obj*)

2.7 flame.c File Reference

```
#include "flame.h"
#include "R_ext/Lapack.h"
#include "FLAME/FLAME.h"
```

Functions

- FLA_Obj ∗ R_to_FLA_copy (SEXP Ain)
- FLA_Obj ∗ R_to_FLA_inPlace (SEXP Ain)
- SEXP R_FLA_Init ()
- SEXP R_FLA_Finalize ()
- int FLA_Abort (char ∗msg, int line, char ∗fname)
- SEXP lsq_Chol_flame (SEXP Xin, SEXP yin)
- SEXP lsq_QR_flame (SEXP Xin, SEXP yin)

2.7.1 Function Documentation

2.7.1.1 int FLA_Abort (char ∗ *msg*, int *line*, char ∗ *fname*)

2.7.1.2 SEXP lsq_Chol_flame (SEXP *Xin*, SEXP *yin*)

2.7.1.3 SEXP lsq_QR_flame (SEXP *Xin*, SEXP *yin*)

2.7.1.4 SEXP R_FLA_Finalize ()

2.7.1.5 SEXP R_FLA_Init ()

2.7.1.6 FLA_Obj∗ R_to_FLA_copy (SEXP *Ain*)

2.7.1.7 FLA_Obj∗ R_to_FLA_inPlace (SEXP *Ain*)

2.8 flame.h File Reference

```
#include "Rdefines.h"
#include "R_ext/Lapack.h"
#include "FLAME/FLAME.h"
```

Defines

- #define RFLAME_CHOL_NB 104
- #define RFLAME_QR_NB 96

Functions

- FLA_Obj * R_to_FLA_copy (SEXP Ain)
- FLA_Obj * R_to_FLA_inPlace (SEXP Ain)
- SEXP R_FLA_Init ()
- SEXP R_FLA_Finalize ()
- SEXP lsq_Chol_flame (SEXP Xin, SEXP yin)

2.8.1 Define Documentation

2.8.1.1 #define RFLAME_CHOL_NB 104

2.8.1.2 #define RFLAME_QR_NB 96

2.8.2 Function Documentation

2.8.2.1 SEXP lsq_Chol_flame (SEXP *Xin*, SEXP *yin*)

2.8.2.2 SEXP R_FLA_Finalize ()

2.8.2.3 SEXP R_FLA_Init ()

2.8.2.4 FLA_Obj* R_to_FLA_copy (SEXP *Ain*)

2.8.2.5 FLA_Obj* R_to_FLA_inPlace (SEXP *Ain*)

2.9 geMatrix.c File Reference

```
#include "geMatrix.h"
```

Functions

- SEXP geMatrix_validate (SEXP obj)
- double get_norm (SEXP obj, char ∗typstr)
- SEXP geMatrix_norm (SEXP obj, SEXP type)
- double set_rcond (SEXP obj, char ∗typstr)
- SEXP geMatrix_rcond (SEXP obj, SEXP type)
- SEXP geMatrix_crossprod (SEXP x)
- SEXP geMatrix_geMatrix_crossprod (SEXP x, SEXP y)
- SEXP geMatrix_matrix_crossprod (SEXP x, SEXP y)
- SEXP geMatrix_getDiag (SEXP x)
- SEXP geMatrix_LU (SEXP x)
- SEXP geMatrix_determinant (SEXP x, SEXP logarithm)
- SEXP geMatrix_solve (SEXP a)
- SEXP geMatrix_geMatrix_mm (SEXP a, SEXP b)

2.9.1 Function Documentation

2.9.1.1 SEXP geMatrix_crossprod (SEXP *x*)

2.9.1.2 SEXP geMatrix_determinant (SEXP *x*, SEXP *logarithm*)

2.9.1.3 SEXP geMatrix_geMatrix_crossprod (SEXP *x*, SEXP *y*)

2.9.1.4 SEXP geMatrix_geMatrix_mm (SEXP *a*, SEXP *b*)

2.9.1.5 SEXP geMatrix_getDiag (SEXP *x*)

2.9.1.6 SEXP geMatrix_LU (SEXP *x*)

2.9.1.7 SEXP geMatrix_matrix_crossprod (SEXP *x*, SEXP *y*)

2.9.1.8 SEXP geMatrix_norm (SEXP *obj*, SEXP *type*)

2.9.1.9 SEXP geMatrix_rcond (SEXP *obj*, SEXP *type*)

2.9.1.10 SEXP geMatrix_solve (SEXP *a*)

2.9.1.11 SEXP geMatrix_validate (SEXP *obj*)

2.9.1.12 double get_norm (SEXP *obj*, char ∗ *typstr*) `[static]`

2.9.1.13 double set_rcond (SEXP *obj*, char ∗ *typstr*) `[static]`

2.10 geMatrix.h File Reference

```
#include <R_ext/Lapack.h>
#include "Mutils.h"
```

Functions

- SEXP geMatrix_validate (SEXP obj)
- SEXP geMatrix_norm (SEXP obj, SEXP norm)
- SEXP geMatrix_crossprod (SEXP x)
- SEXP geMatrix_geMatrix_crossprod (SEXP x, SEXP y)
- SEXP geMatrix_matrix_crossprod (SEXP x, SEXP y)
- SEXP geMatrix_getDiag (SEXP x)
- SEXP geMatrix_LU (SEXP x)
- SEXP geMatrix_determinant (SEXP x, SEXP logarithm)
- SEXP geMatrix_solve (SEXP a)
- SEXP geMatrix_geMatrix_mm (SEXP a, SEXP b)

2.10.1 Function Documentation

2.10.1.1 SEXP geMatrix_crossprod (SEXP *x*)

2.10.1.2 SEXP geMatrix_determinant (SEXP *x*, SEXP *logarithm*)

2.10.1.3 SEXP geMatrix_geMatrix_crossprod (SEXP *x*, SEXP *y*)

2.10.1.4 SEXP geMatrix_geMatrix_mm (SEXP *a*, SEXP *b*)

2.10.1.5 SEXP geMatrix_getDiag (SEXP *x*)

2.10.1.6 SEXP geMatrix_LU (SEXP *x*)

2.10.1.7 SEXP geMatrix_matrix_crossprod (SEXP *x*, SEXP *y*)

2.10.1.8 SEXP geMatrix_norm (SEXP *obj*, SEXP *norm*)

2.10.1.9 SEXP geMatrix_solve (SEXP *a*)

2.10.1.10 SEXP geMatrix_validate (SEXP *obj*)

2.11 geMutils.c File Reference

```
#include "geMutils.h"
```

Functions

- char norm_type (char ∗typstr)
- char rcond_type (char ∗typstr)
- double get_double_by_name (SEXP obj, char ∗nm)
- SEXP set_double_by_name (SEXP obj, double val, char ∗nm)
- SEXP as_det_obj (double val, int log, int sign)
- SEXP get_factorization (SEXP obj, char ∗nm)
- SEXP set_factorization (SEXP obj, SEXP val, char ∗nm)
- SEXP Matrix_init (void)
- SEXP cscMatrix_set_Dim (SEXP x, int nrow)

2.11.1 Function Documentation

2.11.1.1 SEXP as_det_obj (double *val*, int *log*, int *sign*)

2.11.1.2 SEXP cscMatrix_set_Dim (SEXP *x*, int *nrow*)

2.11.1.3 double get_double_by_name (SEXP *obj*, char ∗ *nm*)

2.11.1.4 SEXP get_factorization (SEXP *obj*, char ∗ *nm*)

2.11.1.5 SEXP Matrix_init (void)

2.11.1.6 char norm_type (char ∗ *typstr*)

2.11.1.7 char rcond_type (char ∗ *typstr*)

2.11.1.8 SEXP set_double_by_name (SEXP *obj*, double *val*, char ∗ *nm*)

2.11.1.9 SEXP set_factorization (SEXP *obj*, SEXP *val*, char ∗ *nm*)

2.12 geMutils.h File Reference

```
#include <Rinternals.h>
#include <Rdefines.h>
```

Functions

- char norm_type (char ∗typstr)
- char rcond_type (char ∗typstr)
- double get_double_by_name (SEXP obj, char ∗nm)
- SEXP set_double_by_name (SEXP obj, double val, char ∗nm)
- SEXP as_det_obj (double val, int log, int sign)
- SEXP get_factorization (SEXP obj, char ∗nm)
- SEXP set_factorization (SEXP obj, SEXP val, char ∗nm)
- SEXP cscMatrix_set_Dim (SEXP x, int nrow)

Variables

- SEXP Matrix_DimSym
- SEXP Matrix_xSym
- SEXP Matrix_uploSym
- SEXP Matrix_diagSym
- SEXP Matrix_pSym
- SEXP Matrix_iSym
- SEXP Matrix_zSym

2.12.1 Function Documentation

2.12.1.1 SEXP as_det_obj (double *val*, int *log*, int *sign*)

2.12.1.2 SEXP cscMatrix_set_Dim (SEXP *x*, int *nrow*)

2.12.1.3 double get_double_by_name (SEXP *obj*, char ∗ *nm*)

2.12.1.4 SEXP get_factorization (SEXP *obj*, char ∗ *nm*)

2.12.1.5 char norm_type (char ∗ *typstr*)

2.12.1.6 char rcond_type (char ∗ *typstr*)

2.12.1.7 SEXP set_double_by_name (SEXP *obj*, double *val*, char ∗ *nm*)

2.12.1.8 SEXP set_factorization (SEXP *obj*, SEXP *val*, char ∗ *nm*)

2.12.2 Variable Documentation

2.12.2.1 SEXP Matrix_diagSym

2.12.2.2 SEXP Matrix_DimSym

2.12.2.3 SEXP Matrix_iSym

2.12.2.4 SEXP Matrix_pSym

2.12.2.5 SEXP Matrix_uploSym

2.12.2.6 SEXP Matrix_xSym

2.12.2.7 SEXP Matrix_zSym

2.13 ldl.c File Reference

```
#include "ldl.h"
```

Functions

- void ldl_symbolic (int n, int Ap[ ], int Ai[ ], int Lp[ ], int Parent[ ], int Lnz[ ], int Flag[ ], int P[ ], int Pinv[ ])
- int ldl_numeric (int n, int Ap[ ], int Ai[ ], double Ax[ ], int Lp[ ], int Parent[ ], int Lnz[ ], int Li[ ], double Lx[ ], double D[ ], double Y[ ], int Pattern[ ], int Flag[ ], int P[ ], int Pinv[ ])
- void ldl_lsolve (int n, double X[ ], int Lp[ ], int Li[ ], double Lx[ ])
- void ldl_dsolve (int n, double X[ ], double D[ ])
- void ldl_ltsolve (int n, double X[ ], int Lp[ ], int Li[ ], double Lx[ ])
- void ldl_perm (int n, double X[ ], double B[ ], int P[ ])
- void ldl_permt (int n, double X[ ], double B[ ], int P[ ])
- int ldl_valid_perm (int n, int P[ ], int Flag[ ])
- int ldl_valid_matrix (int n, int Ap[ ], int Ai[ ])

2.13.1 Function Documentation

2.13.1.1 void ldl_dsolve (int *n*, double *X*[ ], double *D*[ ])

2.13.1.2 void ldl_lsolve (int *n*, double *X*[ ], int *Lp*[ ], int *Li*[ ], double *Lx*[ ])

2.13.1.3 void ldl_ltsolve (int *n*, double *X*[ ], int *Lp*[ ], int *Li*[ ], double *Lx*[ ])

2.13.1.4 int ldl_numeric (int *n*, int *Ap*[ ], int *Ai*[ ], double *Ax*[ ], int *Lp*[ ], int *Parent*[ ], int *Lnz*[ ], int *Li*[ ], double *Lx*[ ], double *D*[ ], double *Y*[ ], int *Pattern*[ ], int *Flag*[ ], int *P*[ ], int *Pinv*[ ])

2.13.1.5 void ldl_perm (int *n*, double *X*[ ], double *B*[ ], int *P*[ ])

2.13.1.6 void ldl_permt (int *n*, double *X*[ ], double *B*[ ], int *P*[ ])

2.13.1.7 void ldl_symbolic (int *n*, int *Ap*[ ], int *Ai*[ ], int *Lp*[ ], int *Parent*[ ], int *Lnz*[ ], int *Flag*[ ], int *P*[ ], int *Pinv*[ ])

2.13.1.8 int ldl_valid_matrix (int *n*, int *Ap*[ ], int *Ai*[ ])

2.13.1.9 int ldl_valid_perm (int *n*, int *P*[ ], int *Flag*[ ])

2.14 ldl.h File Reference

Functions

- void ldl_symbolic (int n, int Ap[ ], int Ai[ ], int Lp[ ], int Parent[ ], int Lnz[ ], int Flag[ ], int P[ ], int Pinv[ ])
- int ldl_numeric (int n, int Ap[ ], int Ai[ ], double Ax[ ], int Lp[ ], int Parent[ ], int Lnz[ ], int Li[ ], double Lx[ ], double D[ ], double Y[ ], int Pattern[ ], int Flag[ ], int P[ ], int Pinv[ ])
- void ldl_lsolve (int n, double X[ ], int Lp[ ], int Li[ ], double Lx[ ])
- void ldl_dsolve (int n, double X[ ], double D[ ])
- void ldl_ltsolve (int n, double X[ ], int Lp[ ], int Li[ ], double Lx[ ])
- void ldl_perm (int n, double X[ ], double B[ ], int P[ ])
- void ldl_permt (int n, double X[ ], double B[ ], int P[ ])
- int ldl_valid_perm (int n, int P[ ], int Flag[ ])
- int ldl_valid_matrix (int n, int Ap[ ], int Ai[ ])

2.14.1 Function Documentation

2.14.1.1 void ldl_dsolve (int *n*, double *X*[ ], double *D*[ ])

2.14.1.2 void ldl_lsolve (int *n*, double *X*[ ], int *Lp*[ ], int *Li*[ ], double *Lx*[ ])

2.14.1.3 void ldl_ltsolve (int *n*, double *X*[ ], int *Lp*[ ], int *Li*[ ], double *Lx*[ ])

2.14.1.4 int ldl_numeric (int *n*, int *Ap*[ ], int *Ai*[ ], double *Ax*[ ], int *Lp*[ ], int *Parent*[ ], int *Lnz*[ ], int *Li*[ ], double *Lx*[ ], double *D*[ ], double *Y*[ ], int *Pattern*[ ], int *Flag*[ ], int *P*[ ], int *Pinv*[ ])

2.14.1.5 void ldl_perm (int *n*, double *X*[ ], double *B*[ ], int *P*[ ])

2.14.1.6 void ldl_permt (int *n*, double *X*[ ], double *B*[ ], int *P*[ ])

2.14.1.7 void ldl_symbolic (int *n*, int *Ap*[ ], int *Ai*[ ], int *Lp*[ ], int *Parent*[ ], int *Lnz*[ ], int *Flag*[ ], int *P*[ ], int *Pinv*[ ])

2.14.1.8 int ldl_valid_matrix (int *n*, int *Ap*[ ], int *Ai*[ ])

2.14.1.9 int ldl_valid_perm (int *n*, int *P*[ ], int *Flag*[ ])

2.15 LU.c File Reference

```
#include "LU.h"
```

Functions

- SEXP LU_expand (SEXP x)

2.15.1 Function Documentation

2.15.1.1 SEXP LU_expand (SEXP *x*)

2.16 LU.h File Reference

```
#include "trMatrix.h"
```

Functions

- SEXP LU_expand (SEXP x)

2.16.1 Function Documentation

2.16.1.1 SEXP LU_expand (SEXP *x*)

2.17 Metis_utils.c File Reference

```
#include "Metis_utils.h"
```

Functions

- void ssc_metis_order (int n, const int Tp[ ], const int Ti[ ], int Perm[ ], int iPerm[ ])

2.17.1 Function Documentation

2.17.1.1 void ssc_metis_order (int *n*, const int *Tp*[ ], const int *Ti*[ ], int *Perm*[ ], int *iPerm*[ ])

2.18 Metis_utils.h File Reference

```
#include <Rdefines.h>
#include "metis.h"
```

Functions

- void ssc_metis_order (int n, const int Tp[ ], const int Ti[ ], int perm[ ], int iperm[ ])

2.18.1 Function Documentation

2.18.1.1 void ssc_metis_order (int *n*, const int *Tp*[ ], const int *Ti*[ ], int *perm*[ ], int *iperm*[ ])

2.19 Mutils.c File Reference

```
#include "Mutils.h"
#include "triplet_to_col.h"
#include <R_ext/Lapack.h>
```

Functions

- SEXP Matrix_init (void)
- char norm_type (char *typstr)
- char rcond_type (char *typstr)
- double get_double_by_name (SEXP obj, char *nm)
- SEXP set_double_by_name (SEXP obj, double val, char *nm)
- SEXP as_det_obj (double val, int log, int sign)
- SEXP get_factorization (SEXP obj, char *nm)
- SEXP set_factorization (SEXP obj, SEXP val, char *nm)
- SEXP cscMatrix_set_Dim (SEXP x, int nrow)
- int csc_unsorted_columns (int ncol, const int p[ ], const int i[ ])
- void csc_sort_columns (int ncol, const int p[ ], int i[ ], double x[ ])
- SEXP csc_check_column_sorting (SEXP m)
- SEXP triple_as_SEXP (int nrow, int ncol, int nz, const int Ti[ ], const int Tj[ ], const double Tx[ ], char *Rclass)
- void csc_components_transpose (int m, int n, int nnz, const int xp[ ], const int xi[ ], const double xx[ ], int ap[ ], int ai[ ], double ax[ ])
- void ssc_symbolic_permute (int n, int upper, const int perm[ ], int Ap[ ], int Ai[ ])

- double ∗ nlme_symmetrize (double ∗a, const int nc)
- void nlme_check_Lapack_error (int info, const char ∗laName)
- double ∗ LMEgradient (const double ∗factor, const double ∗A, const int nlev, const int nc, const double ∗pdgradient, const int plen, double ∗value)
- SEXP nlme_replaceSlot (SEXP obj, SEXP names, SEXP value)
- SEXP nlme_weight_matrix_list (SEXP MLin, SEXP wts, SEXP adjst, SEXP MLout)

Variables

- SEXP Matrix_DSym
- SEXP Matrix_DIsqrtSym
- SEXP Matrix_DimSym
- SEXP Matrix_GpSym
- SEXP Matrix_LiSym
- SEXP Matrix_LpSym
- SEXP Matrix_LxSym
- SEXP Matrix_OmegaSym
- SEXP Matrix_ParentSym
- SEXP Matrix_RXXSym
- SEXP Matrix_RZXSym
- SEXP Matrix_XtXSym
- SEXP Matrix_ZtXSym
- SEXP Matrix_bVarSym
- SEXP Matrix_devianceSym
- SEXP Matrix_devCompSym
- SEXP Matrix_diagSym
- SEXP Matrix_iSym
- SEXP Matrix_ipermSym
- SEXP Matrix_jSym
- SEXP Matrix_matSym
- SEXP Matrix_ncSym
- SEXP Matrix_pSym
- SEXP Matrix_permSym
- SEXP Matrix_statusSym
- SEXP Matrix_uploSym
- SEXP Matrix_xSym
- SEXP Matrix_zSym

2.19.1 Function Documentation

2.19.1.1 SEXP as_det_obj (double *val*, int *log*, int *sign*)

2.19.1.2 SEXP csc_check_column_sorting (SEXP *m*)

Check for sorted columns in an object that inherits from the cscMatrix class. Resort the columns if necessary.

Parameters:

- ***m*** pointer to an object that inherits from the cscMatrix class

Returns:

m with the columns sorted by increasing row index

2.19.1.3 void csc_components_transpose (int *m*, int *n*, int *nnz*, const int *xp*[ ], const int *xi*[ ], const double *xx*[ ], int *ap*[ ], int *ai*[ ], double *ax*[ ])

2.19.1.4 void csc_sort_columns (int *ncol*, const int *p*[ ], int *i*[ ], double *x*[ ])

Sort the columns in a sparse column-oriented matrix so that each column is in increasing order of row index.

Parameters:

- ***ncol*** number of columns
- ***p*** column pointers
- ***i*** row indices
- ***x*** values of nonzero elements

2.19.1.5 int csc_unsorted_columns (int *ncol*, const int *p*[ ], const int *i*[ ])

Check for unsorted columns in the row indices

Parameters:

- ***ncol*** number of columns
- ***p*** column pointers
- ***i*** row indices

Returns:

0 if all columns are sorted, otherwise 1

2.19.1.6 SEXP cscMatrix_set_Dim (SEXP *x*, int *nrow*)

2.19.1.7 double get_double_by_name (SEXP *obj*, char ∗ *nm*)

2.19.1.8 SEXP get_factorization (SEXP *obj*, char ∗ *nm*)

2.19.1.9 double∗ LMEgradient (const double ∗ *factor*, const double ∗ *A*, const int *nlev*, const int *nc*, const double ∗ *pdgradient*, const int *plen*, double ∗ *value*)

Calculate the inner product of vec(nlev∗D^{-1} - A'A)/2 and the pdgradient array regarded as a nc∗nc by plen matrix. This calculation is used in several of the LMEgradient methods.

Parameters:

factor The nc by nc factor of the pdMat object

A The nc by nc matrix A from the LME decomposition.

nlev The number of groups associated with the random effect

nc The number of columns in the matrix

pdgradient A pdgradient object of dimension nc by nc by plen

value An array of length plen in which the gradient will be returned

Returns:

value, with the LME gradient

2.19.1.10 SEXP Matrix_init (void)

2.19.1.11 void nlme_check_Lapack_error (int *info*, const char ∗ *laName*)

Check the error code returned by an Lapack routine and create an appropriate error message.

Parameters:

info Error code as returned from the Lapack routine

laName Character string containing the name of the Lapack routine

2.19.1.12 SEXP nlme_replaceSlot (SEXP *obj*, SEXP *names*, SEXP *value*)

Replace the value of a slot or subslot of an object in place. This routine purposely does not copy the value of obj. Use with caution.

Parameters:

obj object with slot to be replaced

names vector of names. The last element is the name of the slot to replace. The leading elements are the names of slots and subslots of obj.

value the replacement value for the slot

Returns:

obj, with the named slot modified in place.

2.19.1.13 double∗ nlme_symmetrize (double ∗ *a*, const int *nc*)

Symmetrize a matrix by copying the strict upper triangle into the lower triangle.

Parameters:

a pointer to a matrix in Fortran storage mode

nc number of columns (and rows and leading dimension) in the matrix

Returns:

a, symmetrized

2.19.1.14 SEXP nlme_weight_matrix_list (SEXP *MLin*, SEXP *wts*, SEXP *adjst*, SEXP *MLout*)

Produce a weighted copy of the matrices in MLin in the storage allocated to MLout

Parameters:

MLin input matrix list

wts real vector of weights

adjst adjusted response

MLout On input a list of matrices of the same dimensions as MLin.

Returns:

MLout with its contents overwritten by a weighted copy of MLin according to wts with adjst overwriting the response.

2.19.1.15 char norm_type (char ∗ *typstr*)

2.19.1.16 char rcond_type (char ∗ *typstr*)

2.19.1.17 SEXP set_double_by_name (SEXP *obj*, double *val*, char ∗ *nm*)

2.19.1.18 SEXP set_factorization (SEXP *obj*, SEXP *val*, char ∗ *nm*)

2.19.1.19 void ssc_symbolic_permute (int *n*, int *upper*, const int *perm*[ ], int *Ap*[ ], int *Ai*[ ])

2.19.1.20 SEXP triple_as_SEXP (int *nrow*, int *ncol*, int *nz*, const int *Ti*[ ], const int *Tj*[ ], const double *Tx*[ ], char ∗ *Rclass*)

2.19.2 Variable Documentation

2.19.2.1 SEXP Matrix_bVarSym

2.19.2.2 SEXP Matrix_devCompSym

2.19.2.3 SEXP Matrix_devianceSym

2.19.2.4 SEXP Matrix_diagSym

2.19.2.5 SEXP Matrix_DimSym

2.19.2.6 SEXP Matrix_DIsqrtSym

2.19.2.7 SEXP Matrix_DSym

2.19.2.8 SEXP Matrix_GpSym

2.19.2.9 SEXP Matrix_ipermSym

2.19.2.10 SEXP Matrix_iSym

2.19.2.11 SEXP Matrix_jSym

2.19.2.12 SEXP Matrix_LiSym

2.19.2.13 SEXP Matrix_LpSym

2.19.2.14 SEXP Matrix_LxSym

2.19.2.15 SEXP Matrix_matSym

2.19.2.16 SEXP Matrix_ncSym

2.19.2.17 SEXP Matrix_OmegaSym

2.19.2.18 SEXP Matrix_ParentSym

2.19.2.19 SEXP Matrix_permSym

2.19.2.20 SEXP Matrix_pSym

2.19.2.21 SEXP Matrix_RXXSym

2.19.2.22 SEXP Matrix_RZXSym

2.19.2.23 SEXP Matrix_statusSym

2.19.2.24 SEXP Matrix_uploSym

2.19.2.25 SEXP Matrix_xSym

2.19.2.26 SEXP Matrix_XtXSym

2.19.2.27 SEXP Matrix_zSym

2.19.2.28 SEXP Matrix_ZtXSym

2.20 Mutils.h File Reference

```
#include <Rdefines.h>
```

Functions

- char norm_type (char ∗typstr)
- char rcond_type (char ∗typstr)
- double get_double_by_name (SEXP obj, char ∗nm)
- SEXP set_double_by_name (SEXP obj, double val, char ∗nm)
- SEXP as_det_obj (double val, int log, int sign)
- SEXP get_factorization (SEXP obj, char ∗nm)
- SEXP set_factorization (SEXP obj, SEXP val, char ∗nm)
- SEXP cscMatrix_set_Dim (SEXP x, int nrow)
- int csc_unsorted_columns (int ncol, const int p[ ], const int i[ ])
- void csc_sort_columns (int ncol, const int p[ ], int i[ ], double x[ ])
- SEXP triple_as_SEXP (int nrow, int ncol, int nz, const int Ti[ ], const int Tj[ ], const double Tx[ ], char ∗Rclass)
- SEXP csc_check_column_sorting (SEXP A)
- void csc_components_transpose (int m, int n, int nnz, const int xp[ ], const int xi[ ], const double xx[ ], int ap[ ], int ai[ ], double ax[ ])
- void triplet_to_col (int nrow, int ncol, int nz, const int Ti[ ], const int Tj[ ], const double Tx[ ], int Ap[ ], int Ai[ ], double Ax[ ])
- void ssc_symbolic_permute (int n, int upper, const int perm[ ], int Ap[ ], int Ai[ ])
- double ∗ nlme_symmetrize (double ∗a, const int nc)
- void nlme_check_Lapack_error (int info, const char ∗laName)
- double ∗ LMEgradient (const double ∗factor, const double ∗A, const int nlev, const int nc, const double ∗pdgradient, const int plen, double ∗value)
- SEXP nlme_replaceSlot (SEXP obj, SEXP names, SEXP value)
- SEXP nlme_weight_matrix_list (SEXP MLin, SEXP wts, SEXP adjst, SEXP MLout)

Variables

- SEXP Matrix_DSym
- SEXP Matrix_DIsqrtSym
- SEXP Matrix_DimSym
- SEXP Matrix_GpSym
- SEXP Matrix_LiSym
- SEXP Matrix_LpSym
- SEXP Matrix_LxSym
- SEXP Matrix_OmegaSym
- SEXP Matrix_ParentSym
- SEXP Matrix_RXXSym
- SEXP Matrix_RZXSym
- SEXP Matrix_XtXSym
- SEXP Matrix_ZtXSym

- SEXP Matrix_bVarSym
- SEXP Matrix_devianceSym
- SEXP Matrix_devCompSym
- SEXP Matrix_diagSym
- SEXP Matrix_iSym
- SEXP Matrix_ipermSym
- SEXP Matrix_jSym
- SEXP Matrix_matSym
- SEXP Matrix_ncSym
- SEXP Matrix_pSym
- SEXP Matrix_permSym
- SEXP Matrix_statusSym
- SEXP Matrix_uploSym
- SEXP Matrix_xSym
- SEXP Matrix_zSym

2.20.1 Function Documentation

2.20.1.1 SEXP as_det_obj (double *val*, int *log*, int *sign*)

2.20.1.2 SEXP csc_check_column_sorting (SEXP *m*)

Check for sorted columns in an object that inherits from the cscMatrix class. Resort the columns if necessary.

Parameters:

m pointer to an object that inherits from the cscMatrix class

Returns:

m with the columns sorted by increasing row index

2.20.1.3 void csc_components_transpose (int *m*, int *n*, int *nnz*, const int *xp*[ ], const int *xi*[ ], const double *xx*[ ], int *ap*[ ], int *ai*[ ], double *ax*[ ])

2.20.1.4 void csc_sort_columns (int *ncol*, const int *p*[ ], int *i*[ ], double *x*[ ])

Sort the columns in a sparse column-oriented matrix so that each column is in increasing order of row index.

Parameters:

ncol number of columns

p column pointers

i row indices

x values of nonzero elements

2.20.1.5 int csc_unsorted_columns (int *ncol*, const int *p*[ ], const int *i*[ ])

Check for unsorted columns in the row indices

Parameters:

ncol number of columns

p column pointers

i row indices

Returns:

0 if all columns are sorted, otherwise 1

2.20.1.6 SEXP cscMatrix_set_Dim (SEXP *x*, int *nrow*)

2.20.1.7 double get_double_by_name (SEXP *obj*, char ∗ *nm*)

2.20.1.8 SEXP get_factorization (SEXP *obj*, char ∗ *nm*)

2.20.1.9 double∗ LMEgradient (const double ∗ *factor*, const double ∗ *A*, const int *nlev*, const int *nc*, const double ∗ *pdgradient*, const int *plen*, double ∗ *value*)

Calculate the inner product of vec(nlev∗D^{-1} - A'A)/2 and the pdgradient array regarded as a nc∗nc by plen matrix. This calculation is used in several of the LMEgradient methods.

Parameters:

factor The nc by nc factor of the pdMat object

A The nc by nc matrix A from the LME decomposition.

nlev The number of groups associated with the random effect

nc The number of columns in the matrix

pdgradient A pdgradient object of dimension nc by nc by plen

value An array of length plen in which the gradient will be returned

Returns:

value, with the LME gradient

2.20.1.10 void nlme_check_Lapack_error (int *info*, const char ∗ *laName*)

Check the error code returned by an Lapack routine and create an appropriate error message.

Parameters:

info Error code as returned from the Lapack routine

laName Character string containing the name of the Lapack routine

2.20.1.11 SEXP nlme_replaceSlot (SEXP *obj*, SEXP *names*, SEXP *value*)

Replace the value of a slot or subslot of an object in place. This routine purposely does not copy the value of obj. Use with caution.

Parameters:

obj object with slot to be replaced

names vector of names. The last element is the name of the slot to replace. The leading elements are the names of slots and subslots of obj.

value the replacement value for the slot

Returns:

obj, with the named slot modified in place.

2.20.1.12 double∗ nlme_symmetrize (double ∗ *a*, const int *nc*)

Symmetrize a matrix by copying the strict upper triangle into the lower triangle.

Parameters:

a pointer to a matrix in Fortran storage mode

nc number of columns (and rows and leading dimension) in the matrix

Returns:

a, symmetrized

2.20.1.13 SEXP nlme_weight_matrix_list (SEXP *MLin*, SEXP *wts*, SEXP *adjst*, SEXP *MLout*)

Produce a weighted copy of the matrices in MLin in the storage allocated to MLout

Parameters:

MLin input matrix list

wts real vector of weights

adjst adjusted response

MLout On input a list of matrices of the same dimensions as MLin.

Returns:

MLout with its contents overwritten by a weighted copy of MLin according to wts with adjst overwriting the response.

2.20.1.14 char norm_type (char ∗ *typstr*)

2.20.1.15 char rcond_type (char ∗ *typstr*)

2.20.1.16 SEXP set_double_by_name (SEXP *obj*, double *val*, char ∗ *nm*)

2.20.1.17 SEXP set_factorization (SEXP *obj*, SEXP *val*, char ∗ *nm*)

2.20.1.18 void ssc_symbolic_permute (int *n*, int *upper*, const int *perm*[ ], int *Ap*[ ], int *Ai*[ ])

2.20.1.19 SEXP triple_as_SEXP (int *nrow*, int *ncol*, int *nz*, const int *Ti*[ ], const int *Tj*[ ], const double *Tx*[ ], char ∗ *Rclass*)

2.20.1.20 void triplet_to_col (int *nrow*, int *ncol*, int *nz*, const int *Ti*[ ], const int *Tj*[ ], const double *Tx*[ ], int *Ap*[ ], int *Ai*[ ], double *Ax*[ ])

2.20.2 Variable Documentation

2.20.2.1 SEXP Matrix_bVarSym

2.20.2.2 SEXP Matrix_devCompSym

2.20.2.3 SEXP Matrix_devianceSym

2.20.2.4 SEXP Matrix_diagSym

2.20.2.5 SEXP Matrix_DimSym

2.20.2.6 SEXP Matrix_DIsqrtSym

2.20.2.7 SEXP Matrix_DSym

2.20.2.8 SEXP Matrix_GpSym

2.20.2.9 SEXP Matrix_ipermSym

2.20.2.10 SEXP Matrix_iSym

2.20.2.11 SEXP Matrix_jSym

2.20.2.12 SEXP Matrix_LiSym

2.20.2.13 SEXP Matrix_LpSym

2.20.2.14 SEXP Matrix_LxSym

2.20.2.15 SEXP Matrix_matSym

2.20.2.16 SEXP Matrix_ncSym

2.20.2.17 SEXP Matrix_OmegaSym

2.20.2.18 SEXP Matrix_ParentSym

2.20.2.19 SEXP Matrix_permSym

2.20.2.20 SEXP Matrix_pSym

2.20.2.21 SEXP Matrix_RXXSym

2.20.2.22 SEXP Matrix_RZXSym

2.20.2.23 SEXP Matrix_statusSym

2.20.2.24 SEXP Matrix_uploSym

2.20.2.25 SEXP Matrix_xSym

2.20.2.26 SEXP Matrix_XtXSym

2.20.2.27 SEXP Matrix_zSym

2.20.2.28 SEXP Matrix_ZtXSym

2.21 pdDiag.c File Reference

```
#include "Mutils.h"
```

Functions

- double pdDiag_ld_factor_from_par (const double ∗par, double ∗factor, int nc)
- SEXP pdDiag_coefGets (SEXP x, SEXP value)
- SEXP pdDiag_LMEgradient (SEXP x, SEXP Ain, SEXP nlev)
- SEXP pdDiag_EMupdate (SEXP x, SEXP nlev, SEXP Ain)

2.21.1 Function Documentation

2.21.1.1 SEXP pdDiag_coefGets (SEXP *x*, SEXP *value*)

2.21.1.2 SEXP pdDiag_EMupdate (SEXP *x*, SEXP *nlev*, SEXP *Ain*)

2.21.1.3 double pdDiag_ld_factor_from_par (const double ∗ *par*, double ∗ *factor*, int *nc*) `[static]`

Populate the factor from the parameter vector and return the logarithm the determinant of the factor.

Parameters:

par vector of parameters

factor pointer to matrix to be overwritten with the factor

nc number of columns

Returns:

logarithm of the determinant of the factor

2.21.1.4 SEXP pdDiag_LMEgradient (SEXP *x*, SEXP *Ain*, SEXP *nlev*)

2.22 pdIdent.c File Reference

```
#include "Mutils.h"
```

Functions

- SEXP pdIdent_gradient (SEXP x, SEXP Ain, SEXP nlev)
- SEXP pdIdent_EMupdate (SEXP x, SEXP nlev, SEXP Ain)

2.22.1 Function Documentation

2.22.1.1 SEXP pdIdent_EMupdate (SEXP *x*, SEXP *nlev*, SEXP *Ain*)

2.22.1.2 SEXP pdIdent_gradient (SEXP *x*, SEXP *Ain*, SEXP *nlev*)

2.23 pdLogChol.c File Reference

```
#include "Mutils.h"
#include <R_ext/Lapack.h>
```

Functions

- double ld_factor_from_par (const double ∗par, double ∗factor, int nc)
- double ∗ gradient (const int nc, const double ∗factor, const double ∗pars, double ∗value)
- SEXP pdLogChol_LMEhessian (SEXP x, SEXP Ain, SEXP Hin, SEXP nlev)
- SEXP pdLogChol_LMEgradient (SEXP x, SEXP Ain, SEXP nlev)
- SEXP pdLogChol_pdgradient (SEXP x)
- SEXP pdLogChol_EMupdate (SEXP x, SEXP nlev, SEXP Ain)
- SEXP pdLogChol_coefGets (SEXP x, SEXP value)

2.23.1 Function Documentation

2.23.1.1 double∗ gradient (const int *nc*, const double ∗ *factor*, const double ∗ *pars*, double ∗ *value*) [static]

An internal function that calculates the gradient of the positive-definite matrix with respect to the parameters. This function is used in both pdLogChol_LMEgradient and pdLogChol_pdgradient.

Parameters:

nc number of columns (and rows) in the matrix

pars parameter vector of length nc∗(nc+1)/2

value array into which the results are written

Returns:

the gradient in value

2.23.1.2 double ld_factor_from_par (const double ∗ *par*, double ∗ *factor*, int *nc*) [static]

Populate the factor from the parameter vector and return the logarithm the determinant of the factor.

Parameters:

par vector of parameters

factor pointer to matrix to be overwritten with the factor

nc number of columns

Returns:

logarithm of the determinant of the factor

2.23.1.3 SEXP pdLogChol_coefGets (SEXP *x*, SEXP *value*)

2.23.1.4 SEXP pdLogChol_EMupdate (SEXP *x*, SEXP *nlev*, SEXP *Ain*)

Perform an EM update on a pdLogChol object.

Parameters:

x Pointer to a pdLogChol object

nlev An integer object - the number of levels in the grouping factor

Ain An upper triangular matrix object

Returns:

The updated pdLogChol object x

2.23.1.5 SEXP pdLogChol_LMEgradient (SEXP *x*, SEXP *Ain*, SEXP *nlev*)

LMEgradient implementation for the pdLogChol class

Parameters:

x Pointer to a pdLogChol object

Ain Pointer to an upper-triangular double precision square matrix

nlev Pointer to an integer scalar giving the number of levels

Returns:

Pointer to a REAL gradient vector

2.23.1.6 SEXP pdLogChol_LMEhessian (SEXP *x*, SEXP *Ain*, SEXP *Hin*, SEXP *nlev*)

2.23.1.7 SEXP pdLogChol_pdgradient (SEXP *x*)

Implementation of the pdgradient method for pdLogChol objects.

Parameters:

x Pointer to a pdLogChol object

Returns:

SEXP of a three-dimensional array with the gradient of the pdgradient with respect to the parameters.

2.24 pdMat.c File Reference

```
#include "Mutils.h"
#include <R_ext/Lapack.h>
```

Functions

- SEXP pdCompSymm_pdFactor (SEXP pd)
- SEXP nlme_Chol (SEXP A)

2.24.1 Function Documentation

2.24.1.1 SEXP nlme_Chol (SEXP *A*)

2.24.1.2 SEXP pdCompSymm_pdFactor (SEXP *pd*)

2.25 pdNatural.c File Reference

`#include "Mutils.h"`

Functions

- void corr_from_par (const double ∗par, double ∗corr, int nc)
- SEXP pdNatural_pdmatrix (SEXP x)
- SEXP pdNatural_corrmatrix (SEXP x)
- double ∗ gradient (int nc, const double ∗param, double ∗value)
- SEXP pdNatural_LMEgradient (SEXP x, SEXP Ain, SEXP nlev)

2.25.1 Function Documentation

2.25.1.1 void corr_from_par (const double ∗ *par*, double ∗ *corr*, int *nc*) `[static]`

2.25.1.2 double∗ gradient (int *nc*, const double ∗ *param*, double ∗ *value*) `[static]`

An internal function that calculates the gradient of the positive-definite matrix with respect to the parameters. This function is used in pdNatural_LMEgradient

Parameters:

nc number of columns (and rows) in the matrix

mat the positive definite matrix

value array into which the results are written

Returns:

the gradient in value

2.25.1.3 SEXP pdNatural_corrmatrix (SEXP *x*)

2.25.1.4 SEXP pdNatural_LMEgradient (SEXP *x*, SEXP *Ain*, SEXP *nlev*)

LMEgradient implementation for the pdNatural class

Parameters:

x Pointer to a pdNatural object

Ain Pointer to an upper-triangular double precision square matrix

nlev Pointer to an integer scalar giving the number of levels

Returns:

Pointer to a REAL gradient vector

2.25.1.5 SEXP pdNatural_pdmatrix (SEXP *x*)

Evaluate the pdMatrix from a pdNatural object

Parameters:

x Pointer to a pdNatural object

Returns:

A newly allocated matrix

2.26 poMatrix.c File Reference

```
#include "poMatrix.h"
```

Functions

- SEXP poMatrix_chol (SEXP x)
- double set_rcond (SEXP obj, char ∗typstr)
- SEXP poMatrix_rcond (SEXP obj, SEXP type)
- SEXP poMatrix_solve (SEXP x)
- SEXP poMatrix_geMatrix_solve (SEXP a, SEXP b)
- SEXP poMatrix_matrix_solve (SEXP a, SEXP b)

2.26.1 Function Documentation

2.26.1.1 SEXP poMatrix_chol (SEXP *x*)

2.26.1.2 SEXP poMatrix_geMatrix_solve (SEXP *a*, SEXP *b*)

2.26.1.3 SEXP poMatrix_matrix_solve (SEXP *a*, SEXP *b*)

2.26.1.4 SEXP poMatrix_rcond (SEXP *obj*, SEXP *type*)

2.26.1.5 SEXP poMatrix_solve (SEXP *x*)

2.26.1.6 double set_rcond (SEXP *obj*, char ∗ *typstr*) `[static]`

2.27 poMatrix.h File Reference

```
#include <R_ext/Lapack.h>
#include "Mutils.h"
```

Functions

- SEXP poMatrix_rcond (SEXP obj, SEXP type)
- SEXP poMatrix_solve (SEXP a)
- SEXP poMatrix_matrix_solve (SEXP a, SEXP b)
- SEXP poMatrix_geMatrix_solve (SEXP a, SEXP b)
- SEXP poMatrix_chol (SEXP x)
- double get_norm_sy (SEXP obj, char ∗typstr)

2.27.1 Function Documentation

2.27.1.1 double get_norm_sy (SEXP *obj*, char ∗ *typstr*)

2.27.1.2 SEXP poMatrix_chol (SEXP *x*)

2.27.1.3 SEXP poMatrix_geMatrix_solve (SEXP *a*, SEXP *b*)

2.27.1.4 SEXP poMatrix_matrix_solve (SEXP *a*, SEXP *b*)

2.27.1.5 SEXP poMatrix_rcond (SEXP *obj*, SEXP *type*)

2.27.1.6 SEXP poMatrix_solve (SEXP *a*)

2.28 sscChol.c File Reference

```
#include "sscChol.h"
```

Functions

- SEXP sscChol_validate (SEXP object)

2.28.1 Function Documentation

2.28.1.1 SEXP sscChol_validate (SEXP *object*)

2.29 sscChol.h File Reference

```
#include "tscMatrix.h"
```

Functions

- SEXP sscChol_validate (SEXP object)

2.29.1 Function Documentation

2.29.1.1 SEXP sscChol_validate (SEXP *object*)

2.30 sscCrosstab.c File Reference

```
#include "sscCrosstab.h"
```

Functions

- SEXP sscCrosstab (SEXP flist, SEXP upper)
- void col_metis_order (int j0, int j1, int i2, const int Tp[ ], const int Ti[ ], int ans[ ])
- SEXP sscCrosstab_groupedPerm (SEXP ctab)
- SEXP sscCrosstab_project (SEXP ctab)
- SEXP sscCrosstab_project2 (SEXP ctab)

2.30.1 Function Documentation

2.30.1.1 void col_metis_order (int *j0*, int *j1*, int *i2*, const int *Tp*[ ], const int *Ti*[ ], int *ans*[ ]) `[static]`

2.30.1.2 SEXP sscCrosstab (SEXP *flist*, SEXP *upper*)

2.30.1.3 SEXP sscCrosstab_groupedPerm (SEXP *ctab*)

2.30.1.4 SEXP sscCrosstab_project (SEXP *ctab*)

Project the (2,1) component of an sscCrosstab object into the (2,2) component (for illustration only)

Parameters:

ctab pointer to a sscCrosstab object

Returns:

a pointer to an sscMatrix giving the projection of the 2,1 component

2.30.1.5 SEXP sscCrosstab_project2 (SEXP *ctab*)

Project the first group of columns in an sscCrosstab object onto the remaining columns.

Parameters:

ctab pointer to a sscCrosstab object

Returns:

a pointer to an sscMatrix with the projection

2.31 sscCrosstab.h File Reference

```
#include "Mutils.h"
#include "ldl.h"
```

Functions

- SEXP sscCrosstab (SEXP flist, SEXP upper)
- void ssc_metis_order (int n, const int Tp[ ], const int Ti[ ], int Perm[ ], int iPerm[ ])
- SEXP sscCrosstab_groupedPerm (SEXP ctab)
- SEXP sscCrosstab_project2 (SEXP ctab)

2.31.1 Function Documentation

2.31.1.1 void ssc_metis_order (int *n*, const int *Tp*[ ], const int *Ti*[ ], int *Perm*[ ], int *iPerm*[ ])

2.31.1.2 SEXP sscCrosstab (SEXP *flist*, SEXP *upper*)

2.31.1.3 SEXP sscCrosstab_groupedPerm (SEXP *ctab*)

2.31.1.4 SEXP sscCrosstab_project2 (SEXP *ctab*)

Project the first group of columns in an sscCrosstab object onto the remaining columns.

Parameters:

ctab pointer to a sscCrosstab object

Returns:

a pointer to an sscMatrix with the projection

2.32 ssclme.c File Reference

```
#include "ssclme.h"
```

Defines

- #define slot_dup(dest, src, sym) SET_SLOT(dest, sym, duplicate(GET_SLOT(src, sym)))

Functions

- void ssclme_copy_ctab (int nf, const int nc[ ], SEXP ctab, SEXP ssc)
- void ssclme_calc_maxod (int n, int Parent[ ])
- SEXP ssclme_create (SEXP facs, SEXP ncv)
- void bVj_to_A (int ncj, int Gpj, int Gpjp, const double bVj[ ], const int Ap[ ], const int Ai[ ], double Ax[ ])
- SEXP ssclme_transfer_dimnames (SEXP x, SEXP facs, SEXP mmats)
- SEXP ssclme_update_mm (SEXP x, SEXP facs, SEXP mmats)
- SEXP ssclme_inflate_and_factor (SEXP x)
- SEXP ssclme_factor (SEXP x)
- int ldl_update_ind (int probe, int start, const int ind[ ])
- SEXP ldl_inverse (SEXP x)
- SEXP ssclme_invert (SEXP x)
- SEXP ssclme_initial (SEXP x)
- SEXP ssclme_fixef (SEXP x)
- SEXP ssclme_ranef (SEXP x)
- SEXP ssclme_sigma (SEXP x, SEXP REML)
- int coef_length (int nf, const int nc[ ])
- SEXP ssclme_coef (SEXP x)
- SEXP ssclme_coefUnc (SEXP x)
- SEXP ssclme_coefGetsUnc (SEXP x, SEXP coef)
- SEXP ssclme_coefGets (SEXP x, SEXP coef)
- SEXP ssclme_EMsteps (SEXP x, SEXP nsteps, SEXP REMLp, SEXP verb)
- SEXP ssclme_gradient (SEXP x, SEXP REMLp, SEXP Uncp)
- SEXP ssclme_fitted (SEXP x, SEXP facs, SEXP mmats, SEXP useRf)
- SEXP ssclme_variances (SEXP x)
- SEXP ssclme_collapse (SEXP x)
- SEXP ssclme_to_lme (SEXP call, SEXP facs, SEXP x, SEXP model, SEXP REML, SEXP rep, SEXP fitted, SEXP residuals)

2.32.1 Define Documentation

2.32.1.1 #define slot_dup(dest, src, sym) SET_SLOT(dest, sym, duplicate(GET_SLOT(src, sym)))

2.32.2 Function Documentation

2.32.2.1 void bVj_to_A (int *ncj*, int *Gpj*, int *Gpjp*, const double *bVj*[ ], const int *Ap*[ ], const int *Ai*[ ], double *Ax*[ ]) `[static]`

Copy information on Z'Z accumulated in the bVar array to Z'Z

Parameters:

ncj number of columns in this block

Gpj initial column for this group

Gpjp initial column for the next group

bVj pointer to the ncj x ncj x mj array to be filled

Ap column pointer array for Z'Z

Ai row indices for Z'Z

Ax elements of Z'Z

2.32.2.2 int coef_length (int *nf*, const int *nc*[ ]) `[static]`

Calculate the length of the parameter vector, which is called coef for historical reasons.

Parameters:

nf number of factors

nc number of columns in the model matrices for each factor

Returns:

total length of the coefficient vector

2.32.2.3 SEXP ldl_inverse (SEXP *x*) `[static]`

Update the diagonal blocks of the inverse of LDL' (=Z'Z+W). The lower Cholesky factors of the updated blocks are stored in the bVar slot.

Parameters:

x pointer to an ssclme object

Returns:

R_NilValue (x is updated in place)

2.32.2.4 int ldl_update_ind (int *probe*, int *start*, const int *ind*[ ]) `[static]`

Return the position of probe in the sorted index vector ind. It is known that the position is greater than or equal to start so a linear search from start is used.

Parameters:

probe value to be matched

start index at which to start

ind vector of indices

Returns:

index of the entry matching probe

2.32.2.5 void ssclme_calc_maxod (int *n*, int *Parent*[ ]) `[static]`

Calculate and store the maximum number of off-diagonal elements in the inverse of L, based on the elimination tree. The maximum is itself stored in the Parent array. (FIXME: come up with a better design.)

Parameters:

n number of columns in the matrix

Parent elimination tree for the matrix

2.32.2.6 SEXP ssclme_coef (SEXP *x*)

Extract the upper triangles of the Omega matrices. These aren't "coefficients" but the extractor is called coef for historical reasons. Within each group these values are in the order of the diagonal entries first then the strict upper triangle in row order.

Parameters:

x pointer to an ssclme object

Returns:

numeric vector of the values in the upper triangles of the Omega matrices

2.32.2.7 SEXP ssclme_coefGets (SEXP *x*, SEXP *coef*)

Assign the upper triangles of the Omega matrices. (Called coef for historical reasons.)

Parameters:

x pointer to an ssclme object

coef pointer to an numeric vector of appropriate length

Returns:

R_NilValue

2.32.2.8 SEXP ssclme_coefGetsUnc (SEXP *x*, SEXP *coef*)

Assign the Omega matrices from the unconstrained parameterization.

Parameters:

x pointer to an ssclme object

coef pointer to an numeric vector of appropriate length

Returns:

R_NilValue

2.32.2.9 SEXP ssclme_coefUnc (SEXP *x*)

Extract the unconstrained parameters that determine the Omega matrices. (Called coef for historical reasons.) The unconstrained parameters are derived from the LDL' decomposition of Omega_i. The first nc[i] entries in each group are the diagonals of log(D) followed by the strict lower triangle of L in column order.

Parameters:

x pointer to an ssclme object

Returns:

numeric vector of unconstrained parameters that determine the Omega matrices

2.32.2.10 SEXP ssclme_collapse (SEXP *x*)

Copy an ssclme object collapsing the fixed effects slots to the response only.

Parameters:

x pointer to an ssclme object

Returns:

a duplicate of x with the fixed effects slots collapsed to the response only

2.32.2.11 void ssclme_copy_ctab (int *nf*, const int *nc*[ ], SEXP *ctab*, SEXP *ssc*) `[static]`

Using the sscCrosstab object from the grouping factors, generate the slots in an ssclme object related to the symmetric sparse matrix representation of Z'Z. If the model matrices for the grouping factors have only one column each then the structure can be copied, otherwise it must be generated from the sscCrosstab and the number of columns per grouping factor.

Parameters:

nf number of factors

nc vector of length nf+2 with number of columns in model matrices

ctab pointer to the sscCrosstab object

ssc pointer to an ssclme object to be filled out

2.32.2.12 SEXP ssclme_create (SEXP *facs*, SEXP *ncv*)

Create an ssclme object from a list of grouping factors, sorted in order of non-increasing numbers of levels, and an integer vector of the number of columns in the model matrices. There is one more element in ncv than in facs. The last element is the number of columns in the model matrix for the fixed effects plus the response. (i.e. p+1)

Parameters:

facs pointer to a list of grouping factors

ncv pointer to an integer vector of number of columns per model matrix

Returns:

pointer to an ssclme object

2.32.2.13 SEXP ssclme_EMsteps (SEXP *x*, SEXP *nsteps*, SEXP *REMLp*, SEXP *verb*)

Perform a number of ECME steps for the REML or ML criterion.

Parameters:

x pointer to an ssclme object

nsteps pointer to an integer scalar giving the number of ECME steps to perform

REMLp pointer to a logical scalar indicating if REML is to be used

verb pointer to a logical scalar indicating verbose mode

Returns:

NULL

2.32.2.14 SEXP ssclme_factor (SEXP *x*)

If status[["factored"]] is FALSE, create and factor Z’Z+Omega, then create RZX and RXX, the deviance components, and the value of the deviance for both ML and REML.

Parameters:

x pointer to an ssclme object

Returns:

NULL

2.32.2.15 SEXP ssclme_fitted (SEXP *x*, SEXP *facs*, SEXP *mmats*, SEXP *useRf*)

Calculate and return the fitted values.

Parameters:

x pointer to an ssclme object

facs list of grouping factors

mmats list of model matrices

useRf pointer to a logical scalar indicating if the random effects should be used

Returns:

pointer to a numeric array of fitted values

2.32.2.16 SEXP ssclme_fixef (SEXP *x*)

Extract the conditional estimates of the fixed effects

Parameters:

x Pointer to an ssclme object

Returns:

a numeric vector containing the conditional estimates of the fixed effects

2.32.2.17 SEXP ssclme_gradient (SEXP *x*, SEXP *REMLp*, SEXP *Uncp*)

Return the gradient of the ML or REML deviance.

Parameters:

x pointer to an ssclme object

REMLp pointer to a logical scalar indicating if REML is to be used

Uncp pointer to a logical scalar indicating if the unconstrained parameterization is to be used

Returns:

pointer to a numeric vector of the gradient.

2.32.2.18 SEXP ssclme_inflate_and_factor (SEXP *x*)

Inflate Z'Z according to Omega and create the factorization LDL'

Parameters:

x pointer to an ssclme object

Returns:

NULL

2.32.2.19 SEXP ssclme_initial (SEXP *x*)

Create and insert initial values for Omega_i.

Parameters:

x pointer to an ssclme object

Returns:

NULL

2.32.2.20 SEXP ssclme_invert (SEXP *x*)

If necessary, factor Z'Z+Omega, ZtX, and XtX then, if necessary, form RZX, RXX, and bVar for the inverse of the Cholesky factor.

Parameters:

x pointer to an ssclme object

Returns:

NULL (x is updated in place)

2.32.2.21 SEXP ssclme_ranef (SEXP *x*)

Extract the conditional modes of the random effects.

Parameters:

x Pointer to an ssclme object

Returns:

a vector containing the conditional modes of the random effects

2.32.2.22 SEXP ssclme_sigma (SEXP *x*, SEXP *REML*)

Extract the ML or REML conditional estimate of sigma

Parameters:

x pointer to an ssclme object

REML logical scalar - TRUE if REML estimates are requested

Returns:

numeric scalar

2.32.2.23 SEXP ssclme_to_lme (SEXP *call*, SEXP *facs*, SEXP *x*, SEXP *model*, SEXP *REML*, SEXP *rep*, SEXP *fitted*, SEXP *residuals*)

Create an lme object from its components. This is not done by new("lme", ...) at the R level because of the possibility of causing the copying of very large objects.

Parameters:

call Pointer to the original call

facs pointer to the list of grouping factors

x pointer to the model matrices (may be of length zero)

model pointer to the model frame

REML pointer to a logical scalar indicating if REML is used

rep pointer to the converged ssclme object

fitted pointer to the fitted values

residuals pointer to the residuals

Returns:

an lme object

2.32.2.24 SEXP ssclme_transfer_dimnames (SEXP *x*, SEXP *facs*, SEXP *mmats*)

Copy the dimnames from the list of grouping factors and the model matrices for the grouping factors into the appropriate parts of the ssclme object.

Parameters:

x pointer to an ssclme object

facs pointer to a list of factors

mmats pointer to a list of model matrices

Returns:

NULL

2.32.2.25 SEXP ssclme_update_mm (SEXP *x*, SEXP *facs*, SEXP *mmats*)

Update the numerical entries x, ZtX, and XtX in an ssclme object according to a set of model matrices.

Parameters:

x pointer to an ssclme object

facs pointer to a list of grouping factors

mmats pointer to a list of model matrices

Returns:

NULL

2.32.2.26 SEXP ssclme_variances (SEXP *x*)

Return the unscaled variances

Parameters:

x pointer to an ssclme object

Returns:

a list similar to the Omega list with the unscaled variances

2.33 ssclme.h File Reference

```
#include "sscCrosstab.h"
#include <R_ext/Lapack.h>
#include <R_ext/Constants.h>
```

Functions

- SEXP ssclme_create (SEXP facs, SEXP ncv)
- SEXP ssclme_transfer_dimnames (SEXP x, SEXP facs, SEXP mmats)
- SEXP ssclme_update_mm (SEXP x, SEXP facs, SEXP mmats)
- SEXP ssclme_inflate_and_factor (SEXP x)
- SEXP ssclme_factor (SEXP x)
- SEXP ssclme_invert (SEXP x)
- SEXP ssclme_initial (SEXP x)
- SEXP ssclme_fixef (SEXP x)
- SEXP ssclme_ranef (SEXP x)
- SEXP ssclme_sigma (SEXP x, SEXP REML)

- SEXP ssclme_coef (SEXP x)
- SEXP ssclme_coefUnc (SEXP x)
- SEXP ssclme_coefGetsUnc (SEXP x, SEXP coef)
- SEXP ssclme_coefGets (SEXP x, SEXP coef)
- SEXP ssclme_EMsteps (SEXP x, SEXP nsteps, SEXP REMLp, SEXP verb)
- SEXP ssclme_fitted (SEXP x, SEXP facs, SEXP mmats, SEXP useRf)
- SEXP ssclme_variances (SEXP x)
- SEXP ssclme_gradient (SEXP x, SEXP REMLp, SEXP Uncp)
- SEXP ssclme_collapse (SEXP x)

2.33.1 Function Documentation

2.33.1.1 SEXP ssclme_coef (SEXP *x*)

Extract the upper triangles of the Omega matrices. These aren't "coefficients" but the extractor is called coef for historical reasons. Within each group these values are in the order of the diagonal entries first then the strict upper triangle in row order.

Parameters:

x pointer to an ssclme object

Returns:

numeric vector of the values in the upper triangles of the Omega matrices

2.33.1.2 SEXP ssclme_coefGets (SEXP *x*, SEXP *coef*)

Assign the upper triangles of the Omega matrices. (Called coef for historical reasons.)

Parameters:

x pointer to an ssclme object

coef pointer to an numeric vector of appropriate length

Returns:

R_NilValue

2.33.1.3 SEXP ssclme_coefGetsUnc (SEXP *x*, SEXP *coef*)

Assign the Omega matrices from the unconstrained parameterization.

Parameters:

x pointer to an ssclme object

coef pointer to an numeric vector of appropriate length

Returns:

R_NilValue

2.33.1.4 SEXP ssclme_coefUnc (SEXP *x*)

Extract the unconstrained parameters that determine the Omega matrices. (Called coef for historical reasons.) The unconstrained parameters are derived from the LDL' decomposition of Omega_i. The first nc[i] entries in each group are the diagonals of log(D) followed by the strict lower triangle of L in column order.

Parameters:

x pointer to an ssclme object

Returns:

numeric vector of unconstrained parameters that determine the Omega matrices

2.33.1.5 SEXP ssclme_collapse (SEXP *x*)

Copy an ssclme object collapsing the fixed effects slots to the response only.

Parameters:

x pointer to an ssclme object

Returns:

a duplicate of x with the fixed effects slots collapsed to the response only

2.33.1.6 SEXP ssclme_create (SEXP *facs*, SEXP *ncv*)

Create an ssclme object from a list of grouping factors, sorted in order of non-increasing numbers of levels, and an integer vector of the number of columns in the model matrices. There is one more element in ncv than in facs. The last element is the number of columns in the model matrix for the fixed effects plus the response. (i.e. p+1)

Parameters:

facs pointer to a list of grouping factors

ncv pointer to an integer vector of number of columns per model matrix

Returns:

pointer to an ssclme object

2.33.1.7 SEXP ssclme_EMsteps (SEXP *x*, SEXP *nsteps*, SEXP *REMLp*, SEXP *verb*)

Perform a number of ECME steps for the REML or ML criterion.

Parameters:

x pointer to an ssclme object

nsteps pointer to an integer scalar giving the number of ECME steps to perform

REMLp pointer to a logical scalar indicating if REML is to be used

verb pointer to a logical scalar indicating verbose mode

Returns:

NULL

2.33.1.8 SEXP ssclme_factor (SEXP *x*)

If status[["factored"]] is FALSE, create and factor Z'Z+Omega, then create RZX and RXX, the deviance components, and the value of the deviance for both ML and REML.

Parameters:

x pointer to an ssclme object

Returns:

NULL

2.33.1.9 SEXP ssclme_fitted (SEXP *x*, SEXP *facs*, SEXP *mmats*, SEXP *useRf*)

Calculate and return the fitted values.

Parameters:

x pointer to an ssclme object

facs list of grouping factors

mmats list of model matrices

useRf pointer to a logical scalar indicating if the random effects should be used

Returns:

pointer to a numeric array of fitted values

2.33.1.10 SEXP ssclme_fixef (SEXP *x*)

Extract the conditional estimates of the fixed effects

Parameters:

x Pointer to an ssclme object

Returns:

a numeric vector containing the conditional estimates of the fixed effects

2.33.1.11 SEXP ssclme_gradient (SEXP *x*, SEXP *REMLp*, SEXP *Uncp*)

Return the gradient of the ML or REML deviance.

Parameters:

x pointer to an ssclme object

REMLp pointer to a logical scalar indicating if REML is to be used

Uncp pointer to a logical scalar indicating if the unconstrained parameterization is to be used

Returns:

pointer to a numeric vector of the gradient.

2.33.1.12 SEXP ssclme_inflate_and_factor (SEXP *x*)

Inflate Z'Z according to Omega and create the factorization LDL'

Parameters:

x pointer to an ssclme object

Returns:

NULL

2.33.1.13 SEXP ssclme_initial (SEXP *x*)

Create and insert initial values for Omega_i.

Parameters:

x pointer to an ssclme object

Returns:

NULL

2.33.1.14 SEXP ssclme_invert (SEXP *x*)

If necessary, factor Z'Z+Omega, ZtX, and XtX then, if necessary, form RZX, RXX, and bVar for the inverse of the Cholesky factor.

Parameters:

x pointer to an ssclme object

Returns:

NULL (x is updated in place)

2.33.1.15 SEXP ssclme_ranef (SEXP *x*)

Extract the conditional modes of the random effects.

Parameters:

x Pointer to an ssclme object

Returns:

a vector containing the conditional modes of the random effects

2.33.1.16 SEXP ssclme_sigma (SEXP *x*, SEXP *REML*)

Extract the ML or REML conditional estimate of sigma

Parameters:

x pointer to an ssclme object

REML logical scalar - TRUE if REML estimates are requested

Returns:

numeric scalar

2.33.1.17 SEXP ssclme_transfer_dimnames (SEXP *x*, SEXP *facs*, SEXP *mmats*)

Copy the dimnames from the list of grouping factors and the model matrices for the grouping factors into the appropriate parts of the ssclme object.

Parameters:

x pointer to an ssclme object

facs pointer to a list of factors

mmats pointer to a list of model matrices

Returns:

NULL

2.33.1.18 SEXP ssclme_update_mm (SEXP *x*, SEXP *facs*, SEXP *mmats*)

Update the numerical entries x, ZtX, and XtX in an ssclme object according to a set of model matrices.

Parameters:

x pointer to an ssclme object

facs pointer to a list of grouping factors

mmats pointer to a list of model matrices

Returns:

NULL

2.33.1.19 SEXP ssclme_variances (SEXP *x*)

Return the unscaled variances

Parameters:

x pointer to an ssclme object

Returns:

a list similar to the Omega list with the unscaled variances

2.34 sscMatrix.c File Reference

```
#include "sscMatrix.h"
```

Functions

- SEXP sscMatrix_validate (SEXP obj)
- SEXP sscMatrix_chol (SEXP x, SEXP pivot)
- SEXP sscMatrix_matrix_solve (SEXP a, SEXP b)
- SEXP sscMatrix_inverse_factor (SEXP A)
- SEXP ssc_transpose (SEXP x)
- SEXP sscMatrix_to_triplet (SEXP x)
- SEXP sscMatrix_ldl_symbolic (SEXP x)
- SEXP sscMatrix_metis_perm (SEXP x)
- SEXP sscMatrix_metis_ldl_symbolic (SEXP x)

2.34.1 Function Documentation

2.34.1.1 SEXP ssc_transpose (SEXP *x*)

2.34.1.2 SEXP sscMatrix_chol (SEXP *x*, SEXP *pivot*)

2.34.1.3 SEXP sscMatrix_inverse_factor (SEXP *A*)

2.34.1.4 SEXP sscMatrix_ldl_symbolic (SEXP *x*)

2.34.1.5 SEXP sscMatrix_matrix_solve (SEXP *a*, SEXP *b*)

2.34.1.6 SEXP sscMatrix_metis_ldl_symbolic (SEXP *x*)

2.34.1.7 SEXP sscMatrix_metis_perm (SEXP *x*)

2.34.1.8 SEXP sscMatrix_to_triplet (SEXP *x*)

2.34.1.9 SEXP sscMatrix_validate (SEXP *obj*)

2.35 sscMatrix.h File Reference

```
#include "taucs_utils.h"
#include "ldl.h"
```

Functions

- SEXP sscMatrix_validate (SEXP x)
- SEXP sscMatrix_chol (SEXP x, SEXP pivot)
- SEXP sscMatrix_inverse_factor (SEXP A)
- SEXP sscMatrix_matrix_solve (SEXP a, SEXP b)
- SEXP ssc_transpose (SEXP x)
- SEXP sscMatrix_to_triplet (SEXP x)
- SEXP sscMatrix_ldl_symbolic (SEXP x)
- void ssc_metis_order (int n, const int Tp[ ], const int Ti[ ], int Perm[ ], int iPerm[ ])

2.35.1 Function Documentation

2.35.1.1 void ssc_metis_order (int *n*, const int *Tp*[ ], const int *Ti*[ ], int *Perm*[ ], int *iPerm*[ ])

2.35.1.2 SEXP ssc_transpose (SEXP *x*)

2.35.1.3 SEXP sscMatrix_chol (SEXP *x*, SEXP *pivot*)

2.35.1.4 SEXP sscMatrix_inverse_factor (SEXP *A*)

2.35.1.5 SEXP sscMatrix_ldl_symbolic (SEXP *x*)

2.35.1.6 SEXP sscMatrix_matrix_solve (SEXP *a*, SEXP *b*)

2.35.1.7 SEXP sscMatrix_to_triplet (SEXP *x*)

2.35.1.8 SEXP sscMatrix_validate (SEXP *x*)

2.36 syMatrix.c File Reference

```
#include "syMatrix.h"
```

Functions

- SEXP syMatrix_validate (SEXP obj)
- void make_symmetric (double ∗to, SEXP from, int n)
- SEXP syMatrix_as_geMatrix (SEXP from)
- SEXP syMatrix_as_matrix (SEXP from)
- double get_norm_sy (SEXP obj, char ∗typstr)
- SEXP syMatrix_norm (SEXP obj, SEXP type)
- SEXP syMatrix_geMatrix_mm (SEXP a, SEXP b)
- SEXP syMatrix_geMatrix_mm_R (SEXP a, SEXP b)

2.36.1 Function Documentation

2.36.1.1 double get_norm_sy (SEXP *obj*, char ∗ *typstr*)

2.36.1.2 void make_symmetric (double ∗ *to*, SEXP *from*, int *n*) `[static]`

2.36.1.3 SEXP syMatrix_as_geMatrix (SEXP *from*)

2.36.1.4 SEXP syMatrix_as_matrix (SEXP *from*)

2.36.1.5 SEXP syMatrix_geMatrix_mm (SEXP *a*, SEXP *b*)

2.36.1.6 SEXP syMatrix_geMatrix_mm_R (SEXP *a*, SEXP *b*)

2.36.1.7 SEXP syMatrix_norm (SEXP *obj*, SEXP *type*)

2.36.1.8 SEXP syMatrix_validate (SEXP *obj*)

2.37 syMatrix.h File Reference

```
#include "geMatrix.h"
#include "R_ext/Lapack.h"
```

Functions

- SEXP syMatrix_validate (SEXP obj)
- SEXP syMatrix_norm (SEXP obj, SEXP type)
- SEXP syMatrix_rcond (SEXP obj, SEXP type)
- SEXP syMatrix_solve (SEXP a)
- SEXP syMatrix_matrix_solve (SEXP a, SEXP b)
- SEXP syMatrix_as_geMatrix (SEXP from)
- SEXP syMatrix_as_matrix (SEXP from)
- double get_norm_sy (SEXP obj, char ∗typstr)
- SEXP syMatrix_geMatrix_mm (SEXP a, SEXP b)
- SEXP syMatrix_geMatrix_mm_R (SEXP a, SEXP b)

2.37.1 Function Documentation

2.37.1.1 double get_norm_sy (SEXP *obj*, char ∗ *typstr*)

2.37.1.2 SEXP syMatrix_as_geMatrix (SEXP *from*)

2.37.1.3 SEXP syMatrix_as_matrix (SEXP *from*)

2.37.1.4 SEXP syMatrix_geMatrix_mm (SEXP *a*, SEXP *b*)

2.37.1.5 SEXP syMatrix_geMatrix_mm_R (SEXP *a*, SEXP *b*)

2.37.1.6 SEXP syMatrix_matrix_solve (SEXP *a*, SEXP *b*)

2.37.1.7 SEXP syMatrix_norm (SEXP *obj*, SEXP *type*)

2.37.1.8 SEXP syMatrix_rcond (SEXP *obj*, SEXP *type*)

2.37.1.9 SEXP syMatrix_solve (SEXP *a*)

2.37.1.10 SEXP syMatrix_validate (SEXP *obj*)

2.38 taucs_utils.c File Reference

```
#include "taucs_utils.h"
```

Functions

- taucs_ccs_matrix ∗ csc_taucs_ptr (SEXP A, int flags)
- SEXP mat_from_taucs (taucs_ccs_matrix ∗tm)
- taucs_ccs_matrix ∗ copy_csc_to_taucs (SEXP A, int typ)
- double taucs_wtime ()
- double taucs_ctime ()
- void ∗ taucs_malloc_stub (size_t size)
- void ∗ taucs_calloc_stub (size_t nmemb, size_t size)
- void ∗ taucs_realloc_stub (void ∗ptr, size_t size)
- void taucs_free_stub (void ∗ptr)
- double taucs_allocation_amount ()
- int taucs_allocation_count ()
- int taucs_allocation_attempts ()
- void taucs_allocation_assert_clean ()
- void taucs_allocation_mark_clean ()
- void taucs_allocation_induce_failure (int i)
- int taucs_printf (char ∗fmt,...)
- double taucs_get_nan ()

Variables

- double taucs_dzero_const = 0.0
- double taucs_done_const = 1.0
- double taucs_dminusone_const = -1.0

2.38.1 Function Documentation

2.38.1.1 taucs_ccs_matrix* copy_csc_to_taucs (SEXP *A*, int *typ*)

2.38.1.2 taucs_ccs_matrix* csc_taucs_ptr (SEXP *A*, int *flags*)

Create a pointer to a taucs_ccs_matrix from an R object that inherits from class cscMatrix according to the flags.

Parameters:

A Pointer to an object that inherits from cscMatrix

flags taucs flags describing the matrix

Returns:

A taucs_ccs_matrix pointer to the existing storage (no copying).

2.38.1.3 SEXP mat_from_taucs (taucs_ccs_matrix * *tm*)

Copy a taucs_ccs_matrix to an R object of the appropriate class and free the storage used by the taucs_ccs_matrix.

Parameters:

tm A pointer to a taucs_ccs_matrix

Returns:

An R object of class "cscMatrix" or "sscMatrix" or "tscMatrix"

2.38.1.4 double taucs_allocation_amount ()

2.38.1.5 void taucs_allocation_assert_clean ()

2.38.1.6 int taucs_allocation_attempts ()

2.38.1.7 int taucs_allocation_count ()

2.38.1.8 void taucs_allocation_induce_failure (int *i*)

2.38.1.9 void taucs_allocation_mark_clean ()

2.38.1.10 void∗ taucs_calloc_stub (size_t *nmemb*, size_t *size*)

2.38.1.11 double taucs_ctime ()

2.38.1.12 void taucs_free_stub (void ∗ *ptr*)

2.38.1.13 double taucs_get_nan ()

2.38.1.14 void∗ taucs_malloc_stub (size_t *size*)

2.38.1.15 int taucs_printf (char ∗ *fmt, ...*)

2.38.1.16 void∗ taucs_realloc_stub (void ∗ *ptr*, size_t *size*)

2.38.1.17 double taucs_wtime ()

2.38.2 Variable Documentation

2.38.2.1 double taucs_dminusone_const = -1.0

2.38.2.2 double taucs_done_const = 1.0

2.38.2.3 double taucs_dzero_const = 0.0

2.39 taucs_utils.h File Reference

```
#include "Mutils.h"
#include "taucs/taucs.h"
```

Functions

- taucs_ccs_matrix ∗ csc_taucs_ptr (SEXP A, int flags)
- SEXP mat_from_taucs (taucs_ccs_matrix ∗tm)

2.39.1 Function Documentation

2.39.1.1 taucs_ccs_matrix∗ csc_taucs_ptr (SEXP *A*, int *flags*)

Create a pointer to a taucs_ccs_matrix from an R object that inherits from class cscMatrix according to the flags.

Parameters:

A Pointer to an object that inherits from cscMatrix

flags taucs flags describing the matrix

Returns:

A taucs_ccs_matrix pointer to the existing storage (no copying).

2.39.1.2 SEXP mat_from_taucs (taucs_ccs_matrix ∗ *tm*)

Copy a taucs_ccs_matrix to an R object of the appropriate class and free the storage used by the taucs_ccs_matrix.

Parameters:

tm A pointer to a taucs_ccs_matrix

Returns:

An R object of class "cscMatrix" or "sscMatrix" or "tscMatrix"

2.40 triplet.c File Reference

```
#include "triplet.h"
```

Functions

- SEXP triplet_validate (SEXP x)
- SEXP triplet_to_geMatrix (SEXP x)

2.40.1 Function Documentation

2.40.1.1 SEXP triplet_to_geMatrix (SEXP *x*)

2.40.1.2 SEXP triplet_validate (SEXP *x*)

2.41 triplet.h File Reference

```
#include "Mutils.h"
```

Functions

- SEXP triplet_validate (SEXP x)
- SEXP triplet_to_geMatrix (SEXP x)

2.41.1 Function Documentation

2.41.1.1 SEXP triplet_to_geMatrix (SEXP *x*)

2.41.1.2 SEXP triplet_validate (SEXP *x*)

2.42 triplet_to_col.c File Reference

```
#include <R_ext/RS.h>
```

Functions

- void triplet_to_col (int n_row, int n_col, int nz, const int Ti[ ], const int Tj[ ], const double Tx[ ], int Ap[ ], int Ai[ ], double Ax[ ])

2.42.1 Function Documentation

2.42.1.1 void triplet_to_col (int *n_row*, int *n_col*, int *nz*, const int *Ti*[ ], const int *Tj*[ ], const double *Tx*[ ], int *Ap*[ ], int *Ai*[ ], double *Ax*[ ])

2.43 triplet_to_col.h File Reference

Functions

- void triplet_to_col (int n_row, int n_col, int nz, const int Ti[ ], const int Tj[ ], const double Tx[ ], int Ap[ ], int Ai[ ], double Ax[ ])

2.43.1 Function Documentation

2.43.1.1 void triplet_to_col (int *n_row*, int *n_col*, int *nz*, const int *Ti*[ ], const int *Tj*[ ], const double *Tx*[ ], int *Ap*[ ], int *Ai*[ ], double *Ax*[ ])

2.44 trMatrix.c File Reference

```
#include "trMatrix.h"
```

Functions

- SEXP trMatrix_validate (SEXP obj)
- double get_norm (SEXP obj, char ∗typstr)
- SEXP trMatrix_norm (SEXP obj, SEXP type)
- double set_rcond (SEXP obj, char ∗typstr)
- SEXP trMatrix_rcond (SEXP obj, SEXP type)
- SEXP trMatrix_solve (SEXP a)
- void make_array_triangular (double ∗to, SEXP from)
- SEXP trMatrix_as_geMatrix (SEXP from)
- SEXP trMatrix_as_matrix (SEXP from)
- SEXP trMatrix_getDiag (SEXP x)
- SEXP trMatrix_geMatrix_mm (SEXP a, SEXP b)
- SEXP trMatrix_geMatrix_mm_R (SEXP a, SEXP b)

2.44.1 Function Documentation

2.44.1.1 double get_norm (SEXP *obj*, char ∗ *typstr*) `[static]`

2.44.1.2 void make_array_triangular (double ∗ *to*, SEXP *from*)

2.44.1.3 double set_rcond (SEXP *obj*, char ∗ *typstr*) `[static]`

2.44.1.4 SEXP trMatrix_as_geMatrix (SEXP *from*)

2.44.1.5 SEXP trMatrix_as_matrix (SEXP *from*)

2.44.1.6 SEXP trMatrix_geMatrix_mm (SEXP *a*, SEXP *b*)

2.44.1.7 SEXP trMatrix_geMatrix_mm_R (SEXP *a*, SEXP *b*)

2.44.1.8 SEXP trMatrix_getDiag (SEXP *x*)

2.44.1.9 SEXP trMatrix_norm (SEXP *obj*, SEXP *type*)

2.44.1.10 SEXP trMatrix_rcond (SEXP *obj*, SEXP *type*)

2.44.1.11 SEXP trMatrix_solve (SEXP *a*)

2.44.1.12 SEXP trMatrix_validate (SEXP *obj*)

2.45 trMatrix.h File Reference

```
#include <R_ext/Lapack.h>
#include "Mutils.h"
```

Functions

- SEXP trMatrix_validate (SEXP obj)
- SEXP trMatrix_norm (SEXP obj, SEXP type)
- SEXP trMatrix_rcond (SEXP obj, SEXP type)
- SEXP trMatrix_solve (SEXP a)
- SEXP trMatrix_matrix_solve (SEXP a, SEXP b)
- SEXP trMatrix_as_geMatrix (SEXP from)
- SEXP trMatrix_as_matrix (SEXP from)
- SEXP trMatrix_getDiag (SEXP x)
- void make_array_triangular (double ∗x, SEXP from)
- SEXP trMatrix_geMatrix_mm (SEXP a, SEXP b)
- SEXP trMatrix_geMatrix_mm_R (SEXP a, SEXP b)

2.45.1 Function Documentation

2.45.1.1 void make_array_triangular (double ∗ *x*, SEXP *from*)

2.45.1.2 SEXP trMatrix_as_geMatrix (SEXP *from*)

2.45.1.3 SEXP trMatrix_as_matrix (SEXP *from*)

2.45.1.4 SEXP trMatrix_geMatrix_mm (SEXP *a*, SEXP *b*)

2.45.1.5 SEXP trMatrix_geMatrix_mm_R (SEXP *a*, SEXP *b*)

2.45.1.6 SEXP trMatrix_getDiag (SEXP *x*)

2.45.1.7 SEXP trMatrix_matrix_solve (SEXP *a*, SEXP *b*)

2.45.1.8 SEXP trMatrix_norm (SEXP *obj*, SEXP *type*)

2.45.1.9 SEXP trMatrix_rcond (SEXP *obj*, SEXP *type*)

2.45.1.10 SEXP trMatrix_solve (SEXP *a*)

2.45.1.11 SEXP trMatrix_validate (SEXP *obj*)

2.46 tscMatrix.c File Reference

```
#include "tscMatrix.h"
```

Functions

- SEXP tsc_validate (SEXP x)
- SEXP tsc_transpose (SEXP x)
- SEXP tsc_to_triplet (SEXP x)

2.46.1 Function Documentation

2.46.1.1 SEXP tsc_to_triplet (SEXP *x*)

2.46.1.2 SEXP tsc_transpose (SEXP *x*)

2.46.1.3 SEXP tsc_validate (SEXP *x*)

2.47 tscMatrix.h File Reference

```
#include "Mutils.h"
#include "cscMatrix.h"
```

Functions

- SEXP tsc_validate (SEXP x)
- SEXP tsc_transpose (SEXP x)
- SEXP tsc_to_triplet (SEXP x)

2.47.1 Function Documentation

2.47.1.1 SEXP tsc_to_triplet (SEXP *x*)

2.47.1.2 SEXP tsc_transpose (SEXP *x*)

2.47.1.3 SEXP tsc_validate (SEXP *x*)

2.48 utils.c File Reference

```
#include "R.h"
#include "taucs/taucs.h"
```

Functions

- double taucs_wtime ()
- double taucs_ctime ()
- void ∗ taucs_malloc_stub (size_t size)
- void ∗ taucs_calloc_stub (size_t nmemb, size_t size)
- void ∗ taucs_realloc_stub (void ∗ptr, size_t size)
- void taucs_free_stub (void ∗ptr)
- double taucs_allocation_amount ()
- int taucs_allocation_count ()
- int taucs_allocation_attempts ()
- void taucs_allocation_assert_clean ()
- void taucs_allocation_mark_clean ()
- void taucs_allocation_induce_failure (int i)
- int taucs_printf (char ∗fmt,...)
- double taucs_get_nan ()

Variables

- double taucs_dzero_const = 0.0
- double taucs_done_const = 1.0
- double taucs_dminusone_const = -1.0

2.48.1 Function Documentation

2.48.1.1 double taucs_allocation_amount ()

2.48.1.2 void taucs_allocation_assert_clean ()

2.48.1.3 int taucs_allocation_attempts ()

2.48.1.4 int taucs_allocation_count ()

2.48.1.5 void taucs_allocation_induce_failure (int *i*)

2.48.1.6 void taucs_allocation_mark_clean ()

2.48.1.7 void∗ taucs_calloc_stub (size_t *nmemb*, size_t *size*)

2.48.1.8 double taucs_ctime ()

2.48.1.9 void taucs_free_stub (void ∗ *ptr*)

2.48.1.10 double taucs_get_nan ()

2.48.1.11 void∗ taucs_malloc_stub (size_t *size*)

2.48.1.12 int taucs_printf (char ∗ *fmt*, ...)

2.48.1.13 void∗ taucs_realloc_stub (void ∗ *ptr*, size_t *size*)

2.48.1.14 double taucs_wtime ()

2.48.2 Variable Documentation

2.48.2.1 double taucs_dminusone_const = -1.0

2.48.2.2 double taucs_done_const = 1.0

2.48.2.3 double taucs_dzero_const = 0.0

Index