

Package ‘fftwtools’

March 25, 2017

Version 0.9-8

Title Wrapper for 'FFTW3' Includes: One-Dimensional Univariate,
One-Dimensional Multivariate, and Two-Dimensional Transform

Author Karim Rahim <karim.rahim@queensu.ca>

Maintainer Karim Rahim <karim.rahim@queensu.ca>

Depends R (>= 2.15.2)

SystemRequirements fftw3 (>= 3.1.2)

Suggests fftw

Description Provides a wrapper for several 'FFTW' functions. This package provides access to the two-dimensional 'FFT', the multivariate 'FFT', and the one-dimensional real to complex 'FFT' using the 'FFTW3' library. The package includes the functions `fftw()` and `mvfftw()` which are designed to mimic the functionality of the R functions `fft()` and `mvfft()`. The 'FFT' functions have a parameter that allows them to not return the redundant complex conjugate when the input is real data.

License GPL (>= 2)

ByteCompile true

URL <https://github.com/krahim/fftwtools>

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-03-25 16:24:52 UTC

R topics documented:

<code>fftw</code>	2
<code>fftw2d</code>	3
<code>mvfftw</code>	4
Index	6

fftw

*Compute fft using fftw3***Description**

These functions compute the FFT using the FFTW3 libraries. Use `fftw_r2c(x, HermConj=0)` for real to complex fft. This will return the result without the redundant complex conjugate. This follows the R convention for returning the unscaled inverse of the FFT. The function `fftw_c2r(res, HermConj=0, n=length(x))` will invert the FFT from the result not containing the redundant complex conjugate. You must specify, `n`, the dimension of the original data—length—if the redundant complex conjugate is not included.

Usage

```
fftw(data, inverse=0, HermConj=1, n=NULL)
fftw(data, inverse=0, HermConj=1, n=NULL)
fftw_r2c(data, HermConj=1)
fftw_c2c(data, inverse=0)
fftw_c2r(data, HermConj=1, n=NULL)
```

Arguments

<code>data</code>	(complex or real) vector to be processed
<code>inverse</code>	(integer) 1 or 0 indicating if inverse FFT is preformed. The return follows the format of the R FFT commands—the output is not scaled.
<code>HermConj</code>	(integer) 1 or 0 indicating if either "Hermitian" redundant conjugate should be returned, or that the complex to real data includes the "Hermitian" redundant conjugate.
<code>n</code>	(integer) column length of the original data set. This is required when using the inverse complex to real FFT without providing the "Hermitian" redundant conjugate.

Author(s)

Karim Rahim

Examples

```
res <- fftw_r2c(1:9)
res
fftw_c2r(res)/9
res
fftw_c2r(res)/9

res <- fftw_r2c(1:10)
res
fftw_c2r(res)/10
```

```

res
fftw_c2r(res)/10

res <- fftw_r2c(1:9, HermConj=0)
res
fftw_c2r(res, HermConj=0, n=9)/9

res <- fftw_r2c(1:10, HermConj=0)
res
fftw_c2r(res, HermConj=0, n=10)/10

fftw_r2c(1:3)
fftw_c2r(fftw_r2c(1:3))/3
fftw_c2r(fftw_r2c(1:2))/2
fftw_c2r(fftw_r2c(1:4))/4

fftw_r2c(1:3, HermConj=1)
fftw_c2r(fftw_r2c(1:3, HermConj=0), HermConj=0, n=3)/3

fftw_c2r(fftw_r2c(1:4, HermConj=0), HermConj=0, n=4)/4
fftw_c2r(fftw_r2c(1:20, HermConj=0), HermConj=0, n=20)/20

```

fftw2d

Compute a two-dimensional FFT on a matrix using FFTW3

Description

Computes two-dimensional FFT on a matrix using the FFTW3 libraries. Use `fftw_r2c_2d(x, HermConj=0)` for real to complex FFT. This will return the result without the "Hermitian" redundancy. These functions follow the R convention when returning the inverse of the FFT. For the two-dimension fft, the inverse is currently requires the entire matrix, including the redundant complex conjugate.

Usage

```

fftw2d(data, inverse=0, HermConj=1)
fftw_r2c_2d(data, HermConj=1)
fftw_c2c_2d(data, inverse=0)

```

Arguments

<code>data</code>	(complex or real) matrix to be processed
<code>inverse</code>	(integer) 1 or 0 indicating if inverse FFT is preformed. The return follows the format of the R FFT commands—the output is not scaled.
<code>HermConj</code>	(integer) 1 or 0 indicating if either "Hermitian" redundant conjugate should be returned.

Author(s)

Karim Rahim

Examples

```

x=c(1, 2, 3, 9, 8, 5, 1, 2, 9, 8, 7, 2)
x= t(matrix(x, nrow=4))
mvfftw(x)
t(mvfftw(t(mvfftw(x))))
fftw2d(x)
fftw2d(x, HermConj=0)

fftw2d(fftw2d(x), inverse=1)/12
fftw2d(fftw2d(t(x)), inverse=1)/12
fftw_r2c_2d(x)
fftw_r2c_2d(x, HermConj=0)

```

mvfftw

*Compute the FFT on each column of a matrix using FFTW3***Description**

This will compute the FFT of each column of a matrix using the FFTW3 libraries. Use `mvfftw_r2c(x, HermConj=0)` for real to complex fft. This will return the result without the redundant complex conjugate. This follows the R convention for returning the unscaled inverse of the FFT. The function `mvfftw_c2r(res, HermConj=0, n=dim(x)[1])` will invert the FFT from the result not containing the "Hermitian" redundant conjugate. You must specify, `n`, the column dimension of the original data—the column length of the original data—if the redundant complex conjugate is not included.

Usage

```

mvfftw(data, inverse=0, HermConj=1, n=NULL, fftplanopt=0)
mvfftw(data, inverse=0, HermConj=1, n=NULL, fftplanopt=0)
mvfftw_r2c(data, HermConj=1, fftplanopt=0)
mvfftw_c2c(data, inverse=0, fftplanopt=0)
mvfftw_c2r(data, HermConj=1, n=NULL, fftplanopt=0)

```

Arguments

<code>data</code>	(complex or real) matrix of columns to be processed
<code>inverse</code>	(integer) 1 or 0 indicating if inverse fft is preformed. The return follows the format of the R FFT commands. The result is not scaled.
<code>HermConj</code>	(integer) 1 or 0 indicating if either "Hermitian" redundant conjugate should be returned, or that the complex to real data includes the "Hermitian" redundant conjugate.
<code>n</code>	(integer) column length of the original data set, when using the inverse complex to real fft without providing the "Hermitian" redundant conjugate.

`fftplanopt` (integer) 0 or 1 specifying the flag passed to FFTW. 0 indicates the flag `FFTW_ESTIMATE` is used, and 1 indicates `FFTW_MEASURE` is used. See FFTW documentation for use of these flags.

Author(s)

Karim Rahim

Examples

```
x=c(1, 2, 3, 9, 8, 5, 1, 2, 9, 8, 7, 2)
x= t(matrix(x, nrow=4))
mvfft(x)
t(mvfft(t(mvfft(x))))

res <- mvfftw_r2c(x, HermConj=1)
res
mvfftw_c2c(res, inverse=1)/3
mvfftw_c2r(res)/3

res <- mvfftw_r2c(x, HermConj=0)
res
mvfftw_c2r(res, HermConj=0, n=3)/3

mvfftw_r2c(x, HermConj=1)
mvfft(x)
res <- mvfftw_r2c(x, HermConj=0)
res
mvfftw_c2r(res, HermConj=0, n=3)/3

res <- mvfftw_r2c(t(x), HermConj=1)
res
mvfftw_c2r(res, HermConj=1)/4
res <- mvfftw_r2c(t(x), HermConj=0)
res
mvfftw_c2r(res, HermConj=0, n=4)/4

mvfftw_r2c(t(x), HermConj=1)
mvfft(t(x))

mvfftw(mvfftw(x, HermConj=0), inverse=1, HermConj=0, n=3)/3
mvfftw(mvfftw(t(x), HermConj=0), inverse=1, HermConj=0, n=4)/4
mvfftw(mvfftw(t(x), inverse=1))/4
```

Index

*Topic **fftw**

- fftw, [2](#)
- fftw2d, [3](#)
- mvfftw, [4](#)

fftw, [2](#)

fftw2d, [3](#)

fftw_c2c (fftw), [2](#)

fftw_c2c_2d (fftw2d), [3](#)

fftw_c2r (fftw), [2](#)

fftw_r2c (fftw), [2](#)

fftw_r2c_2d (fftw2d), [3](#)

mvfftw, [4](#)

mvfftw_c2c (mvfftw), [4](#)

mvfftw_c2r (mvfftw), [4](#)

mvfftw_r2c (mvfftw), [4](#)