Package ‘rsvddpd’

October 27, 2021

Type Package
Encoding UTF-8
Title Robust Singular Value Decomposition using Density Power Divergence
Version 1.0.0
Date 2021-10-23
Description Computing singular value decomposition with robustness is a challenging task. This package provides an implementation of computing robust SVD using density power divergence (<arXiv:2109.10680>). It combines the idea of robustness and efficiency in estimation based on a tuning parameter. It also provides utility functions to simulate various scenarios to compare performances of different algorithms.

License MIT + file LICENSE
Imports Rcpp (>= 1.0.5), MASS, stats, utils, matrixStats
LinkingTo Rcpp, RcppArmadillo
RoxygenNote 7.1.1
Suggests knitr, rmarkdown, microbenchmark, pcaMethods
VignetteBuilder knitr

URL https://github.com/subroy13/rsvddpd
BugReports https://github.com/subroy13/rsvddpd/issues

R topics documented:

AddOutlier ......................................................... 2
cv.alpha .......................................................... 3
rSVDdpd ............................................................ 3
simSVD ............................................................. 5
**Description**

AddOutlier returns a matrix with outliers randomly added to a matrix given certain proportion of contamination.

**Usage**

AddOutlier(X, proportion, value, seed = NULL, method = "element")

**Arguments**

- **X**: matrix, to which outliers are added.
- **proportion**: numeric, proportion of elements, rows or columns to be contaminated. Must be between 0 and 1.
- **value**: numeric, the outlying value to be used for contamination.
- **seed**: numeric, a seed to reproduce the randomization behaviour.
- **method**: character, must be one of the following:
  - "element" - For contaminating at random positions of the matrix.
  - "row" - For contaminating an entire row of the matrix.
  - "col" - For contaminating an entire column of the matrix.

**Value**

A matrix with elements / rows / columns contaminated.

**Note**

Due to randomization, it is possible that none of the entries of the matrix become contaminated. In that case, it is recommended to use different seed value.

**Examples**

X = matrix(1:20, nrow = 4, ncol = 5)
AddOutlier(X, 0.5, 10, seed = 1234)
cv.alpha

*Calculate optimal robustness parameter*

**Description**

`cv.alpha` returns the optimal robustness parameter.

**Usage**

`cv.alpha(X, alphas = 10)`

**Arguments**

- **X** matrix, whose singular value decomposition is required
- **alphas** numeric vector, vector of robustness parameters to try.

**Value**

A list containing

- The choices of the robust parameters.
- Corresponding cross validation score.
- Best choice of the robustness parameter.

**References**


---

rSVDdpd

*Robust Singular Value Decomposition using Density Power Divergence*

**Description**

`rSVDdpd` returns the singular value decomposition of a matrix with robust singular values in presence of outliers.
Usage

```r
rSVDdpd(
  X,
  alpha,
  nd = NA,
  tol = 1e-04,
  eps = 1e-04,
  maxiter = 100L,
  initu = NULL,
  initv = NULL
)
```

Arguments

- `X` matrix, whose singular value decomposition is required
- `alpha` numeric, robustness parameter between 0 and 1. See details for more.
- `nd` integer, must be lower than `nrow(X)` and `ncol(X)` both. If NA, defaults to `min(nrow(X),ncol(X))`
- `tol` numeric, a tolerance level. If the residual matrix has lower norm than this, then subsequent singular values will be taken as 0.
- `eps` numeric, a tolerance level for the convergence of singular vectors. If in subsequent iterations the singular vectors do not change its norm beyond this, then the iteration will stop.
- `maxiter` integer, upper limit to the maximum number of iterations.
- `initu` matrix, initializing vectors for left singular values. Must be of dimension `nrow(X) \times \min(nrow(X),ncol(X))`. If NULL, defaults to random initialization.
- `initv` matrix, initializing vectors for right singular values. Must be of dimension `ncol(X) \times \min(nrow(X),ncol(X))`. If NULL, defaults to random initialization.

Details

The usual singular value decomposition is highly prone to error in presence of outliers, since it tries to minimize the $L_2$ norm of the errors between the matrix $X$ and its best lower rank approximation. While there is considerable effort to impose robustness using $L_1$ norm of the errors instead of $L_2$ norm, such estimation lacks efficiency. Application of density power divergence bridges the gap.

$$DPD(f|g) = \int f^{(1+\alpha)} - (1 + \frac{1}{\alpha}) \int f^\alpha g + \frac{1}{\alpha} \int g^{(1+\alpha)}$$

The parameter `alpha` should be between 0 and 1, if not, then a warning is shown. Lower `alpha` means less robustness but more efficiency in estimation, while higher `alpha` means high robustness but less efficiency in estimation. The recommended value of `alpha` is 0.3. The function tries to obtain the best rank one approximation of a matrix by minimizing this density power divergence of the true errors with that of a normal distribution centered at the origin.
**Value**

A list containing different components of the decomposition $X = UDV'$

- $d$ - The robust singular values, namely the diagonal entries of $D$.
- $u$ - The matrix of left singular vectors $U$. Each column is a singular vector.
- $v$ - The matrix of right singular vectors $V$. Each column is a singular vector.

**References**


**See Also**

`svd`

**Examples**

```r
X = matrix(1:20, nrow = 4, ncol = 5)
rSVDdpd(X, alpha = 0.3)
```

---

**simSVD**  
*Simulate SVD and measure performances of various algorithms*

**Description**

`simSVD` simulates various models for the errors in the data matrix, and summarize performance of a singular value decomposition algorithm under presence or absence of outlying data introduced through various outlying schemes, using Monte Carlo approach.

**Usage**

```r
simSVD(
  trueSVD,  
  svdfun,  
  B = 100,  
  seed = NULL,  
  dist = "normal",  
  tau = 0.95,  
  outlier = FALSE,  
  out_method = "element",  
  out_value = 10,  
  out_prop = 0.1,  
  return_details = FALSE,  
  ...
)
```
Arguments

trueSVD list, containing three different named components.
  • d - a vector containing the singular values.
  • u - a matrix with left singular vectors, each column being a singular vector.
  • v - a matrix with right singular vectors, each column being a singular vector.

svdfun function which takes a numeric matrix as first argument and returns singular value decomposition of it as a list, with three components d, u and v as indicated before.

B numeric, denoting the number of Monte Carlo simulation.

seed numeric, a seed value used for reproducibility.

dist character string, denoting the distribution from which errors will be generated. It must be equal to one of the following: normal, cauchy, exp, logis, lognormal

tau numeric, a value between 0 and 1, see details for more.

outlier logical, if TRUE, simulates the situation by adding outliers.

out_method character, the method to add outliers. Must be one of "element", "row" or "col". See AddOutlier for details.

out_value numeric, the outlying observation. See AddOutlier for details.

out_prop a numeric, between 0 and 1 denoting the proportion of contamination. See AddOutlier for details.

return_details logical, whether to return detailed results for each Monte Carlo simulation. See value for details.

... extra arguments to be passed to svdfun function.

Value

Based on whether return_details is TRUE or FALSE, returns a list with two or one components.

• Simulations :
  – Lambda - A matrix containing obtained singular values from all Monte Carlo Simulations.
  – Left - A matrix containing the dissimilarities between left singular vectors of true SVD and obtained SVD.
  – Right - A matrix containing the dissimilarities between right singular vectors of true SVD and obtained SVD.

• Summary :
  – Bias - A numeric vector showing biases of the singular vectors obtained by svdfun algorithm.
  – MSE - A numeric vector showing MSE of the singular vectors obtained by svdfun algorithm.
  – Variance - A numeric vector showing variances of the singular vectors obtained by svdfun algorithm.
- Left - A numeric vector showing average dissimilarities between true and estimated left singular vectors.
- Right - A numeric vector showing average dissimilarities between true and estimated right singular vectors.

If `return_details` is FALSE, only Summary component of the larger list is returned.
Index

AddOutlier, 2, 6
cauchy, 6
cv.alpha, 3
exp, 6
logis, 6
lognormal, 6
normal, 6
rSVDdpd, 3
simSVD, 5
svd, 5