psd

Adaptive, sine multitaper power spectral density estimation for R

by Andrew J Barbour, Jonathan Kennel, and Robert L Parker

Latest News

As of version 2.0, one can calculate the multivariate PSD ("cross spectrum") between two signals.

Description

This is an R package for computing univariate power spectral density estimates with little or no tuning effort. We employ sine multitapers, allowing the number to vary with frequency in order to reduce mean square error, the sum of squared bias and variance, at each point. The approximate criterion of Riedel and Sidorenko (1995) is modified to prevent runaway averaging that otherwise occurs when the curvature of the spectrum goes to zero. An iterative procedure refines the number of tapers employed at each frequency. The resultant power spectra possess significantly lower variances than those of traditional, non-adaptive estimators. The sine tapers also provide useful spectral leakage suppression. Resolution and uncertainty can be estimated from the number of degrees of freedom (twice the number of tapers).

This technique is particularly suited to long time series, because it demands only one numerical Fourier transform, and requires no costly additional computation of taper functions, like the Slepian functions. It also avoids the degradation of the low-frequency performance associated with record segmentation in Welch's method. Above all, the adaptive process relieves the user of the need to set a tuning parameter, such as time-bandwidth product or segment length, that fixes frequency resolution for the entire frequency interval; instead it provides frequency-dependent spectral resolution tailored to the shape of the spectrum itself.

`psd` elegantly handles spectra with large dynamic range and mixed-bandwidth features|features typically found in geophysical datasets.

How to Cite

Bob and Andy have a paper in Computers & Geosciences to accompany this software (download a pdf, 1MB); it describes the theory behind the estimation process, and how we apply it in practice. If you find **`psd`** useful in your research, we kindly request you cite our paper. See also:

```
citation("psd")
```

Getting Started

You can to install the package and it's dependencies with CRAN (from within the **`R`** environment):

```
install.packages("psd")
```

then load the package library

```
library(psd)
```

We have included a dataset to play with, namely **`Tohoku`**, which represents recordings of high-frequency borehole strainmeter data during teleseismic waves from the 2011 Mw 9.0 Tohoku earthquake (original data source). Access and inspect these data with:

```
data(Tohoku)
print(str(Tohoku))
```

The 'preseismic' data has interesting spectral features, so we subset it, and analyze the areal strain (the change in borehole diameter):

```
Dat <- subset(Tohoku, epoch=="preseismic")
Areal <- ts(Dat$areal)
```

For the purposes of improving the accuracy of the spectrum, we remove a linear trend:

```
Dat <- prewhiten(Areal, plot=FALSE)
```

Now we can calculate the adaptive PSD:

```
mtpsd <- pspectrum(Dat[['prew_lm']], plot=TRUE)
print(class(mtpsd))
```

In the previous example the **`plot=TRUE`** flag produces a comparison with a basic periodogram, but we can also visualize the spectrum with builtin plotting methods:

```
plot(mtpsd, log="dB")
```

The spectral uncertainty can be easily calculated:

```
sprop <- spectral_properties(mtpsd)
with(sprop, {
```

COPY LINK

https://rdocumentation.org/packages/psd/versio

VERSION

2.1.1

INSTALL

```
install.packages('psd')
```

MONTHLY DOWNLOADS

603

VERSION	LICENSE
2.1.1	GPL (>= 2)

ISSUES	PULL REQUESTS
2	0

STARS	FORKS
9	4

REPOSITORY

https://github.com/abarbour/psd

HOMEPAGE

Barbour and Parker (2014): https://doi.org/10.1016/j.cageo.2013.09.015

MAINTAINER	LAST PUBLISHED
Andrew Barbour	January 31st, 2022

```
    plot(taper/max(taper), type="h", ylim=c(0,2), col="dark grey")
    lines(stderr.chi.lower)

    lines(stderr.chi.upper)
})
```

Installing the Development Version

Should you wish to install the development version of this software, the remotes library will be useful:

```
library(remotes)
install_github("abarbour/psd")
```

Functions in psd (2.1.1)

Search all functions

as.tapers
Coerce an object into a 'tapers' object.

det_vector
det_vector

phase
phase

pilot_spec
Calculate initial power spectral density estimates

Tohoku
Observations of teleseismic strains from the 2011 Tohoku earthquake.

psdcore
Multitaper power spectral density estimates of a series

hfsnm
Noise levels found in PBO strainmeter data at seismic frequencies.

pspectrum
Adaptive sine multitaper power spectral density estimation

prewhiten
Prepare a series for spectral estimation

modulo_floor
Nearest value below

magnet
A single line of Project MAGNET horizontal field intensity

psd-environment
Various environment manipulation functions.

tapers-constraints
Taper constraint methods

spec-methods
Generic methods for objects with class 'spec'

spec_confint
Confidence intervals for multitaper power spectral density estimates

rcpp_ctap_simple
c++ implementation of the RLP constraint filter

riedsid
Constrained, optimal tapers using the Riedel & Sidorenko--Parker method

resample_fft_rcpp
Resample an fft using varying numbers of sine tapers

tapers-methods
Generic methods for objects with class 'tapers'

psd-normalization
Normalization of power spectral density estimates.

riedsid_rcpp
replaces time consuming portion of riedsid2

resample_mvfft
Resample an fft using varying numbers of sine tapers

tapers-refinement
Taper constraints using simple derivatives

spectral_properties
Calculate properties of multitaper power spectral density estimates

psd-package
Adaptive power spectral density estimation using optimal sine multitapers

splineGrad
Numerical derivatives of a series based on its smooth-spline representation

wipp30
Water levels from borehole WIPP30

parabolic_weights_rcpp
parabolic_weights_field

pgram_compare
Compare multitaper spectrum with cosine-tapered periodogram

ctap_loess
Taper constraints using loess smoothing

coherence
coherence

psd-utilities
Various utility functions.