amplitude
spectrum
phase spectrum

Normalization of Power Spectral Density

Andrew J. Barbour <andy.barbour@gmail.com> and Robert L. Parker

January 25, 2013

Abstract

Normalizations come in many forms, but here we outline the normalization used by `rlpSpec`: power spectral density.

Contents

1 Background

There can often be confusion about the different quantities used in spectral analysis [1], partly due to myriad nomenclature within the incredibly vast literature on the subject. Regarding nomenclature: Phrases including "amplitude spectrum", "energy spectral density", "power", "power spectra", and even "spectra" all mean *something*, but are rarely equivalent.

Let us, for the sake of brevity, assume we have a stationary signal $f(x)$ having Fourier Transform (FT) $F(s)$, which is complex. The of this signal is simply the real component of the FT, $\arg\{F\}$, which we'll denote as ${}^{(\mathtt{A})}S$. The corresponding is thus $\mathrm{mod}F$, or ${}^{(\phi)}S$. These quantities are, however, meaningless.

The latter assumes that the spectrum has the normalization used in power – which is that the Nyquist is assumed to be 1,0, always (power doesn't ask for a sample interval). But psd assumes that the interval is 1 (unless you specify otherwise) so that the Nyquist is 0.5. So say you have a white noise with variance 1: power will return a flat spectrum with level 0 db (=1) but psd will return a level of 2 (=3 db), both so that the level times the Nyquist will be 1. But logsmoo, told that the interval is 1, will multiply the spectrum by 2 to covert from lc**2/Nyquist to lc**2/Hz: giving 3 dB for the spectrum from power, but 6 dB (=4) for the spectrum from psd.

First load the package into the namespace:

```
> library(rlpSpec)
```

1.1 `stats::spectrum`

Included in the core distribution of R is `stats::spectrum`, which accesses `stats::spec.ar` or `stats::spec.pgram` for either parametric and non-parametric estimation, respectively. The user can optionally apply a single cosine taper, and/or a smoothing kernel. Our method is non-parametric; hence, we will compare to the latter.

[1] This post to `R-help` very eloquently describes the problem, and provides some guidance: `http://r.789695.n4.nabble.com/Re-How-do-I-normalize-a-PSD-td792902.html`

```
> spec.pgram(X, pad=1, taper=0.2, detrend=FALSE, demean=FALSE, plot=FALSE)
```

However, the logical arguments `detrend` and `demean` to `psdcore` are passed to `spec.pgram`; they are, by default, both TRUE.

As a matter of bookkeeping, we must deal with the working environment accessed by `rlpSpec` functions. Specifically, we should ensure `psdcore` does not access any inappropriate information by setting `refresh=TRUE`. We can then re-calculate the multitaper PSD and the raw periodogram with `plotpsd=TRUE`. The results are shown in Figure 1.1.

```
> data(magsat)
> psdcore(magsat$clean, ntaper=10, refresh=TRUE, plotpsd=TRUE)
```

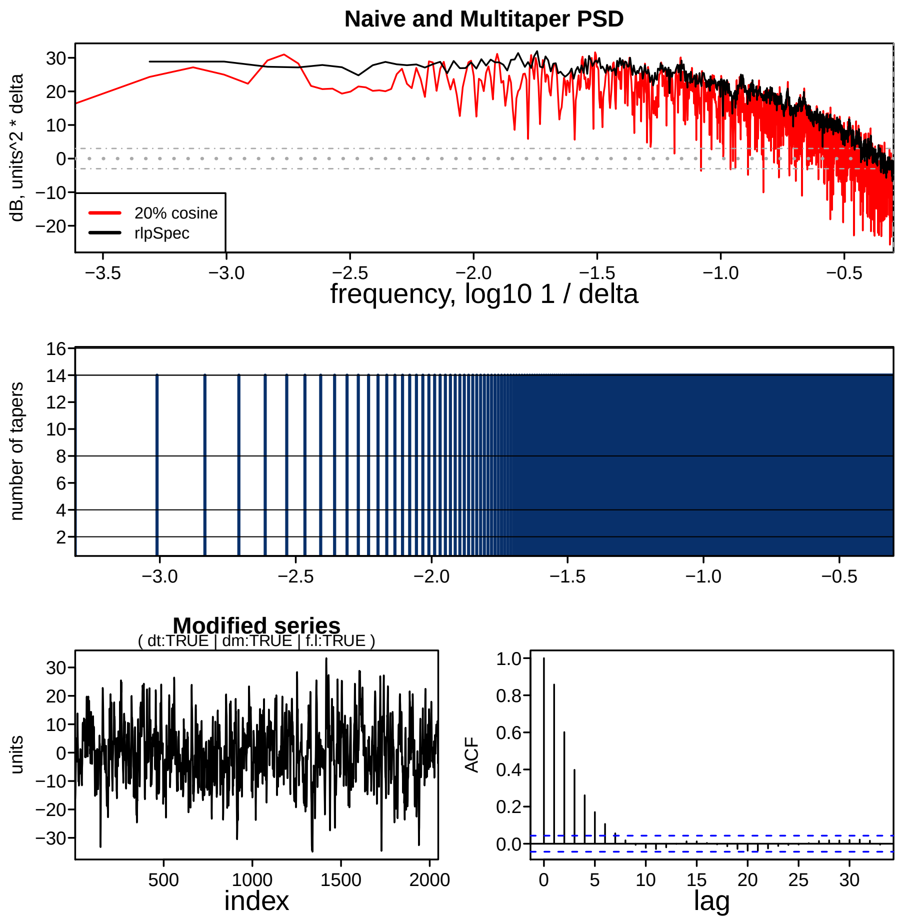

Figure 1: Top: Comparison between naïve and multitaper PSD estimators for the clean `MAGSAT` data. The frequency axis is in units of $\log_{10}$ km^{-1}, and power axis is in decibels. Bottom: The spatial series used to estimate the PSDs.

1.2 `multitaper::spec.mtm`

1.3 `SDF::sapa`

Index