matrix_decomposition Documentation

Release 0.1

Joscha Reimer

Jul 19, 2018

Contents:

CHAPTER 1

Functions

Several functions are included in this package. The most important are summarized here.

1.1 decompose a matrix

matrix.**decompose**(*A*, *permutation_method=None*, *check_finite=True*, *return_type=None*)

Computes a decomposition of a matrix.

Parameters

- **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – Matrix to be decomposed. It is assumed, that A is Hermitian. The matrix must be a squared matrix.
- **permutation_method** (*str*) – The symmetric permutation method that is applied to the matrix before it is decomposed. It has to be a value in *matrix.PERMUTATION_METHODS*. If *A* is sparse, it can also be a value in *matrix.SPARSE_PERMUTATION_METHODS*. optional, default: no permutation
- **check_finite** (*bool*) – Whether to check that the input matrix contains only finite numbers. Disabling may result in problems (crashes, non-termination) if the inputs do contain infinities or NaNs. (disabling may improve performance) optional, default: True
- **return_type** (*str*) – The type of the decomposition that should be calculated. It has to be a value in *matrix.DECOMPOSITION_TYPES*. If return_type is None the type of the returned decomposition is chosen by the function itself. optional, default: the type of the decomposition is chosen by the function itself

Returns A decompostion of *A* of type *return_type*.

Return type *matrix.decompositions.DecompositionBase*

Raises *matrix.errors.MatrixNoDecompositionPossibleError* – If the decomposition of *A* is not possible.

matrix.**PERMUTATION_METHODS = (None, '', 'none', 'natural', 'decreasing_diagonal_values', '**

Supported permutation methods for dense and sparse matrices.

matrix.**SPARSE_PERMUTATION_METHODS = ()**
Supported permutation methods only for sparse matrices.

matrix.**DECOMPOSITION_TYPES = ('LDL', 'LDL_compressed', 'LL')**
Supported types of decompositions.

1.2 examine positive definiteness

matrix.**is_positive_semi_definite**(*A*)
Checks if the passed matrix is positive semi-definite.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be checked. It is assumed, that A is Hermitian. The matrix must be a squared matrix.

Returns Whether *A* is positive semi-definite.

Return type bool

matrix.**is_positive_definite**(*A*)
Checks if the passed matrix is positive definite.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be checked. It is assumed, that A is Hermitian. The matrix must be a squared matrix.

Returns Whether *A* is positive definite.

Return type bool

1.3 approximate by a decomposition

matrix.**approximate**(*A*, *t=None*, *min_diag_value=None*, *max_diag_value=None*, *min_abs_value=None*, *permutation_method=None*, *check_finite=True*, *return_type=None*, *callback=None*)
Computes an approximative decomposition of a matrix.

If *A* is decomposable in a decomposition of type *return_type*, this decomposition is returned. Otherwise a decomposition of type *return_type* is retuned which represents an approximation of *A*.

Parameters

- **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be approximated by a decomposition. It is assumed, that A is Hermitian. The matrix must be a squared matrix.
- **t** (*numpy.ndarray*) – The targed vector used for the approximation. For each i in range(M) *min_diag_value* <= *t[i]* <= *max_diag_value* must hold. *t* and *A* must have the same length. optional, default : The diagonal of *A* is used as *t*.
- **min_diag_value** (*float*) – Each component of the diagonal of the matrix *D* in an returned *LDL* decomposition is forced to be greater or equal to *min_diag_value*. optional, default : No minimal value is forced.
- **max_diag_value** (*float*) – Each component of the diagonal of the matrix *D* in an returned *LDL* decomposition is forced to be lower or equal to *max_diag_value*. optional, default : No maximal value is forced.
- **min_abs_value** (*float*) – Absolute values below *min_abs_value* are considered as zero. optional, default : The resolution of the underlying data type is used.

- **permutation_method** (*str*) – The symmetric permutation method that is applied to the matrix before it is decomposed. It has to be a value in *matrix.PERMUTATION_METHODS*. If *A* is sparse, it can also be a value in *matrix.SPARSE_PERMUTATION_METHODS*. optional, default: No permutation is done.
- **check_finite** (*bool*) – Whether to check that the input matrix contains only finite numbers. Disabling may result in problems (crashes, non-termination) if the inputs do contain infinities or NaNs. (disabling may improve performance) optional, default: True
- **return_type** (*str*) – The type of the decomposition that should be calculated. It has to be a value in *matrix.DECOMPOSITION_TYPES*. optional, default : The type of the decomposition is chosen by the function itself.
- **callback** (*callable*) – In each iteration *callback(i, r)* is called where *i* is the index of the row and column where components of *A* are reduced by the factor *r*. optional, default : No callback function is called.

Returns An approximative decompostion of *A* of type *return_type*.

Return type *matrix.decompositions.DecompositionBase*

CHAPTER 2

Matrix decompositions

Several matrix decompositions are supported. They are available in *matrix.decompositions*:

2.1 LL decomposition

class matrix.decompositions.**LL_Decomposition**(*L*, *p=None*)

Bases: *matrix.decompositions.DecompositionBase*

A matrix decomposition where LL^H is the decomposed (permuted) matrix.

L is a lower triangle matrix with ones on the diagonal. This decomposition is also called Cholesky decomposition.

Parameters

- **L** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix *L* of the decomposition.
- **p** (*numpy.ndarray*) – The permutation vector used for the decomposition. This decomposition is of A[p[:, np.newaxis], p[np.newaxis, :]] where A is a matrix. optional, default: no permutation

L

numpy.matrix or scipy.sparse.spmatrix – The matrix *L* of the decomposition.

P

scipy.sparse.dok_matrix – The permutation matrix. P @ A @ P.H is the matrix A permuted by the permutation of the decomposition

composed_matrix

numpy.matrix or scipy.sparse.spmatrix – The composed matrix represented by this decomposition.

copy()

Copy this decomposition.

Returns A copy of this decomposition.

Return type *matrix.decompositions.DecompositionBase*

decomposition_type
str – The type of this decompositon.

is_permuted
bool – Whether this is a decompositon with permutation.

is_positive_definite()
bool: Whether the matrix represented by this decomposition is positive definite.

is_positive_semi_definite()
bool: Whether the matrix represented by this decomposition is positive semi-definite.

is_sparse
bool – Whether this is a sparse decompositon.

is_type(*decomposition_type*)
Whether this is a decomposition of the passed type.

Parameters **decomposition_type** (*str*) – The decomposition type according to which is checked.

Returns Whether this is a decomposition of the passed type.

Return type bool

n
int – The dimension of the squared decomposed matrix.

p
numpy.ndarray – The permutation vector. A[p[:, np.newaxis], p[np.newaxis, :]] is the matrix A permuted by the permutation of the decomposition

p_inverse
numpy.ndarray – The permutation vector that undos the permutation.

permute_matrix(*A*)
Permute a matrix by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be permuted.

Returns The matrix *A* permuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

to(*decomposition_type*, *copy=False*)
Convert decomposition to passed type.

Parameters

- **decomposition_type** (*str*) – The decomposition type to which this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not *decomposition_type*, a decompostion of type *decomposition_type* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

to_LDL_Decomposition()

to_any(**decomposition_types*, *copy=False*)
Convert decomposition to any of the passed types.

Parameters

- ***decomposition_types** (*str*) – The decomposition types to any of them this this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not in *decomposition_types*, a decompostion of type *decomposition_type[0]* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

unpermute_matrix(*A*)
Unpermute a matrix permuted by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be unpermuted.

Returns The matrix *A* unpermuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

2.2 LDL decomposition

class matrix.decompositions.**LDL_Decomposition**(*L*, *d*, *p=None*)
Bases: *matrix.decompositions.DecompositionBase*

A matrix decomposition where LDL^H is the decomposed (permuted) matrix.

L is a lower triangle matrix with ones on the diagonal. *D* is a diagonal matrix. Only the diagonal values of *D* are stored.

Parameters

- **L** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix *L* of the decomposition.
- **d** (*numpy.ndarray*) – The vector of the diagonal components of *D* of the decompositon.
- **p** (*numpy.ndarray*) – The permutation vector used for the decomposition. This decomposition is of A[p[:, np.newaxis], p[np.newaxis, :]] where A is a matrix. optional, default: no permutation

D
scipy.sparse.dia_matrix – The permutation matrix.

L
numpy.matrix or scipy.sparse.spmatrix – The matrix *L* of the decomposition.

LD
numpy.matrix or scipy.sparse.spmatrix – A matrix whose diagonal values are the diagonal values of *D* and whose off-diagonal values are those of *L*.

P
scipy.sparse.dok_matrix – The permutation matrix. P @ A @ P.H is the matrix A permuted by the permutation of the decomposition

composed_matrix
numpy.matrix or scipy.sparse.spmatrix – The composed matrix represented by this decomposition.

copy()
Copy this decomposition.

Returns A copy of this decomposition.

Return type *matrix.decompositions.DecompositionBase*

d
numpy.ndarray – The diagonal vector of the matrix *D* of the decomposition.

decomposition_type
str – The type of this decompositon.

is_permuted
bool – Whether this is a decompositon with permutation.

is_positive_definite()
bool: Whether the matrix represented by this decomposition is positive definite.

is_positive_semi_definite()
bool: Whether the matrix represented by this decomposition is positive semi-definite.

is_sparse
bool – Whether this is a sparse decompositon.

is_type(*decomposition_type*)
Whether this is a decomposition of the passed type.

Parameters **decomposition_type** (*str*) – The decomposition type according to which is checked.

Returns Whether this is a decomposition of the passed type.

Return type bool

n
int – The dimension of the squared decomposed matrix.

P
numpy.ndarray – The permutation vector. A[p[:, np.newaxis], p[np.newaxis, :]] is the matrix A permuted by the permutation of the decomposition

p_inverse
numpy.ndarray – The permutation vector that undos the permutation.

permute_matrix(*A*)
Permute a matrix by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be permuted.

Returns The matrix *A* permuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

to(*decomposition_type*, *copy=False*)
Convert decomposition to passed type.

Parameters

- **decomposition_type** (*str*) – The decomposition type to which this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not *decomposition_type*, a decompostion of type *decomposition_type* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

to_LDL_DecompositionCompressed()

to_LL_Decomposition()

to_any(**decomposition_types*, *copy=False*)

Convert decomposition to any of the passed types.

Parameters

- ***decomposition_types** (*str*) – The decomposition types to any of them this this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not in *decomposition_types*, a decompostion of type *decomposition_type[0]* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

unpermute_matrix(*A*)

Unpermute a matrix permuted by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be unpermuted.

Returns The matrix *A* unpermuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

2.3 LDL decomposition compressed

class matrix.decompositions.**LDL_DecompositionCompressed**(*LD*, *p=None*)

Bases: *matrix.decompositions.DecompositionBase*

A matrix decomposition where LDL^H is the decomposed (permuted) matrix.

L is a lower triangle matrix with ones on the diagonal. D is a diagonal matrix. L and D are stored in one matrix whose diagonal values are the diagonal values of D and whose off-diagonal values are those of L.

Parameters

- **LD** (*numpy.ndarray or scipy.sparse.spmatrix*) – A matrix whose diagonal values are the diagonal values of D and whose off-diagonal values are those of L.

- **p** (*numpy.ndarray*) – The permutation vector used for the decomposition. This decomposition is of A[p[:, np.newaxis], p[np.newaxis, :]] where A is a matrix. optional, default: no permutation

D
scipy.sparse.dia_matrix – The permutation matrix.

L
numpy.matrix or scipy.sparse.spmatrix – The matrix *L* of the decomposition.

LD
numpy.matrix or scipy.sparse.spmatrix – A matrix whose diagonal values are the diagonal values of *D* and whose off-diagonal values are those of *L*.

P
scipy.sparse.dok_matrix – The permutation matrix. P @ A @ P.H is the matrix A permuted by the permutation of the decomposition

composed_matrix
numpy.matrix or scipy.sparse.spmatrix – The composed matrix represented by this decomposition.

copy()
Copy this decomposition.

Returns A copy of this decomposition.

Return type *matrix.decompositions.DecompositionBase*

d
numpy.ndarray – The diagonal vector of the matrix *D* of the decomposition.

decomposition_type
str – The type of this decompositon.

is_permuted
bool – Whether this is a decompositon with permutation.

is_positive_definite()
bool: Whether the matrix represented by this decomposition is positive definite.

is_positive_semi_definite()
bool: Whether the matrix represented by this decomposition is positive semi-definite.

is_sparse
bool – Whether this is a sparse decompositon.

is_type(*decomposition_type*)
Whether this is a decomposition of the passed type.

Parameters **decomposition_type** (*str*) – The decomposition type according to which is checked.

Returns Whether this is a decomposition of the passed type.

Return type bool

n
int – The dimension of the squared decomposed matrix.

P
numpy.ndarray – The permutation vector. A[p[:, np.newaxis], p[np.newaxis, :]] is the matrix A permuted by the permutation of the decomposition

p_inverse
numpy.ndarray – The permutation vector that undos the permutation.

permute_matrix(*A*)
Permute a matrix by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be permuted.

Returns The matrix *A* permuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

to(*decomposition_type*, *copy=False*)
Convert decomposition to passed type.

Parameters

- **decomposition_type** (*str*) – The decomposition type to which this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not *decomposition_type*, a decompostion of type *decomposition_type* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

to_LDL_Decomposition()

to_any(**decomposition_types*, *copy=False*)
Convert decomposition to any of the passed types.

Parameters

- ***decomposition_types** (*str*) – The decomposition types to any of them this this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not in *decomposition_types*, a decompostion of type *decomposition_type[0]* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

unpermute_matrix(*A*)
Unpermute a matrix permuted by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be unpermuted.

Returns The matrix *A* unpermuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

2.4 base decomposition

class matrix.decompositions.**DecompositionBase**(*p=None*, *decomposition_type=None*)
Bases: object

A matrix decomposition.

This class is a base class for matrix decompositions.

Parameters

- **p** (*numpy.ndarray*) – The permutation vector used for the decomposition. This decomposition is of A[p[:, np.newaxis], p[np.newaxis, :]] where A is a matrix. optional, default: no permutation
- **decomposition_type** (*str*) – Type of this decomposition. optional, default: type not specified

P
scipy.sparse.dok_matrix – The permutation matrix. P @ A @ P.H is the matrix A permuted by the permutation of the decomposition

composed_matrix
numpy.matrix or scipy.sparse.spmatrix – The composed matrix represented by this decomposition.

copy()
Copy this decomposition.

Returns A copy of this decomposition.

Return type *matrix.decompositions.DecompositionBase*

decomposition_type
str – The type of this decompositon.

is_permuted
bool – Whether this is a decompositon with permutation.

is_positive_definite
bool – Whether the matrix represented by this decomposition is positive definite.

is_positive_semi_definite
bool – Whether the matrix represented by this decomposition is positive semi-definite.

is_sparse
bool – Whether this is a sparse decompositon.

is_type(*decomposition_type*)
Whether this is a decomposition of the passed type.

Parameters **decomposition_type** (*str*) – The decomposition type according to which is checked.

Returns Whether this is a decomposition of the passed type.

Return type bool

n
int – The dimension of the squared decomposed matrix.

P
numpy.ndarray – The permutation vector. A[p[:, np.newaxis], p[np.newaxis, :]] is the matrix A permuted by the permutation of the decomposition

p_inverse
numpy.ndarray – The permutation vector that undos the permutation.

permute_matrix(*A*)
Permute a matrix by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be permuted.

Returns The matrix *A* permuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

to(*decomposition_type*, *copy=False*)
Convert decomposition to passed type.

Parameters

- **decomposition_type** (*str*) – The decomposition type to which this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not *decomposition_type*, a decompostion of type *decomposition_type* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

to_any(**decomposition_types*, *copy=False*)
Convert decomposition to any of the passed types.

Parameters

- ***decomposition_types** (*str*) – The decomposition types to any of them this this decomposition is converted.
- **copy** (*bool*) – Whether the data of this decomposition should always be copied or only if needed.

Returns If the type of this decomposition is not in *decomposition_types*, a decompostion of type *decomposition_type[0]* is returned which represents the same decomposed matrix as this decomposition. Otherwise this decomposition or a copy of it is returned, depending on *copy*.

Return type *matrix.decompositions.DecompositionBase*

unpermute_matrix(*A*)
Unpermute a matrix permuted by the permutation of the decomposition.

Parameters **A** (*numpy.ndarray or scipy.sparse.spmatrix*) – The matrix that should be unpermuted.

Returns The matrix *A* unpermuted by the permutation of the decomposition.

Return type numpy.ndarray or scipy.sparse.spmatrix

CHAPTER 3

Errors

This is an overview about the exceptions that could arise in this package. They are available in *matrix.errors*:

3.1 MatrixNoDecompositionPossibleError

class matrix.errors.**MatrixNoDecompositionPossibleError**(*matrix=None*, *decomposition_decription=None*, *message=None*)

Bases: *matrix.errors.MatrixError*

The matrix decomposition is not possible for this matrix.

3.2 MatrixNoLDLDecompositionPossibleError

class matrix.errors.**MatrixNoLDLDecompositionPossibleError**(*matrix=None*, *problematic_leading_principal_submatrix_index=None*, *subdecomposition=None*)

Bases: *matrix.errors.MatrixNoDecompositionPossibleWithProblematicSubdecompositionError*

A LDL decomposition is not possible for this matrix.

3.3 MatrixNoLLDecompositionPossibleError

class matrix.errors.**MatrixNoLLDecompositionPossibleError**(*matrix=None*, *problematic_leading_principal_submatrix_index=None*, *subdecomposition=None*)

Bases: *matrix.errors.MatrixNoDecompositionPossibleWithProblematicSubdecompositionError*

A LL decomposition is not possible for this matrix.

3.4 MatrixDecompositionNoConversionImplementedError

class matrix.errors.**MatrixDecompositionNoConversionImplementedError**(*original_decomposition=None*, *desired_decomposition_type=None*)

Bases: *matrix.errors.MatrixError*

A decomposition conversion is not implemented for this type.

3.5 MatrixNoDecompositionPossibleWithProblematicSubdecompositionError

class matrix.errors.**MatrixNoDecompositionPossibleWithProblematicSubdecompositionError**(*matrix*, *decomposition_d*, *problematic_le*, *subdecomposition=N*)

Bases: *matrix.errors.MatrixNoDecompositionPossibleError*

The desired matrix decomposition is not possible for this matrix. Only a subdecompostion could be calculated

3.6 MatrixError

class matrix.errors.**MatrixError**(*matrix=None*, *message=None*)

Bases: Exception

An exception related to a matrix.

This is the base exception for all exceptions in this package.

CHAPTER 4

Changelog

4.1 v0.5

- matrices can now be approximated by decompositions

4.2 v0.4

- matrices can now be examined if they are positive definite or positive semi-definite

4.3 v0.3

- dense and sparse matrices are now decomposable into several types (LL, LDL, LDL compressed)

4.4 v0.2

- decompositons are now convertable to other decompositon types
- decompositions are now comparable

4.5 v0.1

- several decompositions types added (LL, LDL, LDL compressed)
- permutation capabilities added

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Index

L

M

N

P

S

T

U