Package ‘rpca’

July 31, 2015

Type Package

Title RobustPCA: Decompose a Matrix into Low-Rank and Sparse Components

Version 0.2.3

Date 2015-07-19

Author Maciek Sykulski [aut, cre]

Maintainer Maciek Sykulski <macieksk@gmail.com>

Description Suppose we have a data matrix, which is the superposition of a low-rank component and a sparse component. Candes, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal component analysis?. Journal of the ACM (JACM), 58(3), 11. prove that we can recover each component individually under some suitable assumptions. It is possible to recover both the low-rank and the sparse components exactly by solving a very convenient convex program called Principal Component Pursuit; among all feasible decompositions, simply minimize a weighted combination of the nuclear norm and of the L1 norm. This package implements this decomposition algorithm resulting with Robust PCA approach.

License GPL-2 | GPL-3

Imports compiler

NeedsCompilation no

Repository CRAN

Date/Publication 2015-07-31 01:15:38

R topics documented:

rpca-package ........................................... 2
F2norm .................................................. 3
rpca ..................................................... 4
thresh.11 .............................................. 6
thresh.nuclear ........................................ 7

Index 9
Description

Suppose we have a data matrix, which is the superposition of a low-rank component and a sparse component. Candes, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal component analysis?. Journal of the ACM (JACM), 58(3), 11. prove that we can recover each component individually under some suitable assumptions. It is possible to recover both the low-rank and the sparse components exactly by solving a very convenient convex program called Principal Component Pursuit; among all feasible decompositions, simply minimize a weighted combination of the nuclear norm and of the L1 norm. This package implements this decomposition algorithm resulting with Robust PCA approach.

Details

Index of help topics:

- F2norm: Frobenius norm of a matrix
- rpca: Decompose a matrix into a low-rank component and a sparse component by solving Principal Components Pursuit
- rpca-package: RobustPCA: Decompose a Matrix into Low-Rank and Sparse Components
- thresh.l1: Shrinkage operator
- thresh.nuclear: Thresholding operator

This package contains rpca function,

which decomposes a rectangular matrix $M$ into a low-rank component, and a sparse component, by solving a convex program called Principal Component Pursuit:

$$\text{minimize} \quad \|L\|_* + \lambda \|S\|_1$$

subject to $L + S = M$

where $\|L\|_*$ is the nuclear norm of $L$ (sum of singular values).

Note

Use citation("rpca") to cite this R package.

Author(s)

Maciek Sykulski [aut, cre]
Maintainer: Maciek Sykulski <macieksk@gmail.com>
References


See Also

rpca

Description

Frobenius norm of a matrix.

Usage

F2norm(M)

Arguments

M A matrix.

Value

Frobenius norm of M.

Examples

## The function is currently defined as
function (M)
  sqrt(sum(M^2))

F2norm(matrix(runif(100),nrow=5))
**rpca**

*Decompose a matrix into a low-rank component and a sparse component by solving Principal Components Pursuit*

**Description**

This function decomposes a rectangular matrix $M$ into a low-rank component, and a sparse component, by solving a convex program called Principal Component Pursuit.

**Usage**

```r
rpca(M,
    lambda = 1/sqrt(max(dim(M))),
    mu = prod(dim(M))/(4 * sum(abs(M))),
    term.delta = 10^(-7),
    max.iter = 5000,
    trace = FALSE,
    thresh.nuclear.fun = thresh.nuclear,
    thresh.l1.fun = thresh.l1,
    F2norm.fun = F2norm)
```

**Arguments**

- **M**: a rectangular matrix that is to be decomposed into a low-rank component and a sparse component $M = L + S$.
- **lambda**: parameter of the convex problem $\|L\|_* + \lambda \|S\|_1$ which is minimized in the Principal Components Pursuit algorithm. The default value is the one suggested in Candès, E. J., section 1.4, and together with reasonable assumptions about $L$ and $S$ guarantees that a correct decomposition is obtained.
- **mu**: parameter from the augmented Lagrange multiplier formulation of the PCP, Candès, E. J., section 5. Default value is the one suggested in references.
- **term.delta**: The algorithm terminates when $\|M - L - S\|_F \leq \delta \|M\|_F$ where $\| \|$ is Frobenius norm of a matrix.
- **max.iter**: Maximal number of iterations of the augmented Lagrange multiplier algorithm. A warning is issued if the algorithm does not converge by then.
- **trace**: Print out information with every iteration.
- **thresh.nuclear.fun**, **thresh.l1.fun**, **F2norm.fun**: Arguments for internal use only.

**Details**

These functions decompose a rectangular matrix $M$ into a low-rank component, and a sparse component, by solving a convex program called Principal Component Pursuit:

\[
\begin{align*}
\text{minimize} & \quad \|L\|_* + \lambda \|S\|_1 \\
\text{subject to} & \quad L + S = M
\end{align*}
\]

where $\|L\|_*$ is the nuclear norm of $L$ (sum of singular values).
The function returns two matrices $S$ and $L$, which have the property that $L + S \simeq M$, where the quality of the approximation depends on the argument `term.delta`, and the convergence of the algorithm.

$S$  The sparse component of the matrix decomposition.
$L$  The low-rank component of the matrix decomposition.
$L.svd$  The singular value decomposition of $L$, as returned by the function `la.svd`.
$convergence$converged  TRUE if the algorithm converged with respect to `term.delta`.
$convergence$iterations  Number of performed iterations.
$convergence$final.delta  The final iteration `delta` which is compared with `term.delta`.
$convergence$all.delta  All `delta` from all iterations.

Author(s)

Maciek Sykulski [aut, cre]

References


Examples

data(iris)
M <- as.matrix(iris[,1:4])
Mcent <- sweep(M,2,colMeans(M))

res <- rpca(Mcent)

## Check convergence and number of iterations
with(res$convergence,list(converged,iterations))
## Final delta F2 norm divided by F2norm(Mcent)
with(res$convergence,final.delta)

## Check properites of the decomposition
with(res,c(
  all(abs( L+S - Mcent ) < 10^-5),
  all( L == L.svd$d%*%(L.svd$d*L.svd$vt) )
))
# [1] TRUE TRUE

## The low rank component has rank 2
length(res$L.svd$D)

## However, the sparse component is not sparse
## - thus this data set is not the best example here.
mean(res$S==0)

## Plot the first (the only) two principal components
## of the low-rank component
rpc<-(res$L.svd$D$U%*%diag(res$L.svd$D$D))
plot(jitter(rpc[,1:2],amount=.001),col=iris[,5])

## Compare with classical principal components
pc<-prcomp(M,center=TRUE)
plot(pc$x[,1:2],col=iris[,5])
points(rpc[,1:2],col=iris[,5],pch="+")

## "Sparse" elements distribution
plot(density(abs(res$S),from=0))
curve(dexp(x,rate=1/mean(abs(res$S))),add=TRUE,lty=2)

## Plot measurements against measurements corrected by sparse components
par(mfcol=c(2,2))
for(i in 1:T) {
  plot(M[,i],M[,i]-res$S[,i],col=iris[,5],xlab=colnames(M)[i])
}

thresh.l1

## Shrinkage operator

### Description
Shrinkage operator: S[x] = sgn(x) max(|x| - thr, 0). For description see section 5 of Candès, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal component analysis?

### Usage
thresh.l1(x, thr)

### Arguments
- **x**: a vector or a matrix.
- **thr**: threshold >= 0 to shrink with.

### Value
S[x] = sgn(x) max(|x| - thr, 0)
References


See Also

thresh.nuclear

Examples

```r
## The function is currently defined as
function(x, thr) {sign(x)*pmax(abs(x) - thr, 0)}

summary(thresh.l1(runif(100), 0.3))
```

thresh.nuclear  Thresholding operator

Description

Thresholding operator, an application of the shrinkage operator on a singular value decomposition: \( D[X] = U S[\Sigma] V \). For description see section 5 of Candès, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal component analysis?.

Usage

thresh.nuclear(M, thr)

Arguments

- `M` a rectangular matrix.
- `thr` threshold >= 0 to shrink singular values with.

Value

Returned is a thresholded Singular Value Decomposition with `thr` subtracted from singular values, and values smaller than 0 dropped together with their singular vectors.

- `u`, `d`, `vt` as in return value of `la.svd`
- `L` the resulting low-rank matrix: \( L = UDV^t \)
References

See Also
thresh.l1

Examples
## The function is currently defined as
function (M, thr) {
  s <- La.svd.cmp(M)
  dd <- thresh.l1(s$d, thr)
  id <- which(dd != 0)
  s$d <- dd[id]
  s$u <- s$u[, id, drop = FALSE]
  s$v <- s$v[, id, drop = FALSE]
  s$L <- s$u %*% (s$d * s$v)
  s
}

l <- thresh.nuclear(matrix(runif(600), nrow=20), 2)
l$d
Index

* Topic **Frobenius norm**
  > F2norm, 3
* Topic **low-rank and sparse components**
  > rpca, 4
* Topic **package**
  > rpca-package, 2
* Topic **robust pca**
  > rpca, 4
  > rpca-package, 2
* Topic **rpca**
  > rpca, 4
  > rpca-package, 2
* Topic **shrinkage operator**
  > thresh.l1, 6
* Topic **thresholding operator**
  > thresh.nuclear, 7

F2norm, 3

rpca, 2, 3, 4
rpca-package, 2

thresh.l1, 6, 8
thresh.nuclear, 7, 7