Package 'swdft'

October 14, 2022

Title Sliding Window Discrete Fourier Transform (SWDFT)

Version 1.0.0

Description Implements the Sliding Window Discrete Fourier Transform (SWDFT). Also provides statistical methods based on the SWDFT, and graphical tools to display the outputs.

Depends R (>= 3.3.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Suggests testthat, fftwtools, fields, signal, nloptr, knitr, rmarkdown, devtools

RoxygenNote 6.1.1

Imports stats, graphics

VignetteBuilder knitr

NeedsCompilation no

Author Lee F. Richardson [aut, cre]

Maintainer Lee F. Richardson <leerichardson2013@gmail.com>

Repository CRAN

Date/Publication 2019-04-17 04:22:46 UTC

R topics documented:

`coefficients.swdft_mod` *Coefficients method for swdft_cosreg objects*

Description

Coefficients method for swdft_cosreg objects

Usage

```
## S3 method for class 'swdft_mod'
coefficients(object, ...)
```

Arguments

object	A swdft_cosreg object
...	optional arguments to match generic function

---

complex_demod *Complex Demodulation*

Description

Complex Demodulation

Usage

```
complex_demod(x, f0, smooth = "butterworth", order = 5,
  passfreq = 0.1, match_swdft = FALSE, window_size = NULL)
```

Arguments

x	numeric vector
f0	numeric scalar. Frequency to demodulate
smooth	character. Type of smoothing to use, accepts either 'ma', 'double_ma', or 'butterworth' (the default)
order	moving average parameter if 'smooth' argument equals 'ma' or 'double_ma'. Defaults to 5
passfreq	numeric scalar. Pass frequency used in butterworth low-pass filter. Defaults to .1 which corresponds to a pass frequency of 2 * f0.
match_swdft	logical. Only used to demonstrate equivalence w/ SWDFT when a moving average filter is used. Otherwise, never used.
window_size	defaults to NULL, only used when match_swdft=TRUE, so can ignore.

Value

An S3 'swdft_demod' object. See ?new_swdft_matching_demod for details.

References

Chapter 7 of 'Fourier Analysis of Time-Series' by Peter Bloomfield and this blog post: https://dankelley.github.io/r/2014/02/1
for the idea of using a butterworth filter.

cosine *Cosine signal with adjustable parameters*

Description

Cosine signal with adjustable parameters

Usage

```
cosine(N, A = 1, Fr = 1, phase = 0)
```

Arguments

N	signal length
A	Amplitude
Fr	Frequency: Number of cycles in a length N period
phase	phase

Value

numeric vector with cosine function of x

cosine_taper *Cosine bell data taper*

Description

Cosine bell data taper

Usage

```
cosine_taper(n, p = 0.1)
```

Arguments

n	length of time-series to taper
p	proportion of ends to taper

Value

length n cosine bell taper w/ proportion p

cosreg *Cosine regression*

Description

Cosine regression

Usage

```
cosreg(x, f)
```

Arguments

x	numeric. Signal.
f	numeric. scalar or vector of frequencies to fit.

Value

S3 object of class 'swdft_cosreg'. See ?new_swdft_cosreg for details.

cov_swdft_cnum *Covariance between two complex-numbered outputs*

Description

Covariance between two complex-numbered outputs

Usage

```
cov_swdft_cnum(k, l, delta, n, sigma)
```

Arguments

k	frequency of first coefficient
l	frequency of second coefficient
delta	window position shift of second coefficient
n	window size
sigma	white noise standard error

Value

complex-valued number of the covariance

demod_swdft *Demodulate a Fourier Frequency with the SWDFT*

Description

Demodulate a Fourier Frequency with the SWDFT

Usage

```
demod_swdft(a, k)
```

Arguments

a	swdft
k	frequency to demodulate

dirichlet *Dirichlet Kernel (Weight) for arbitrary summation indices*

Description

Dirichlet Kernel (Weight) for arbitrary summation indices

Usage

```
dirichlet(x, phase = 0, a = 0, b = length(x) - 1)
```

Arguments

x	numeric to evaluate
phase	defaults to 0
a	start of summation index
b	end of summation index

Value

sum of a complex exponential sum

dirichlet_kernel *Dirichlet Kernel*

Description

Dirichlet Kernel

Usage

```
dirichlet_kernel(x, n, dw = FALSE)
```

Arguments

x	variable evaluated by dirichlet kernel
n	size of Dirichlet kernel
dw	logical whether to add the Dirichlet Weight (DW) factor

Value

evaluation of the Dirichlet Kernel (D_n(x))

fitted.swdft_mod *Fitted values method for swdft_cosreg objects*

Description

Fitted values method for swdft_cosreg objects

Usage

```
## S3 method for class 'swdft_mod'
fitted(object, ...)
```

Arguments

object	A swdft_cosreg object
...	optional arguments to match generic function

get_aphi *Extract amplitude and phase*

Description

Extract amplitude and phase

Usage

```
get_aphi(x, S, L, f)
```

Arguments

x	signal
S	start parameter
L	length pe
f	frequency

get_freq_range *Get range of frequencies to search*

Description

Get range of frequencies to search

Usage

```
get_freq_range(a, kwidth)
```

Arguments

a	2D complex-valued array. The SWDFT to search
kwidth	integer. the width of frequencies to search

get_loglik *Compute the log likelihood*

Description

Compute the log likelihood

Usage

```
get_loglik(x, fitted, sigma, N)
```

Arguments

x	signal
fitted	fitted values
sigma	estimated standard deviation
N	length of x

get_max_freq *Get the maximum DFT coefficient*

Description

Get the maximum DFT coefficient

Usage

```
get_max_freq(x)
```

Arguments

x	numeric vector

Value

numeric of largest frequency. Will be between 0 and .5

get_p_range *Get range of P's to search*

Description

Get range of P's to search

Usage

```
get_p_range(phat, n, N, pwidth, type = "around_max")
```

Arguments

phat	integer. Window position with largest SWDFT coefficient
n	integer. window size
N	integer. Signal length
pwidth	integer. the range of window positions to search for each window size
type	character. either 'around max' or 'fullp'.

get_sigma *Extract estimator of sigma*

Description

Extract estimator of sigma

Usage

```
get_sigma(x, fitted, N)
```

Arguments

x	signal
fitted	fitted values
N	length of x

get_sl *Extract signal parameters*

Description

Extract signal parameters

Usage

```
get_sl(n, p)
```

Arguments

n	window size
p	window position

get_taper *Create taper for the SWDFT*

Description

Create taper for the SWDFT

Usage

```
get_taper(n, taper, p)
```

Arguments

n	window size
taper	taper type. Can be either 'none' (default) or 'cosine'
p	proportion to taper on each end, if cosine taper is used

Value

length n taper

lcr_loglik *Log Likelihood*

Description

Log Likelihood

Usage

```
lcr_loglik(f, x, S, L, ftype = "full")
```

Arguments

f	frequency
x	signal
S	start parameter
L	length pe
ftype	what to return

local_cosreg *Local cosine regression*

Description

Local cosine regression

Usage

```
local_cosreg(x, lmin = 6, pwidth = 5, kwidth = 1, verbose = FALSE)
```

Arguments

x	numeric signal to apply local cosine regression on
lmin	integer. minimum signal length (L parameter) to search
pwidth	integer. the range of window positions to search for each window size
kwidth	integer. the width of frequencies to search
verbose	logical. whether or not to print intermediate results

Value

S3 object of class 'swdft_local_cosreg'

local_signal *Local Periodic Signal*

Description

Local Periodic Signal

Usage

```
local_signal(N, A = 1, Fr = 1, phase = 0, S = 0, L = N)
```

Arguments

N	signal length
A	Amplitude
Fr	Frequency: Number of cycles in a length N period
phase	phase
S	start of local signal
L	length of local signal

Value

length N local periodic signal

matching_demod *Matching Demodulation*

Description

Matching Demodulation

Usage

```
matching_demod(x, n, thresh = 0.05, max_cycles = 5,
  smooth = "butterworth", order = 5, passfreq = 0.1, debug = FALSE)
```

Arguments

x	numeric. Signal to demodulate
n	integer. Window size for SWDFT
thresh	numeric. Threshold to determine whether to continue demodulating
max_cycles	maximum number of demodulation cycles
smooth	character. Type of smoothing to use, accepts either 'ma', 'double_ma', or 'butterworth' (the default)
order	moving average parameter if 'smooth' argument equals 'ma' or 'double_ma'. Defaults to 5
passfreq	numeric scalar. Pass frequency used in butterworth low-pass filter. defaults to .1
debug	Logical. Whether to print out intermediate output.

Value

An S3 'swdft_matching_demod' object. See ?new_swdft_matching_demod for details.

---

moving_average *Simple high pass filter*

Description

Simple high pass filter

Usage

```
moving_average(x, order)
```

Arguments

x	the vector or time-series
order	the order of the filter

new_swdft *Constructor function for class 'swdft'*

Description

Constructor function for class 'swdft'

Usage

```
new_swdft(a, x, n, type, pad, taper_type, taper, p, smooth, m, num_convs)
```

Arguments

a	2D complex array of SWDFT coefficients. If there is smoothing, then this represents the smoothed squared modulus coefficients.
x	numeric input signal
n	window size
type	'fftw' or 'fft'
pad	whether or not it was padded
taper_type	type of taper
taper	numeric values of the taper
p	of cosine taper (if used)
smooth	type of smoother
m	width of kernel for smoothing (optional)
num_convs	number of kernel convolutions (optional)

Value

list w/ the same elements as the arguments, an S3 object of class 'swdft'

new_swdft2d *Constructor function for class 'swdft2d'*

Description

Constructor function for class 'swdft2d'

Usage

```
new_swdft2d(a, x, n0, n1, type)
```

Arguments

a	4D complex-valued array of 2D SWDFT coefficients
x	2D real or complex valued signal
n0	window size in row direction
n1	window size in column direction
type	algorithm to implement. defaults to "fftw", other option 'fft' for R's base FFT function. R's base fft function is used if

Value

S3 object w/ the same elements as arguments to this constructor function

---

new_swdft3d *Constructor function for class 'swdft3d'*

Description

Constructor function for class 'swdft3d'

Usage

```
new_swdft3d(a, x, n0, n1, n2, type)
```

Arguments

a	4D complex-valued array of 2D SWDFT coefficients
x	3D real or complex-valued array
n0	window size in dimension 0
n1	window size in dimension 1
n2	window size in dimension 2
type	defaults to 'base', which is the only option

Value

S3 object w/ the same elements as arguments to this constructor function

new_swdft_cosreg *Constructor function for class swdft_mod*

Description

Constructor function for class swdft_mod

Usage

```
new_swdft_cosreg(coefficients, fitted, residuals, data)
```

Arguments

coefficients	matrix of coefficients for cosine regression model
fitted	fitted values of cosine regression model
residuals	residuals of cosine regression model
data	original signal used to fit cosine regression

Value

list with the following elements

- coefficients. A matrix of parameters, the three columns are: 1. amplitude 2. phase, and 3. frequency. There is only more that one row used when multiple frequencies are fit sequentially.
- fitted. fitted values of cosine regression model
- residuals. residuals of cosine regression model
- data. original signal used to fit cosine regression

new_swdft_demod *Constructor function for class 'swdft_demod'*

Description

Constructor function for class 'swdft_demod'

Usage

```
new_swdft_demod(x, f0, A_t, Phi_t, fitted, y, y_smooth, smooth, order,
  passfreq)
```

Arguments

x	numeric vector
f0	numeric scalar. Frequency to demodulate
A_t	extracted amplitude from y_smooth
Phi_t	extracted phase from y_smooth
fitted	fitted values
y	non-smoothed demodulated signal
y_smooth	smoothed demodulated signal
smooth	character. Type of smoothing to use, accepts either 'ma', 'double_ma', or 'butterworth' (the default)
order	moving average parameter if 'smooth' argument equals 'ma' or 'double_ma'. Defaults to 5
passfreq	numeric frequency used as the passfreq in the low-pass filter

Value

list with the following elements

- coefficients. A matrix of parameters, the three columns are: 1. amplitude 2. phase, and 3. frequency. There is only more that one row used when multiple frequencies are fit sequentially.
- fitted. fitted values of cosine regression model
- residuals. residuals of cosine regression model
- data. original signal used to fit cosine regression
- list with the filter used ('smooth') and parameters ('order' for 'ma' or 'double_ma', 'passfreq' for butterworth)
- list w/ the demodulated signal, and smoothed demodulated signal

---

new_swdft_local_cosreg *Constructor function for class 'swdft_local_cosreg'*

Description

Constructor function for class 'swdft_local_cosreg'

Usage

```
new_swdft_local_cosreg(coefficients, fitted, residuals, data,
  window_params)
```

Arguments

coefficients	matrix of coefficients for cosine regression model
fitted	fitted values of cosine regression model
residuals	residuals of cosine regression model
data	original signal used to fit cosine regression
window_params	data frame of fitted coefficients for each window size

Value

list with the following elements

- coefficients. A matrix of parameters, the three columns are: 1. amplitude 2. phase, and 3. frequency. There is only more that one row used when multiple frequencies are fit sequentially.
- fitted. fitted values of cosine regression model
- residuals. residuals of cosine regression model
- data. original signal used to fit cosine regression
- window_params. data frame of fitted coefficients for each window size

---

new_swdft_matching_demod

Constructor function for class 'swdft_matching_demod'

Description

Constructor function for class 'swdft_matching_demod'

Usage

```
new_swdft_matching_demod(x, n, fitted, thresh, max_cycles, smooth, order,
  passfreqs, maxvals, freqs, khats, amps, phases, demods, cycle, resids,
  fits, return_rows)
```

Arguments

x	numeric. Signal to demodulate
n	integer. Window size for SWDFT
fitted	fitted values
thresh	numeric. Threshold to determine whether to continue demodulating
max_cycles	maximum number of demodulation cycles
smooth	character. Type of smoothing to use, accepts either 'ma', 'double_ma', or 'butterworth' (the default)
order	moving average parameter if 'smooth' argument equals 'ma' or 'double_ma'. Defaults to 5

passfreqs	pass frequency used in each iteration
maxvals	Maximum SWDFT coefficient for each iteration
freqs	Frequencies used in each iteration
khats	Integer version of frequency.
amps	Instantaneous amplitude for each iteration
phases	Instantaneous phase for each iteration
demods	List of demodulated signal and smoothed demodulated signal for each iteration
cycle	Number of cycles used
resids	Residuals for each iteration
fits	Fitted values for each iteration
return_rows	Logical vector indicating which iterations occurred. Used for subsetting.

Value

list with the following elements

- coefficients. coefficients from the R local signals with time-varying amplitude and phase model.
- fitted. fitted values of cosine regression model
- residuals. residuals of cosine regression model
- data. original signal used to fit cosine regression
- smooth. list with the filter used ('smooth') and parameters ('order' for 'ma' or 'double_ma', 'passfreq' for butterworth)
- demod. list w/ the demodulated signal, and smoothed demodulated signal
- thresh. Threshold used.
- iterations. List of fits, residuals, and maximum values for each iteration

---

plot.swdft *Plot method for 'swdft' object*

Description

Plot method for 'swdft' object

Usage

```
## S3 method for class 'swdft'
plot(x, freq_type = "cycles", fs = NULL,
  hertz_range = NULL, take_log = FALSE, log_thresh = 1e-05,
  use_fields = TRUE, scale_shrink = 0.9, zlim = NULL,
  xlab = "Window Position", ylab = "Frequency (Cycles/Window)",
  title = "SWDFT", cex_main = 1, cex_lab = 1, cex_axis = 1,
  xaxis_subset = NULL, custom_xaxis = NULL, custom_yaxis = NULL,
  col = "grayscale", display = TRUE, ...)
```

Arguments

x	Object of class 'swdft'. If x$a is complex-valued, it is converted to the squared modulus. If x$a is real-valued, then we assume that it represents the squared
freq_type	Specify how to display the frequency axis. Either 'cycles' (default), 'fraction', or 'hertz'
fs	sample rate. Used if freq_type='hertz'
hertz_range	integer vector, given by (low, high). Specifies the range of hertz to display and is only used when freq_type='hertz'
take_log	logical. Whether to take the log before plotting
log_thresh	numeric. Threshold for smallest possible value. Defaults to .000001, and is used to keep plots from displaying of ~ -40.
use_fields	logical. Determines whether we use image.plot from the fields package, or 'image' from the graphics package. The advantage of image.plot is that we get a color scale, so the default is TRUE
scale_shrink	Proportion between 0 and 1 to shrink the scale
zlim	Custom z range
xlab	Custom x-label
ylab	Custom y-label
title	Custom title
cex_main	how large to make the title
cex_lab	how large to make the labels
cex_axis	how large to make the axis labels
xaxis_subset	subset of x-axis (time / window position) for plotting
custom_xaxis	Defaults to NULL. Otherwise, used to change the x-axis
custom_yaxis	Defaults to NULL. Otherwise, used to change the y-axis
col	defaults to grayscale, can also be 'tim.colors' from fields package
display	logical. Defaults to TRUE, only used for testing purposes, so it should always be TRUE.
...	optional arguments to match the plot generic function

---

plot.swdft_mod *Plot method for swdft_mod object*

Description

Plot method for swdft_mod object

Usage

```
## S3 method for class 'swdft_mod'
plot(x, y = NULL, ...)
```

Arguments

x	A swdft_cosreg object
y	not used, but required by plot generic function
...	optional arguments to match the plot generic function

---

prou *The principal nth root of unity*

Description

The principal nth root of unity

Usage

```
prou(n)
```

Arguments

n	integer root

Value

complex number

---

residuals.swdft_mod *Residuals method for swdft_cosreg objects*

Description

Residuals method for swdft_cosreg objects

Usage

```
## S3 method for class 'swdft_mod'
residuals(object, ...)
```

Arguments

object	A swdft_cosreg object
...	optional arguments to match generic function

sine *Sine signal with adjustable parameters*

Description

Sine signal with adjustable parameters

Usage

```
sine(N, A = 1, Fr = 1, phase = 0)
```

Arguments

N	length signal
A	Amplitude
Fr	Frequency: Number of cycles in a length N period
phase	phase

Value

numeric vector with sine

smooth_pgram *Smooth SWDFT coefficients with a convolution*

Description

Smooth SWDFT coefficients with a convolution

Usage

```
smooth_pgram(a, fft_weight = NULL)
```

Arguments

a	real-valued length n periodogram
fft_weight	optionally specify the pre-computed FFT of the weights

Value

smoothed coefficients

smooth_swdft *Smooth the SWDFT coefficients*

Description

Smooth the SWDFT coefficients

Usage

```
smooth_swdft(a, ktype = "daniell", m = 2, num_convs = 1)
```

Arguments

a	real or complex-valued swdft. If real-valued, then we assume it's the squared modules already. If it's complex valued, we convert to the squared modulus.
ktype	either 'daniell' or 'modified.daniell'
m	kernel width from stats::kernel
num_convs	num_convs from stats::kernel

Value

Smooth squared modulues SWDFT coefficients

swdft *Sliding Window Discrete Fourier Transform*

Description

Sliding Window Discrete Fourier Transform

Usage

```
swdft(x, n, type = "fftw", pad = TRUE, taper_type = "none",
  p = 0.1, smooth = "none", m = 2, num_convs = 1)
```

Arguments

x	real or complex vector
n	integer window size.
type	algorithm to implement. defaults to "fftw", other option 'fft' for R's base FFT function. R's base fft function is used if
pad	optionally zero-pad the array to that the output array has the same dimension as the original time-series
taper_type	type of taper for each window position. defaults to 'none', can also be 'cosine'.

p	Proportion to be tapered at each end of the series. Argument copied from the spec.taper function in the default stats package. Defaults to .1.
smooth	Type of smoother. Defaults to 'none', can also be 'daniell' or 'modified daniell'. If smooth is 'none', then the SWDFT returns the smoothed squared modulus coefficients, not the complex numbers
m	width of kernel. Defaults to 2
num_convs	Number of times to convolve the kernel. Defaults to 1

Value

An S3 'swdft' object. See ?new_swdft for details.

Examples

```
x <- rnorm(n = 20)
a <- swdft(x, n = 2^3)
```

---

swdft2d *2D Sliding Window Discrete Fourier Transform*

Description

2D Sliding Window Discrete Fourier Transform

Usage

```
swdft2d(x, n0, n1, type = "fftw")
```

Arguments

x	2D input signal
n0	window size in row direction
n1	window size in column direction
type	algorithm to implement. defaults to "fftw", other option 'fft' for R's base FFT function. R's base fft function is used if 'fftwtools' library is not installed.

Value

An S3 'swdft2d' object. See ?new_swdft for details.

swdft2d_fft *2D Sliding Window Discrete Fourier Transform using base R*

Description

2D Sliding Window Discrete Fourier Transform using base R

Usage

```
swdft2d_fft(x, n0, n1)
```

Arguments

x	2D input signal
n0	window size in row direction
n1	window size in column direction

swdft2d_fftw *2D Sliding Window Discrete Fourier Transform using fftw*

Description

2D Sliding Window Discrete Fourier Transform using fftw

Usage

```
swdft2d_fftw(x, n0, n1)
```

Arguments

x	2D input signal
n0	window size in row direction
n1	window size in column direction

swdft3d *3D Sliding Window Discrete Fourier Transform*

Description

3D Sliding Window Discrete Fourier Transform

Usage

```
swdft3d(x, n0, n1, n2, type = "base")
```

Arguments

x	3D real or complex-valued array
n0	window size in dimension 0
n1	window size in dimension 1
n2	window size in dimension 2
type	defaults to 'base', which is the only option

Value

An S3 'swdft3d' object. See ?new_swdft for details.

swdft_base_3d *3D SWDFT using base R*

Description

3D SWDFT using base R

Usage

```
swdft_base_3d(x, n0, n1, n2)
```

Arguments

x	3D real or complex-valued array
n0	window size in dimension 0
n1	window size in dimension 1
n2	window size in dimension 2

swdft_fft *Sliding Window Discrete Fourier Transform with base R*

Description

Sliding Window Discrete Fourier Transform with base R

Usage

```
swdft_fft(x, n, taper)
```

Arguments

x	real or complex vector
n	integer window size.
taper	length n vector to multiply against the input data for each window position

Value

n x P array, where P = length(x) - n + 1

swdft_fftw *Sliding Window Discrete Fourier Transform using fftw*

Description

Sliding Window Discrete Fourier Transform using fftw

Usage

```
swdft_fftw(x, n, taper)
```

Arguments

x	real or complex vector
n	integer window size.
taper	length n vector to multiply against the input data for each window position

Value

n x P array, where P = length(x) - n + 1

swdft_to_props *Convert the SWDFT to proportions of frequency*

Description

Convert the SWDFT to proportions of frequency

Usage

```
swdft_to_props(a)
```

Arguments

a	swdft

unwrap_phase *Phase unwrapping*

Description

Phase unwrapping

Usage

```
unwrap_phase(p)
```

Arguments

p	vector of phases fit by demodulation

Index