Package ‘eimpute’

October 22, 2022

Type Package

Title Efficiently Impute Large Scale Incomplete Matrix

Version 0.2.2

Date 2022-10-22

Author Zhe Gao [aut, cre],
        Jin Zhu [aut],
        Junxian Zhu [aut],
        Xueqin Wang [aut],
        Yixuan Qiu [cph],
        Gael Guennebaud [cph, ctb],
        Jitse Niesen [cph, ctb],
        Ray Gardner [ctb]

Maintainer Zhe Gao <gaozh8@mail2.sysu.edu.cn>

Description Efficiently impute large scale matrix with missing values via its unbiased low-rank matrix approximation. Our main approach is Hard-Impute algorithm proposed in <https://www.jmlr.org/papers/v11/mazumder10a.html>, which achieves highly computational advantage by truncated singular-value decomposition.

License GPL-3 | file LICENSE

Imports Rcpp (>= 0.12.6)

LinkingTo Rcpp, RcppEigen

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation yes

Suggests knitr

VignetteBuilder knitr

Repository CRAN

Date/Publication 2022-10-22 08:15:06 UTC
### R topics documented:

- `biscale` .......................................................... 2
- `biscale.control` .................................................. 3
- `eimpute` .......................................................... 4
- `incomplete.generator` .......................................... 5
- `r.search` .......................................................... 6

### Index

<table>
<thead>
<tr>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>biscale</code></td>
<td>2</td>
</tr>
<tr>
<td><code>eimpute</code></td>
<td>4</td>
</tr>
<tr>
<td><code>incomplete.generator</code></td>
<td>5</td>
</tr>
<tr>
<td><code>r.search</code></td>
<td>6</td>
</tr>
</tbody>
</table>

### Description

Standardize a matrix rows and/or columns to have zero mean or unit variance

### Usage

```r
biscale(x, thresh.sd = 1e-05, maxit.sd = 100, control = list(...), ...)  
```

### Arguments

- **x**
  - An `m` by `n` matrix possibly with NAs.
- **thresh.sd**
  - Convergence threshold, measured as the relative change in the Frobenius norm between two successive estimates.
- **maxit.sd**
  - Maximum number of iterations.
- **control**
  - A list of parameters that control details of standard procedure. See `biscale.control`.
- **...**
  - Arguments to be used to form the default control argument if it is not supplied directly.

### Value

A list is returned

- **x.st**
  - The matrix after standardization.
- **alpha**
  - The row mean after iterative process.
- **beta**
  - The column mean after iterative process.
- **tau**
  - The row standard deviation after iterative process.
- **gamma**
  - The column standard deviation after iterative process.

### References

**Examples**

```r
# Quick Start
m <- 100
g <- 100
r <- 10
x_na <- incomplete.generator(m, n, r)

##### Standardize both mean and variance
xs <- biscale(x_na)

##### Only standardize mean ######
x_mean <- biscale(x_na, row.mean = TRUE, col.mean = TRUE)

##### Only standardize variance ######
x_std <- biscale(x_na, row.std = TRUE, col.std = TRUE)
```

**Control for standard procedure**

**Description**

Various parameters that control aspects of the standard procedure.

**Usage**

```r
biscale.control(
  row.mean = FALSE,
  row.std = FALSE,
  col.mean = FALSE,
  col.std = FALSE
)
```

**Arguments**

- **row.mean** if `row.mean = TRUE` (the default), row centering will be performed resulting in a matrix with row means zero. If `row.mean` is a vector, it will be used in the iterative process. If `row.mean = FALSE` nothing is done.
- **row.std** if `row.std = TRUE`, row scaling will be performed resulting in a matrix with row variance one. If `row.std` is a vector, it will be used in the iterative process. If `row.std = FALSE` (the default) nothing is done.
- **col.mean** similar to `row.mean`.
- **col.std** similar to `row.std`.

**Value**

A list with components named as the arguments.
Efficiently impute missing values for a large scale matrix

Description

Fit a low-rank matrix approximation to a matrix with missing values. The algorithm iterates like EM: filling the missing values with the current guess, and then approximating the complete matrix via truncated SVD.

Usage

eimpute(
  x,
  r,
  svd.method = c("tsvd", "rsvd"),
  noise.var = 0,
  thresh = 1e-05,
  maxit = 100,
  init = FALSE,
  init.mat = 0,
  override = FALSE,
  control = list(...),
  ...)

Arguments

x an \( m \) by \( n \) matrix with NAs.
r the rank of low-rank matrix for approximating \( x \)
svd.method a character string indicating the truncated SVD method. If \( \text{svd.method} = \text{"rsvd"} \), a randomized SVD is used, else if \( \text{svd.method} = \text{"tsvd"} \), standard truncated SVD is used. Any unambiguous substring can be given. Default \( \text{svd.method} = \text{"tsvd"} \).
noise.var the variance of noise.
thresh convergence threshold, measured as the relative change in the Frobenius norm between two successive estimates.
maxit maximal number of iterations.
init if init = FALSE (the default), the missing entries will initialize with mean.
init.mat the initialization matrix.
override logical value indicating whether the observed elements in \( x \) should be overwritten by its low-rank approximation.
control a list of parameters that control details of standard procedure, See \texttt{biscale.control}.
... arguments to be used to form the default control argument if it is not supplied directly.
incomplete.generator

Value

A list containing the following components

- `x.imp` the matrix after completion.
- `rmse` the relative mean square error of matrix completion, i.e., training error.
- `iter.count` the number of iterations.

References


Examples

`# Quick Start` `#`
```r
m <- 100
n <- 100
r <- 10
x_na <- incomplete.generator(m, n, r)
head(x_na[, 1:6])
x_impute <- eimpute(x_na, r)
head(x_impute[['x.imp']][, 1:6])
x_impute[['rmse']]```

---

incomplete.generator  
Incomplete data generator

Description

Generate a matrix with missing values, where the indices of missing values are uniformly randomly distributed in the matrix.

Usage

`incomplete.generator(m, n, r, snr = 3, prop = 0.5, seed = 1)`

Arguments

- `m` the rows of the matrix.
- `n` the columns of the matrix.
- `r` the rank of the matrix.
- `snr` the signal-to-noise ratio in generating the matrix. Default `snr = 3`.
- `prop` the proportion of missing observations. Default `prop = 0.5`.
- `seed` the random seed. Default `seed = 1`. 
Details

We generate the matrix by $UV + \epsilon$, where $U, V$ are $m$ by $r$, $r$ by $n$ matrix satisfy standard normal distribution. $\epsilon$ has a normal distribution with mean 0 and variance $\frac{r}{snr}$.

Value

A matrix with missing values.

Examples

```r
m <- 100
n <- 100
r <- 10
x_na <- incomplete.generator(m, n, r)
head(x_na[, 1:6])
```

---

**r.search**

*Search rank magnitude of the best approximating matrix*

Description

Estimate a preferable matrix rank magnitude for fitting a low-rank matrix approximation to a matrix with missing values. The algorithm use GIC/CV to search the rank in a given range, and then fill the missing values with the estimated rank.

Usage

```r
r.search(
  x,
  r.min = 1,
  r.max = "auto",
  svd.method = c("tsvd", "rsvd"),
  rule.type = c("gic", "cv"),
  noise.var = 0,
  init = FALSE,
  init.mat = 0,
  maxit.rank = 1,
  nfolds = 5,
  thresh = 1e-05,
  maxit = 100,
  override = FALSE,
  control = list(...),
  ...
)
```
Arguments

x an \( m \) by \( n \) matrix with NAs.

r.min the start rank for searching. Default \( r.min = 1 \).

r.max the max rank for searching.

svd.method a character string indicating the truncated SVD method. If \( svd.method = "rsvd" \), a randomized SVD is used, else if \( svd.method = "tsvd" \), standard truncated SVD is used. Any unambiguous substring can be given. Default \( svd.method = "tsvd" \).

rule.type a character string indicating the information criterion rule. If \( rule.type = "gic" \), generalized information criterion rule is used, else if \( rule.type = "cv" \), cross validation is used. Any unambiguous substring can be given. Default \( rule.type = "gic" \).

noise.var the variance of noise.

init if init = FALSE (the default), the missing entries will initialize with mean.

init.mat the initialization matrix.

maxit.rank maximal number of iterations in searching rank. Default \( maxit.rank = 1 \).

nfolds number of folds in cross validation. Default \( nfolds = 5 \).

thresh convergence threshold, measured as the relative change in the Frobenius norm between two successive estimates.

maxit maximal number of iterations.

override logical value indicating whether the observed elements in \( x \) should be overwritten by its low-rank approximation.

control a list of parameters that control details of standard procedure, See \texttt{biscale.control}.

... arguments to be used to form the default control argument if it is not supplied directly.

Value

A list containing the following components

x.imp the matrix after completion with the estimated rank.

r.est the rank estimation.

rmse the relative mean square error of matrix completion, i.e., training error.

iter.count the number of iterations.

Examples

############################################################ # Quick Start ###################################################
m <- 100
n <- 100
r <- 10
x_na <- incomplete.generator(m, n, r)
head(x_na[, 1:6])
x_impute <- r.search(x_na, 1, 15, "rsvd", "gic")
x_impute["r.est"]
Index

biscale, 2
biscale.control, 2, 3, 4, 7

eimpute, 4

incomplete.generator, 5

r.search, 6