

---

# **matrix\_decomposition Documentation**

***Release 0.1***

**Joscha Reimer**

**Jul 19, 2018**



---

## Contents:

---

<b>1</b>	<b>Matrix decompositions</b>	<b>1</b>
1.1	LL decomposition . . . . .	1
1.2	LDL decomposition . . . . .	3
1.3	LDL decomposition compressed . . . . .	5
1.4	base decomposition . . . . .	7
<b>2</b>	<b>Errors</b>	<b>11</b>
2.1	MatrixNoDecompositionPossibleError . . . . .	11
2.2	MatrixNoLDLDecompositionPossibleError . . . . .	11
2.3	MatrixNoLLDecompositionPossibleError . . . . .	11
2.4	MatrixDecompositionNoConversionImplementedError . . . . .	12
2.5	MatrixNoDecompositionPossibleWithProblematicSubdecompositionError . . . . .	12
2.6	MatrixError . . . . .	12
<b>3</b>	<b>Changelog</b>	<b>13</b>
3.1	v0.2 . . . . .	13
3.2	v0.1 . . . . .	13
<b>4</b>	<b>Indices and tables</b>	<b>15</b>



---

## Matrix decompositions

---

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

### 1.1 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*

## 1.2 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*

## 1.3 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 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*

## 1.4 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*:

### 2.1 MatrixNoDecompositionPossibleError

```
class matrix.errors.MatrixNoDecompositionPossibleError (matrix=None, decomposition_description=None, message=None)
```

Bases: *matrix.errors.MatrixError*

The matrix decomposition is not possible for this matrix.

### 2.2 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.

### 2.3 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.

## 2.4 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.

## 2.5 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

## 2.6 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.



#### 3.1 v0.2

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

#### 3.2 v0.1

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



## CHAPTER 4

---

### Indices and tables

---

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



C		D	
composed_matrix	(matrix.decompositions.DecompositionBase attribute), 7	decomposition_type	(matrix.decompositions.LL_Decomposition attribute), 2
composed_matrix	(matrix.decompositions.LDL_Decomposition attribute), 3	DecompositionBase	(class in matrix.decompositions), 7
composed_matrix	(matrix.decompositions.LDL_DecompositionCompressed attribute), 6	is_permuted	(matrix.decompositions.DecompositionBase attribute), 8
composed_matrix	(matrix.decompositions.LL_Decomposition attribute), 1	is_permuted	(matrix.decompositions.LDL_Decomposition attribute), 4
copy()	(matrix.decompositions.DecompositionBase method), 8	is_permuted	(matrix.decompositions.LDL_DecompositionCompressed attribute), 6
copy()	(matrix.decompositions.LDL_Decomposition method), 3	is_sparse	(matrix.decompositions.DecompositionBase attribute), 8
copy()	(matrix.decompositions.LDL_DecompositionCompressed method), 6	is_sparse	(matrix.decompositions.LDL_Decomposition attribute), 4
copy()	(matrix.decompositions.LL_Decomposition method), 1	is_sparse	(matrix.decompositions.LDL_DecompositionCompressed attribute), 6
		is_sparse	(matrix.decompositions.LL_Decomposition attribute), 2
D	(matrix.decompositions.LDL_Decomposition attribute), 3	is_type()	(matrix.decompositions.DecompositionBase method), 8
d	(matrix.decompositions.LDL_Decomposition attribute), 4	is_type()	(matrix.decompositions.LDL_Decomposition method), 4
D	(matrix.decompositions.LDL_DecompositionCompressed attribute), 5	is_type()	(matrix.decompositions.LDL_DecompositionCompressed method), 6
d	(matrix.decompositions.LDL_DecompositionCompressed attribute), 6	is_type()	(matrix.decompositions.LL_Decomposition method), 2
decomposition_type	(matrix.decompositions.DecompositionBase attribute), 8	L	(matrix.decompositions.LDL_Decomposition attribute), 3
decomposition_type	(matrix.decompositions.LDL_Decomposition attribute), 4	L	(matrix.decompositions.LDL_DecompositionCompressed attribute), 5
decomposition_type	(matrix.decompositions.LDL_DecompositionCompressed attribute), 1	L	(matrix.decompositions.LL_Decomposition attribute), 1

- LD (matrix.decompositions.LDL\_Decomposition attribute), 3
- LD (matrix.decompositions.LDL\_DecompositionCompressed attribute), 5
- LDL\_Decomposition (class in matrix.decompositions), 3
- LDL\_DecompositionCompressed (class in matrix.decompositions), 5
- LL\_Decomposition (class in matrix.decompositions), 1
- ## M
- MatrixDecompositionNoConversionImplementedError (class in matrix.errors), 12
- MatrixError (class in matrix.errors), 12
- MatrixNoDecompositionPossibleError (class in matrix.errors), 11
- MatrixNoDecompositionPossibleWithProblematicSubdecompositionError (class in matrix.errors), 12
- MatrixNoLDLDecompositionPossibleError (class in matrix.errors), 11
- MatrixNoLLDecompositionPossibleError (class in matrix.errors), 11
- ## N
- n (matrix.decompositions.DecompositionBase attribute), 8
- n (matrix.decompositions.LDL\_Decomposition attribute), 4
- n (matrix.decompositions.LDL\_DecompositionCompressed attribute), 6
- n (matrix.decompositions.LL\_Decomposition attribute), 2
- ## P
- P (matrix.decompositions.DecompositionBase attribute), 7
- p (matrix.decompositions.DecompositionBase attribute), 8
- P (matrix.decompositions.LDL\_Decomposition attribute), 3
- p (matrix.decompositions.LDL\_Decomposition attribute), 4
- P (matrix.decompositions.LDL\_DecompositionCompressed attribute), 5
- p (matrix.decompositions.LDL\_DecompositionCompressed attribute), 6
- P (matrix.decompositions.LL\_Decomposition attribute), 1
- p (matrix.decompositions.LL\_Decomposition attribute), 2
- p\_inverse (matrix.decompositions.DecompositionBase attribute), 8
- p\_inverse (matrix.decompositions.LDL\_Decomposition attribute), 4
- p\_inverse (matrix.decompositions.LDL\_DecompositionCompressed attribute), 6
- p\_inverse (matrix.decompositions.LL\_Decomposition attribute), 2
- permute\_matrix() (matrix.decompositions.DecompositionBase method), 8
- permute\_matrix() (matrix.decompositions.LDL\_Decomposition method), 4
- permute\_matrix() (matrix.decompositions.LDL\_DecompositionCompressed method), 6
- permute\_matrix() (matrix.decompositions.LL\_Decomposition method), 2
- ## T
- to() (matrix.decompositions.DecompositionBase method), 8
- to() (matrix.decompositions.LDL\_Decomposition method), 4
- to() (matrix.decompositions.LDL\_DecompositionCompressed method), 6
- to() (matrix.decompositions.LL\_Decomposition method), 2
- to\_any() (matrix.decompositions.DecompositionBase method), 9
- to\_any() (matrix.decompositions.LDL\_Decomposition method), 5
- to\_any() (matrix.decompositions.LDL\_DecompositionCompressed method), 7
- to\_any() (matrix.decompositions.LL\_Decomposition method), 2
- to\_LDL\_Decomposition() (matrix.decompositions.LDL\_DecompositionCompressed method), 7
- to\_LDL\_Decomposition() (matrix.decompositions.LL\_Decomposition method), 2
- to\_LDL\_DecompositionCompressed() (matrix.decompositions.LDL\_Decomposition method), 4
- to\_LL\_Decomposition() (matrix.decompositions.LDL\_Decomposition method), 5
- ## U
- unpermute\_matrix() (matrix.decompositions.DecompositionBase method), 9
- unpermute\_matrix() (matrix.decompositions.LDL\_Decomposition method), 5
- unpermute\_matrix() (matrix.decompositions.LDL\_DecompositionCompressed method), 7

unpermute\_matrix()  
trix.decompositions.LL-Decomposition  
method), 3