
matrix_decomposition Documentation

Release 0.1

Joscha Reimer

Jul 19, 2018

Contents:

1	Functions	1
1.1	decompose	1
2	Matrix decompositions	3
2.1	LL decomposition	3
2.2	LDL decomposition	5
2.3	LDL decomposition compressed	7
2.4	base decomposition	9
3	Errors	13
3.1	MatrixNoDecompositionPossibleError	13
3.2	MatrixNoLDLDecompositionPossibleError	13
3.3	MatrixNoLLDecompositionPossibleError	13
3.4	MatrixDecompositionNoConversionImplementedError	14
3.5	MatrixNoDecompositionPossibleWithProblematicSubdecompositionError	14
3.6	MatrixError	14
4	Changelog	15
4.1	v0.3	15
4.2	v0.2	15
4.3	v0.1	15
5	Indices and tables	17

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

1.1 decompose

`matrix.calculate.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.constants.PERMUTATION_METHODS`. If A is sparse, it can also be a value in `matrix.sparse.constants.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.constants.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 decomposition of A of type *return_type*.

Return type `matrix.decompositions.DecompositionBase`

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

`matrix.constants.PERMUTATION_METHODS = (None, '', 'none', 'natural', 'decreasing_diagonal_`
Supported permutation methods for dense and sparse matrices.

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

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

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 $A[p[:, \text{np.newaxis}], p[\text{np.newaxis}, :]]$ 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 decomposition.

is_permuted

`bool` – Whether this is a decomposition with permutation.

is_sparse

`bool` – Whether this is a sparse decomposition.

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 undoes 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 decomposition 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 decomposition 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 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

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 decomposition.

is_permuted

bool – Whether this is a decomposition with permutation.

is_sparse

bool – Whether this is a sparse decomposition.

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. $A[p[:, \text{np.newaxis}], p[\text{np.newaxis}, :]]$ is the matrix A permuted by the permutation of the decomposition

p_inverse

numpy.ndarray – The permutation vector that undoes 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 decomposition 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 decomposition 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 $A[p[:, \text{np.newaxis}], p[\text{np.newaxis}, :]]$ 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 decomposition.

is_permuted

`bool` – Whether this is a decomposition with permutation.

is_sparse

`bool` – Whether this is a sparse decomposition.

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 undoes 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 decomposition 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 decomposition 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 decomposition.

is_permuted

`bool` – Whether this is a decomposition with permutation.

is_sparse

`bool` – Whether this is a sparse decomposition.

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 undoes 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 decomposition 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 decomposition 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*

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,    decompo-
                                                    sition_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, problem-
                                                    atic_leading_principal_submatrix_index=None,
                                                    subdecomposi-
                                                    tion=None)
```

Bases: *matrix.errors.MatrixNoDecompositionPossibleWithProblematicSubdecompositionError*

A LDL decomposition is not possible for this matrix.

3.3 MatrixNoLLDecompositionPossibleError

```
class matrix.errors.MatrixNoLLDecompositionPossibleError (matrix=None,    problem-
                                                    atic_leading_principal_submatrix_index=None,
                                                    subdecomposi-
                                                    tion=None)
```

Bases: *matrix.errors.MatrixNoDecompositionPossibleWithProblematicSubdecompositionError*

A LL decomposition is not possible for this matrix.

3.4 MatrixDecompositionNoConversionImplementedError

```
class matrix.errors.MatrixDecompositionNoConversionImplementedError(original_decomposition=None,  
                                                                    de-  
                                                                    sired_decomposition_type=None)
```

Bases: `matrix.errors.MatrixError`

A decomposition conversion is not implemented for this type.

3.5 MatrixNoDecompositionPossibleWithProblematicSubdecompositionError

```
class matrix.errors.MatrixNoDecompositionPossibleWithProblematicSubdecompositionError(matrix,  
                                                                    de-  
                                                                    com-  
                                                                    po-  
                                                                    si-  
                                                                    tion_d  
                                                                    prob-  
                                                                    lem-  
                                                                    atic_le  
                                                                    sub-  
                                                                    de-  
                                                                    com-  
                                                                    po-  
                                                                    si-  
                                                                    tion=N)
```

Bases: `matrix.errors.MatrixNoDecompositionPossibleError`

The desired matrix decomposition is not possible for this matrix. Only a subdecomposition 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.

4.1 v0.3

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

4.2 v0.2

- decompositions are now convertible to other decomposition types
- decompositions are now comparable

4.3 v0.1

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

CHAPTER 5

Indices and tables

- `genindex`
- `modindex`
- `search`

C

`composed_matrix` (matrix.decompositions.DecompositionBase attribute), 9
`composed_matrix` (matrix.decompositions.LDL_Decomposition attribute), 5
`composed_matrix` (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
`composed_matrix` (matrix.decompositions.LL_Decomposition attribute), 3
`copy()` (matrix.decompositions.DecompositionBase method), 10
`copy()` (matrix.decompositions.LDL_Decomposition method), 5
`copy()` (matrix.decompositions.LDL_DecompositionCompressed method), 8
`copy()` (matrix.decompositions.LL_Decomposition method), 3

D

`D` (matrix.decompositions.LDL_Decomposition attribute), 5
`d` (matrix.decompositions.LDL_Decomposition attribute), 6
`D` (matrix.decompositions.LDL_DecompositionCompressed attribute), 7
`d` (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
`decompose()` (in module matrix.calculate), 1
`decomposition_type` (matrix.decompositions.DecompositionBase attribute), 10
`decomposition_type` (matrix.decompositions.LDL_Decomposition attribute), 6
`decomposition_type` (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
`decomposition_type` (matrix.decompositions.LL_Decomposition attribute), 4
`is_permuted` (matrix.decompositions.DecompositionBase attribute), 10
`is_permuted` (matrix.decompositions.LDL_Decomposition attribute), 6
`is_permuted` (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
`is_permuted` (matrix.decompositions.LL_Decomposition attribute), 4
`is_sparse` (matrix.decompositions.DecompositionBase attribute), 10
`is_sparse` (matrix.decompositions.LDL_Decomposition attribute), 6
`is_sparse` (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
`is_sparse` (matrix.decompositions.LL_Decomposition attribute), 4
`is_type()` (matrix.decompositions.DecompositionBase method), 10
`is_type()` (matrix.decompositions.LDL_Decomposition method), 6
`is_type()` (matrix.decompositions.LDL_DecompositionCompressed method), 8
`is_type()` (matrix.decompositions.LL_Decomposition method), 4

L

`L` (matrix.decompositions.LDL_Decomposition attribute), 5
`L` (matrix.decompositions.LDL_DecompositionCompressed attribute), 7

- L (matrix.decompositions.LL_Decomposition attribute), 3
 - LD (matrix.decompositions.LDL_Decomposition attribute), 5
 - LD (matrix.decompositions.LDL_DecompositionCompressed attribute), 7
 - LDL_Decomposition (class in matrix.decompositions), 5
 - LDL_DecompositionCompressed (class in matrix.decompositions), 7
 - LL_Decomposition (class in matrix.decompositions), 3
- ## M
- MatrixDecompositionNoConversionImplementedError (class in matrix.errors), 14
 - MatrixError (class in matrix.errors), 14
 - MatrixNoDecompositionPossibleError (class in matrix.errors), 13
 - MatrixNoDecompositionPossibleWithProblematicSubdecompositionError (class in matrix.errors), 14
 - MatrixNoLLDLDecompositionPossibleError (class in matrix.errors), 13
 - MatrixNoLLDLDecompositionPossibleError (class in matrix.errors), 13
- ## N
- n (matrix.decompositions.DecompositionBase attribute), 10
 - n (matrix.decompositions.LDL_Decomposition attribute), 6
 - n (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
 - n (matrix.decompositions.LL_Decomposition attribute), 4
- ## P
- P (matrix.decompositions.DecompositionBase attribute), 9
 - p (matrix.decompositions.DecompositionBase attribute), 10
 - P (matrix.decompositions.LDL_Decomposition attribute), 5
 - p (matrix.decompositions.LDL_Decomposition attribute), 6
 - P (matrix.decompositions.LDL_DecompositionCompressed attribute), 7
 - p (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
 - P (matrix.decompositions.LL_Decomposition attribute), 3
 - p (matrix.decompositions.LL_Decomposition attribute), 4
 - p_inverse (matrix.decompositions.DecompositionBase attribute), 10
- p_inverse (matrix.decompositions.LDL_Decomposition attribute), 6
 - p_inverse (matrix.decompositions.LDL_DecompositionCompressed attribute), 8
 - p_inverse (matrix.decompositions.LL_Decomposition attribute), 4
- ## PERMUTATION_METHODS (in module matrix.constants), 1
- permute_matrix() (matrix.decompositions.DecompositionBase method), 10
 - permute_matrix() (matrix.decompositions.LDL_Decomposition method), 6
 - permute_matrix() (matrix.decompositions.LDL_DecompositionCompressed method), 8
 - permute_matrix() (matrix.decompositions.LL_Decomposition method), 4
- ## S
- ### SPARSE_PERMUTATION_METHODS (in module matrix.sparse.constants), 2
- ## T
- to() (matrix.decompositions.DecompositionBase method), 10
 - to() (matrix.decompositions.LDL_Decomposition method), 6
 - to() (matrix.decompositions.LDL_DecompositionCompressed method), 8
 - to() (matrix.decompositions.LL_Decomposition method), 4
 - to_any() (matrix.decompositions.DecompositionBase method), 11
 - to_any() (matrix.decompositions.LDL_Decomposition method), 7
 - to_any() (matrix.decompositions.LDL_DecompositionCompressed method), 9
 - to_any() (matrix.decompositions.LL_Decomposition method), 4
 - to_LDL_Decomposition() (matrix.decompositions.LDL_DecompositionCompressed method), 9
 - to_LDL_Decomposition() (matrix.decompositions.LL_Decomposition method), 4
 - to_LDL_DecompositionCompressed() (matrix.decompositions.LDL_Decomposition method), 6
 - to_LL_Decomposition() (matrix.decompositions.LDL_Decomposition method), 7
- ## U
- unpermute_matrix() (matrix.decompositions.DecompositionBase method), 10

method), [11](#)

`unpermute_matrix()` (`matrix.decompositions.LDL_Decomposition`
method), [7](#)

`unpermute_matrix()` (`matrix.decompositions.LDL_DecompositionCompressed`
method), [9](#)

`unpermute_matrix()` (`matrix.decompositions.LL_Decomposition`
method), [5](#)