Package 'invertiforms'

November 25, 2022

Title Invertible Transforms for Matrices

Version 0.1.1

Description Provides composable invertible transforms for (sparse) matrices.

License MIT + file LICENSE

URL https://rohelab.github.io/invertiforms/, https://github.com/RoheLab/invertiforms

BugReports https://github.com/RoheLab/invertiforms/issues

Depends Matrix, methods

Imports sparseLRMatrix (>= 0.1.0), glue

Suggests covr, testthat (>= 3.0.0), igraph, igraphdata

Encoding UTF-8

RoxygenNote 7.2.1.9000

Collate 's4-generics.R' 'DoubleCenter.R' 'NormalizedLaplacian.R' 'PerturbedLaplacian.R' 'RegularizedLaplacian.R' 'invertiforms-package.R' 'utils.R'

Config/testthat/edition 3

NeedsCompilation no

Author Alex Hayes [aut, cre, cph] (<https://orcid.org/0000-0002-4985-5160>)

Maintainer Alex Hayes <alexpghayes@gmail.com>

Repository CRAN

Date/Publication 2022-11-25 12:40:10 UTC

R topics documented:

`DoubleCenter` *Construct and use DoubleCenter transformations*

Description

A convenience function to create DoubleCenter S4 objects, which are useful for **simultaneously row and column centering** a matrix.

Usage

```
DoubleCenter(A)

## S4 method for signature 'DoubleCenter,sparseMatrix'
transform(iform, A)

## S4 method for signature 'DoubleCenter,sparseLRMatrix'
inverse_transform(iform, A)

## S4 method for signature 'DoubleCenter,vsp_fa'
inverse_transform(iform, A)
```

Arguments

`A`	A matrix to transform.
`iform`	An Invertiform object describing the transformation.

Value

- `DoubleCenter()` creates a DoubleCenter object.
- `transform()` returns the transformed matrix, typically as a sparseLRMatrix::sparseLRMatrix.
- `inverse_transform()` returns the inverse transformed matrix, typically as a sparseLRMatrix::sparseLRMatrix in most cases. When possible reduces the sparseLRMatrix::sparseLRMatrix to a `Matrix::sparseMatrix()`.

Examples

```
library(igraph)
library(igraphdata)

data("karate", package = "igraphdata")

A <- get.adjacency(karate)

iform <- DoubleCenter(A)

A_tilde <- transform(iform, A)
A_recovered <- inverse_transform(iform, A_tilde)

all.equal(A, A_recovered)
```

DoubleCenter-class *Row and column centering transformation*

Description

Row and column centering transformation

Slots

`row_means` numeric.

`col_means` numeric.

`overall_mean` numeric.

---

inverse_transform *Apply the inverse of an invertible transformation*

Description

Apply the inverse of an invertible transformation

Usage

```
inverse_transform(iform, A)
```

Arguments

`iform`	An Invertiform object describing the transformation.
`A`	A matrix to inverse transform.

Value

The inverse transformed matrix.

`Invertiform-class` *An abstract S4 class representing an invertible transformation*

Description

An abstract S4 class representing an invertible transformation

---

`NormalizedLaplacian` *Construct and use the Normalized Laplacian*

Description

A convenience function to create NormalizedLaplacian S4 objects, which are useful for finding the normalized Laplacian of the adjacency matrix of a graph.

Usage

```
NormalizedLaplacian(A)

## S4 method for signature 'NormalizedLaplacian,sparseMatrix'
transform(iform, A)

## S4 method for signature 'NormalizedLaplacian,sparseMatrix'
inverse_transform(iform, A)
```

Arguments

`A`	A matrix to transform.
`iform`	An Invertiform object describing the transformation.

Details

We define the *normalized Laplacian* $L(A)$ of an $n \times n$ graph adjacency matrix A as

$$L(A)_{ij} = \frac{A_{ij}}{\sqrt{d_i^{out}}\sqrt{d_j^{in}}}$$

where

$$d_i^{out} = \sum_{j=1}^{n} \|A_{ij}\|$$

and

$$d_j^{in} = \sum_{i=1}^{n} \|A_{ij}\|.$$

When A_{ij} denotes the present of an edge *from* node i *to* node j, which is fairly standard notation, d_i^{out} denotes the (absolute) out-degree of node i and d_j^{in} denotes the (absolute) in-degree of node j.

Note that this documentation renders most clearly at https://rohelab.github.io/invertiforms/.

Value

- `NormalizedLaplacian()` creates a NormalizedLaplacian object.
- `transform()` returns the transformed matrix, typically as a Matrix.
- `inverse_transform()` returns the inverse transformed matrix, typically as a Matrix.

Examples

```
library(igraph)
library(igraphdata)

data("karate", package = "igraphdata")

A <- get.adjacency(karate)

iform <- NormalizedLaplacian(A)

L <- transform(iform, A)
A_recovered <- inverse_transform(iform, L)

all.equal(A, A_recovered)
```

`NormalizedLaplacian-class`

Normalized graph Laplacian transformation

Description

Normalized graph Laplacian transformation

Slots

`rsA` numeric.

`csA` numeric.

`PerturbedLaplacian` *Construct and use the Perturbed Laplacian*

Description

Construct and use the Perturbed Laplacian

Usage

```
PerturbedLaplacian(A, tau = NULL)

## S4 method for signature 'PerturbedLaplacian,sparseMatrix'
transform(iform, A)

## S4 method for signature 'PerturbedLaplacian,sparseLRMatrix'
inverse_transform(iform, A)
```

Arguments

`A`	A matrix to transform.
`tau`	Additive regularizer for row and column sums of `abs(A)`. Typically this corresponds to inflating the (absolute) out-degree and the (absolute) in-degree of each node by `tau`. Defaults to `NULL`, in which case we set `tau` to the mean value of `abs(A)`.
`iform`	An Invertiform object describing the transformation.

Details

We define the *perturbed Laplacian* $L^\tau(A)$ of an $n \times n$ graph adjacency matrix A as

$$L^\tau(A)_{ij} = \frac{A_{ij} + \frac{\tau}{n}}{\sqrt{d_i^{out} + \tau}\sqrt{d_j^{in} + \tau}}$$

where

$$d_i^{out} = \sum_{j=1}^{n} \|A_{ij}\|$$

and

$$d_j^{in} = \sum_{i=1}^{n} \|A_{ij}\|.$$

When A_{ij} denotes the present of an edge *from* node i *to* node j, which is fairly standard notation, d_i^{out} denotes the (absolute) out-degree of node i and d_j^{in} denotes the (absolute) in-degree of node j.

Note that this documentation renders more clearly at https://rohelab.github.io/invertiforms/.

Value

- `PerturbedLaplacian()` creates a PerturbedLaplacian object.
- `transform()` returns the transformed matrix, typically as a Matrix.
- `inverse_transform()` returns the inverse transformed matrix, typically as a Matrix.

Examples

```
library(igraph)
library(igraphdata)

data("karate", package = "igraphdata")

A <- get.adjacency(karate)

iform <- PerturbedLaplacian(A)

L <- transform(iform, A)
L

## Not run:
A_recovered <- inverse_transform(iform, L)
all.equal(A, A_recovered)

## End(Not run)
```

`PerturbedLaplacian-class`

Perturbed graph Laplacian transformation

Description

Perturbed graph Laplacian transformation

Slots

`tau` numeric.

`rsA` numeric.

`csA` numeric.

`tau_choice` character.

RegularizedLaplacian *Construct and use the Regularized Laplacian*

Description

Construct and use the Regularized Laplacian

Usage

```
RegularizedLaplacian(A, tau_row = NULL, tau_col = NULL)

## S4 method for signature 'RegularizedLaplacian,Matrix'
transform(iform, A)

## S4 method for signature 'RegularizedLaplacian,matrix'
transform(iform, A)

## S4 method for signature 'RegularizedLaplacian,sparseLRMatrix'
transform(iform, A)

## S4 method for signature 'RegularizedLaplacian,Matrix'
inverse_transform(iform, A)

## S4 method for signature 'RegularizedLaplacian,matrix'
inverse_transform(iform, A)

## S4 method for signature 'RegularizedLaplacian,vsp_fa'
inverse_transform(iform, A)
```

Arguments

A	A matrix to transform.
tau_row	Additive regularizer for row sums of abs(A). Typically this corresponds to inflating the (absolute) out-degree of each node by tau_row. Defaults to NULL, in which case we set tau_row to the mean (absolute) row sum of A.
tau_col	Additive regularizer for column sums of abs(A). Typically this corresponds to inflating the (absolute) in-degree of each node by tau_col. Defaults to NULL, in which case we set tau_col to the mean (absolute) column sum of A.
iform	An Invertiform object describing the transformation.

Details

We define the *regularized Laplacian* $L^\tau(A)$ of an $n \times n$ graph adjacency matrix A as

$$L^\tau(A)_{ij} = \frac{A_{ij}}{\sqrt{d_i^{out} + \tau_{row}}\sqrt{d_j^{in} + \tau_{col}}}$$

where

$$d_i^{out} = \sum_{j=1}^{n} \|A_{ij}\|$$

and

$$d_j^{in} = \sum_{i=1}^{n} \|A_{ij}\|.$$

When A_{ij} denotes the present of an edge *from* node i *to* node j, which is fairly standard notation, d_i^{out} denotes the (absolute) out-degree of node i and d_j^{in} denotes the (absolute) in-degree of node j. Then τ_{row} is an additive out-degree regularizer and τ_{col} is an additive in-degree regularizer.

Note that this documentation renders more clearly at https://rohelab.github.io/invertiforms/.

Value

- `RegularizedLaplacian()` creates a RegularizedLaplacian object.
- `transform()` returns the transformed matrix, typically as a Matrix.
- `inverse_transform()` returns the inverse transformed matrix, typically as a Matrix.

Examples

```
library(igraph)
library(igraphdata)

data("karate", package = "igraphdata")

A <- get.adjacency(karate)

iform <- RegularizedLaplacian(A)

L <- transform(iform, A)
L

A_recovered <- inverse_transform(iform, L)

all.equal(A, A_recovered)
```

RegularizedLaplacian-class *Regularized graph Laplacian transformation*

Description

Regularized graph Laplacian transformation

Slots

`tau_row` numeric.

`tau_col` numeric.

`rsA` numeric.

`csA` numeric.

`tau_choice_row` character.

`tau_choice_col` character.

transform *Apply an invertible transformation*

Description

Apply an invertible transformation

Usage

```
transform(iform, A)
```

Arguments

`iform`	An Invertiform object describing the transformation.
`A`	A matrix to transform.

Value

The transformed matrix.

Index