

Package ‘LRMF3’

July 21, 2025

Title Low Rank Matrix Factorization S3 Objects

Version 0.1.0

Description Provides S3 classes to represent low rank matrix decompositions.

License MIT + file LICENSE

URL <https://github.com/RoheLab/LRMF3>

BugReports <https://github.com/RoheLab/LRMF3/issues>

Depends Matrix, R (>= 3.1)

Imports glue

Suggests covr, testthat

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

Config/testthat/edition 3

NeedsCompilation no

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

Maintainer Alex Hayes <alexpgghayes@gmail.com>

Repository CRAN

Date/Publication 2022-02-09 19:30:05 UTC

Contents

as_fa_like	2
as_svd_like	2
dim_and_class	3
fa_like	3
ml100k	4
svd_like	4

Index	6
--------------	----------

as_fa_like	<i>Coerce an object to a factor analysis like factorization</i>
------------	---

Description

Coerce an object to a factor analysis like factorization

Usage

```
as_fa_like(x, ...)  
  
## S3 method for class 'list'  
as_fa_like(x, ...)
```

Arguments

x	Object to coerce
...	Ignored.

Value

Object as [svd_like\(\)](#) object.

as_svd_like	<i>Coerce an object to LRMF class</i>
-------------	---------------------------------------

Description

Coerce an object to LRMF class

Usage

```
as_svd_like(x, ...)  
  
## S3 method for class 'list'  
as_svd_like(x, ...)
```

Arguments

x	Object to coerce
...	Ignored.

Value

Object as [svd_like\(\)](#) object.

dim_and_class	<i>Description array size and type</i>
---------------	--

Description

Description array size and type

Usage

```
dim_and_class(x)
```

Arguments

x	Matrix or vector
---	------------------

Value

Description as character vector

fa_like	<i>Create a Factor Analysis-like low rank matrix factorization object</i>
---------	---

Description

A low rank matrix factorization of a matrix X is parameterized by $X \approx X \%*\% B \%*\% t(Y)$. The object is "factor analysis-like" because the middle matrix in the decomposition is arbitrary rather than diagonal.

Usage

```
fa_like(Z, B, Y, subclasses = NULL, ...)
```

Arguments

Z	A <i>matrix</i> of embeddings for each observation.
B	A mixing <i>matrix</i> describing how observation embeddings and topics interact. Does not have to be diagonal!
Y	A <i>matrix</i> describing the compositions of various topics or factors.
subclasses	A character vector of subclasses. Optional, defaults to NULL.
...	Optional additional items to pass to the constructor.

Examples

```
s <- svd(as.matrix(trees))
fa_like(s$u, diag(s$d), s$v)
```

m1100k	<i>MovieLens 100K dataset</i>
--------	-------------------------------

Description

Standard benchmarking dataset for recommendation systems. 100k movie ratings on 1682 movies by 943 users. Each user has rated at least 20 movies.

Usage

m1100k

Format

An object of class dgCMatrix with 943 rows and 1682 columns.

Details

Stored as a Matrix::dgCMatrix object, which is a sparse matrix. Each row corresponds to a user and each column to a movie.

References

F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=<http://dx.doi.org/10.1145/2827872>
<https://grouplens.org/datasets/movielens/100k/>

svd_like	<i>Create a SVD-like low rank matrix factorization object</i>
----------	---

Description

A low rank matrix factorization of a matrix X is parameterized by $X \approx u \% \% \text{diag}(d) \% \% t(v)$. The object is "svd-like" because the middle matrix in the decomposition must be strictly diagonal.

Usage

```
svd_like(u, d, v, subclasses = NULL, ...)
```

Arguments

u	A <i>matrix</i> "left singular-ish" vectors.
d	A <i>vector</i> of "singular-ish" values.
v	A <i>matrix</i> of "right singular-ish" vectors.
subclasses	A character vector of subclasses. Optional, defaults to NULL.
...	Optional additional items to pass to the constructor.

Examples

```
s <- svd(as.matrix(trees))

# using the constructor directly
svd_like(s$u, s$d, s$v)

# coercing svd-like lists
as_svd_like(s)
```

Index

* **datasets**

ml100k, [4](#)

as_fa_like, [2](#)

as_svd_like, [2](#)

dim_and_class, [3](#)

fa_like, [3](#)

ml100k, [4](#)

svd_like, [4](#)

svd_like(), [2](#)