Package ‘rARPACK’

October 14, 2022

Type Package

Title Solvers for Large Scale Eigenvalue and SVD Problems

Version 0.11-0

Date 2016-03-07

Author Yixuan Qiu, Jiali Mei and authors of the ARPACK library. See file AUTHORS for details.

Maintainer Yixuan Qiu <yixuan.qiu@cos.name>

Description Previously an R wrapper of the 'ARPACK' library <http://www.caam.rice.edu/software/ARPACK/>, and now a shell of the R package 'RSpectra', an R interface to the 'Spectra' library <http://yixuan.cos.name/spectra/> for solving large scale eigenvalue/vector problems. The current version of 'rARPACK' simply imports and exports the functions provided by 'RSpectra'. New users of 'rARPACK' are advised to switch to the 'RSpectra' package.

License BSD_3_clause + file LICENSE

Copyright see file COPYRIGHTS

URL https://github.com/yixuan/rARPACK

BugReports https://github.com/yixuan/rARPACK/issues

Imports RSpectra

Suggests Matrix (>= 1.1-0)

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2016-03-10 00:06:16

R topics documented:

eigs *Find a Specified Number of Eigenvalues/vectors for Square Matrix*

Description

This function is a simple wrapper of the eigs() function in the **RSpectra** package. Also see the documentation there.

Given an n by n matrix A, function eigs() can calculate a limited number of eigenvalues and eigenvectors of A. Users can specify the selection criteria by argument which, e.g., choosing the k largest or smallest eigenvalues and the corresponding eigenvectors.

Currently eigs() supports matrices of the following classes:

matrix	The most commonly used matrix type, defined in **base** package.
dgeMatrix	General matrix, equivalent to matrix, defined in **Matrix** package.
dgCMatrix	Column oriented sparse matrix, defined in **Matrix** package.
dgRMatrix	Row oriented sparse matrix, defined in **Matrix** package.
dsyMatrix	Symmetrix matrix, defined in **Matrix** package.
function	Implicitly specify the matrix through a function that has the effect of calculating $f(x) = Ax$. See section **Funct**

eigs_sym() assumes the matrix is symmetric, and only the lower triangle (or upper triangle, which is controlled by the argument lower) is used for computation, which guarantees that the eigenvalues and eigenvectors are real, and in some cases reduces the workload. One exception is when A is a function, in which case the user is responsible for the symmetry of the operator.

eigs_sym() supports "matrix", "dgeMatrix", "dgCMatrix", "dgRMatrix" and "function" typed matrices.

Usage

```
eigs(A, k, which = "LM", sigma = NULL, opts = list(), ...)

eigs_sym(A, k, which = "LM", sigma = NULL, opts = list(),
   lower = TRUE, ...)
```

Arguments

A	The matrix whose eigenvalues/vectors are to be computed. It can also be a function which receives a vector x and calculates Ax. See section **Function Interface** for details.
k	Number of eigenvalues requested.
which	Selection criteria. See **Details** below.
sigma	Shift parameter. See section **Shift-And-Invert Mode**.
opts	Control parameters related to the computing algorithm. See **Details** below.
lower	For symmetric matrices, should the lower triangle or upper triangle be used.
...	Additional arguments such as n and args that are related to the Function Interface. See eigs() in the **RSpectra** package.

Details

The which argument is a character string that specifies the type of eigenvalues to be computed. Possible values are:

"LM"	The k eigenvalues with largest magnitude. Here the magnitude means the Euclidean norm of complex numbers.
"SM"	The k eigenvalues with smallest magnitude.
"LR"	The k eigenvalues with largest real part.
"SR"	The k eigenvalues with smallest real part.
"LI"	The k eigenvalues with largest imaginary part.
"SI"	The k eigenvalues with smallest imaginary part.
"LA"	The k largest (algebraic) eigenvalues, considering any negative sign.
"SA"	The k smallest (algebraic) eigenvalues, considering any negative sign.
"BE"	Compute k eigenvalues, half from each end of the spectrum. When k is odd, compute more from the high and then fr

eigs() with matrix type "matrix", "dgeMatrix", "dgCMatrix" and "dgRMatrix" can use "LM", "SM", "LR", "SR", "LI" and "SI".

eigs_sym(), and eigs() with matrix type "dsyMatrix" can use "LM", "SM", "LA", "SA" and "BE".

The opts argument is a list that can supply any of the following parameters:

ncv Number of Lanzcos basis vectors to use. More vectors will result in faster convergence, but with greater memory use. For general matrix, ncv must satisfy $k + 2 \leq ncv \leq n$, and for symmetric matrix, the constraint is $k < ncv \leq n$. Default is min(n, max(2*k+1, 20)).

tol Precision parameter. Default is 1e-10.

maxitr Maximum number of iterations. Default is 1000.

retvec Whether to compute eigenvectors. If FALSE, only calculate and return eigenvalues.

Value

A list of converged eigenvalues and eigenvectors.

values	Computed eigenvalues.
vectors	Computed eigenvectors. vectors[, j] corresponds to values[j].
nconv	Number of converged eigenvalues.
niter	Number of iterations used in the computation.
nops	Number of matrix operations used in the computation.

Shift-And-Invert Mode

The sigma argument is used in the shift-and-invert mode.

When sigma is not NULL, the selection criteria specified by argument which will apply to

$$\frac{1}{\lambda - \sigma}$$

where λ's are the eigenvalues of A. This mode is useful when user wants to find eigenvalues closest to a given number. For example, if $\sigma = 0$, then `which = "LM"` will select the largest values of $1/|\lambda|$, which turns out to select eigenvalues of A that have the smallest magnitude. The result of using `which = "LM", sigma = 0` will be the same as `which = "SM"`, but the former one is preferable in that ARPACK is good at finding large eigenvalues rather than small ones. More explanation of the shift-and-invert mode can be found in the SciPy document, http://docs.scipy.org/doc/scipy/reference/tutorial/arpack.html.

Function Interface

The matrix A can be specified through a function with the definition

```
function(x, args)
{
    ## should return A %*% x
}
```

which receives a vector `x` as an argument and returns a vector of the same length. The function should have the effect of calculating Ax, and extra arguments can be passed in through the `args` parameter. In `eigs()`, user should also provide the dimension of the implicit matrix through the argument `n`.

Author(s)

Yixuan Qiu http://statr.me

Jiali Mei <vermouthmjl@gmail.com>

See Also

eigen(), svd(), svds()

Examples

```
library(Matrix)
n = 20
k = 5

## general matrices have complex eigenvalues
set.seed(111)
A1 = matrix(rnorm(n^2), n)  ## class "matrix"
A2 = Matrix(A1)             ## class "dgeMatrix"

eigs(A1, k)
eigs(A2, k, opts = list(retvec = FALSE))  ## eigenvalues only

## sparse matrices
A1[sample(n^2, n^2 / 2)] = 0
A3 = as(A1, "dgCMatrix")
A4 = as(A1, "dgRMatrix")
```

```
eigs(A3, k)
eigs(A4, k)

## function interface
f = function(x, args)
{
    as.numeric(args %*% x)
}
eigs(f, k, n = n, args = A3)

## symmetric matrices have real eigenvalues
A5 = crossprod(A1)
eigs_sym(A5, k)

## find the smallest (in absolute value) k eigenvalues of A5
eigs_sym(A5, k, which = "SM")

## another way to do this: use the sigma argument
eigs_sym(A5, k, sigma = 0)

## The results should be the same,
## but the latter method is far more stable on large matrices
```

---

svds *Find the Largest k Singular Values/Vectors of a Matrix*

Description

This function is a simple wrapper of the `svds()` function in the **RSpectra** package. Also see the documentation there.

Given an m by n matrix A, function `svds()` can find its largest k singular values and the corresponding singular vectors. It is also called the Truncated Singular Value Decomposition since it only contains a subset of the whole singular triplets.

Currently `svds()` supports matrices of the following classes:

`matrix`	The most commonly used matrix type, defined in **base** package.
`dgeMatrix`	General matrix, equivalent to `matrix`, defined in **Matrix** package.
`dgCMatrix`	Column oriented sparse matrix, defined in **Matrix** package.
`dgRMatrix`	Row oriented sparse matrix, defined in **Matrix** package.
`dsyMatrix`	Symmetrix matrix, defined in **Matrix** package.

Note that when A is symmetric, SVD reduces to eigen decomposition, so you may consider using `eigs()` instead.

Usage

```
svds(A, k, nu = k, nv = k, opts = list(), ...)
```

Arguments

`A`	The matrix whose truncated SVD is to be computed.
`k`	Number of singular values requested.
`nu`	Number of left singular vectors to be computed. This must be between 0 and `k`.
`nv`	Number of right singular vectors to be computed. This must be between 0 and `k`.
`opts`	Control parameters related to the computing algorithm. See **Details** below.
`...`	Currently not used.

Details

The `opts` argument is a list that can supply any of the following parameters:

`ncv` Number of Lanzcos basis vectors to use. More vectors will result in faster convergence, but with greater memory use. `ncv` must be satisfy $k < ncv \leq p$ where `p = min(m, n)`. Default is `min(p, max(2*k+1, 20))`.

`tol` Precision parameter. Default is 1e-10.

`maxitr` Maximum number of iterations. Default is 1000.

Value

A list with the following components:

`d`	A vector of the computed singular values.
`u`	An `m` by `nu` matrix whose columns contain the left singular vectors. If `nu == 0`, `NULL` will be returned.
`v`	An `n` by `nv` matrix whose columns contain the right singular vectors. If `nv == 0`, `NULL` will be returned.
`nconv`	Number of converged singular values.
`niter`	Number of iterations used.
`nops`	Number of matrix-vector multiplications used.

Author(s)

Yixuan Qiu <`http://statr.me`>

See Also

`eigen()`, `svd()`, `eigs()`.

Examples

```
m = 100
n = 20
k = 5
set.seed(111)
A = matrix(rnorm(m * n), m)

svds(A, k)
svds(t(A), k, nu = 0, nv = 3)

## Sparse matrices
library(Matrix)
A[sample(m * n, m * n / 2)] = 0
Asp1 = as(A, "dgCMatrix")
Asp2 = as(A, "dgRMatrix")

svds(Asp1, k)
svds(Asp2, k, nu = 0, nv = 0)
```

Index