`rlpSpec`: Adaptive sine multitaper power spectral density estimation

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

January 25, 2013

Abstract

The purpose of this vignette is to provide an overview of the features included in `rlpSpec`, which allow the user to compute sophisticated power spectral density (PSD) estimates for a univariate series, with very little tuning effort. The sine multitapers are used in which the number of tapers varies with spectral shape, according to the optimal value proposed by Riedel and Sidorenko [1995]. The adaptive procedure iteratively refines the optimal number of tapers at each frequency, which, assuming convergence, will produce spectra with significantly reduced variance (compared to naïve estimators), and minimum biasing effects. Resolution and uncertainty in a multitaper scheme are controlled by the number of tapers used. This means we do not need to resort to either windowing methods which inherently degrade resolution of low-frequency features (e.g. Welch), or smoothing kernels which can badly distort important features without careful tuning (e.g. Daniell, as in `stats::spec.pgram`). In this sense `rlpSpec` is best suited for data having spectra with both large dynamic range and strong, sharply changing features.

Contents

1 Quick start: A minimal example.

First load the package into the namespace:

```
> library(rlpSpec)
```

We now need a dataset to analyze. Among the datasets included in `rlpSpec` is a subset of the Magnetic Satellite (MAGSAT) mission [Langel et al., 1982]. Specifically, we have included along-track measurements of horizontal magnetic-field strength from a gimballed, airborne magnetometer, sampled once every kilometer, which means the spectrum may represent crustal magnetization with wavelengths longer than 2 km.

```
> data(magsat)
```

The format of the data set is a `data.frame` with four sets of information:

```
> names(magsat)

[1] "km"    "raw"   "clean" "mdiff"
```

The `raw` and `clean` names represent raw and edited intensities respectively, expressed in units of nanoTesla; `mdiff` is the difference between them. The difference between them is a matter of just a few points attributable to instrumental malfunction.

```
> subset(magsat, abs(mdiff)>0)

    km    raw   clean     mdiff
403  0  209.1 -3.6355 -212.7355
717  0 -248.7 -9.7775  238.9225
```

These deviations can, as we will see, adversely affect the accuracy of any PSD estimate, multitaper or otherwise.

Setting aside any discussion regarding sample stationarity, we can find power spectral density (PSD) estimates for the two series quite simply:

```
> psdr <- pspectrum(magsat$raw, verbose=FALSE)
> psdc <- pspectrum(magsat$clean, verbose=FALSE)
```

Each `pspectrum` command calculates a pilot PSD, followed by four iterations of refinement (the default). With each iteration the number of tapers is adjusted to the optimal number, based on the weighted spectral derivatives, following Riedel and Sidorenko [1995]. In general, spectral variance is reduced with sequential refinements, but is not necessarily quaranteed to converge. Note that in the example the sampling frequency of both series is km^{-1}, so we need not change the sampling rate argument.

Let's now visualize the two PSD estimates, recalling that the difference between the `raw` and `clean` samples is a mere two points. [1]

```
> plot(psdc, log="dB")
> lines(psdr$freq, dB(psdr$spec), col="red")
> legend("bottomleft",c("magsat$raw","magsat$clean"),col=c("red","black"),lty=1,lwd=2)
```

Figure 1: Comparison of power spectral densities for the `MAGSAT` data included with `rlpSpec`.

Figure 1 compares the spectra for the `raw` and `clean` samples. This plot shows a drastic improvement in shape between the two series, simply because the large outliers have been removed. The `clean` PSD shows the very red spectrum typical of geophysical processes [Agnew, 1992]. It also shows a rolloff in signal somewhere around the 20 kilometer wavelengths; whereas, the `raw` PSD looks highly unrealistic at higher wavelengths, and shows some curvature bias at low frequencies.

2 Comparisons with other methods

As we have shown in the MAGSAT example, improved understanding of the physics behind the signals in the data is of great concern. Assuming a sample is free of non-physical points, how do PSD estimates from `rlpSpec` compare with other methods? Unfortunately the suite of extensions with similar functionality is relatively limited; hopefully we have summarized the available functions in Table 1.

We now perform some tests to get a sense of how the results of `rlpSpec` compare with the methods in Table 1.

[1] Note that `pspectrum` returns an object with class `spec`, so we have access to methods within `stats`, including `plot.spec`.

Table 1: A comparison of functions comparable to `rlpSpec`, excluding raw periodogram estimators.

Function	Namespace	Sine m.t.?	Adaptive?	Reference
`mtapspec`	`RSEIS`	Yes	No	Lees and Park [1995]
`spectrum`	`stats`	No	No	R Core Team [2012]
`spec.mtm`	`multitaper`	Yes	Yes	Rahim and Burr [2012]
`SDF`	`sapa`	Yes	No	Percival and Walden [1993]

2.1 `RSEIS::mtapspec`

For this comparison we will use a dataset contained in `RSEIS` representing δO_{18} timeseries, namely `OH`.

```
> require(stats)
> require(RSEIS)
> data(OH)
> print(dt <- OH$dt[1]) # sampling interval, seconds

[1] 0.3

> a <- OH$JSTR[[1]]     # OH18 data
```

These data are shown in Figure 2.1.

For harmonic analysis we will follow, but modify slightly, the procedure outlined in the `RSEIS` vignette `rseis` [2] and calculate the multitaper spectrum for the following parameters:

```
> Mspec <- mtapspec(a, dt, MTP=list(kind=1, nwin=5, npi=3, inorm = 3))
> str(Mspec)

List of 12
 $ dat     : num [1:866] -2.22 -1.16 -1 -0.69 -0.56 -0.46 -0.73 -0.84 -0.97 -1.12 ...
 $ dt      : num 0.3
 $ spec    : num [1:2048] 18.3 17.7 18 17.9 17.2 ...
 $ dof     : num [1:2048] 4 4 4 4 4 4 4 4 4 4 ...
 $ Fv      : num [1:2048] 1451.689 5.816 0.18 0.162 0.205 ...
 $ Rspec   : num [1:1025, 1:5] -893 -190 559 255 -122 ...
 $ Ispec   : num [1:1025, 1:5] 0 -796 -283 302 161 ...
 $ freq    : num [1:1025] 0 0.00163 0.00326 0.00488 0.00651 ...
 $ df      : num 0.00163
 $ numfreqs: num 1025
 $ klen    : num 2048
 $ mtm     :List of 4
  ..$ kind : num 1
  ..$ nwin : num 5
  ..$ npi  : num 3
  ..$ inorm: num 3
```

where `nwin` is the number of tapers taken, and `inorm` is he normalization flag. Digging a bit into . We obviously need to deal with the sampling information since it is not one (as assumed by default). The information we have is that the measurements are taken every 0.3 seconds. We can use account for the sampling interval (as opposed to sampling rate) easily by using a negative value for `X.frq`, which `psdcore` will interpret as an interval instead of a frequency. An example highlights this:

[2] `http://cran.r-project.org/web/packages/RSEIS/vignettes/rseis.pdf`

```
> ats <- prewhiten(ts(a, deltat=dt)) # remove an Ax+B model by default
```

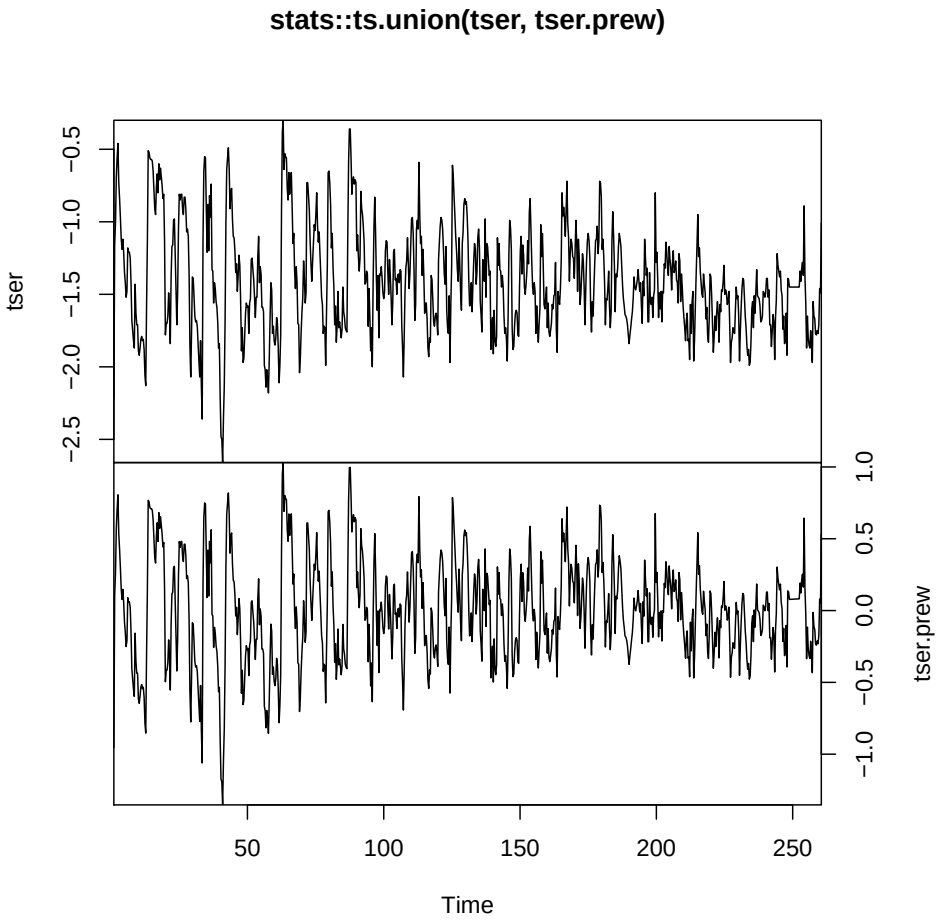


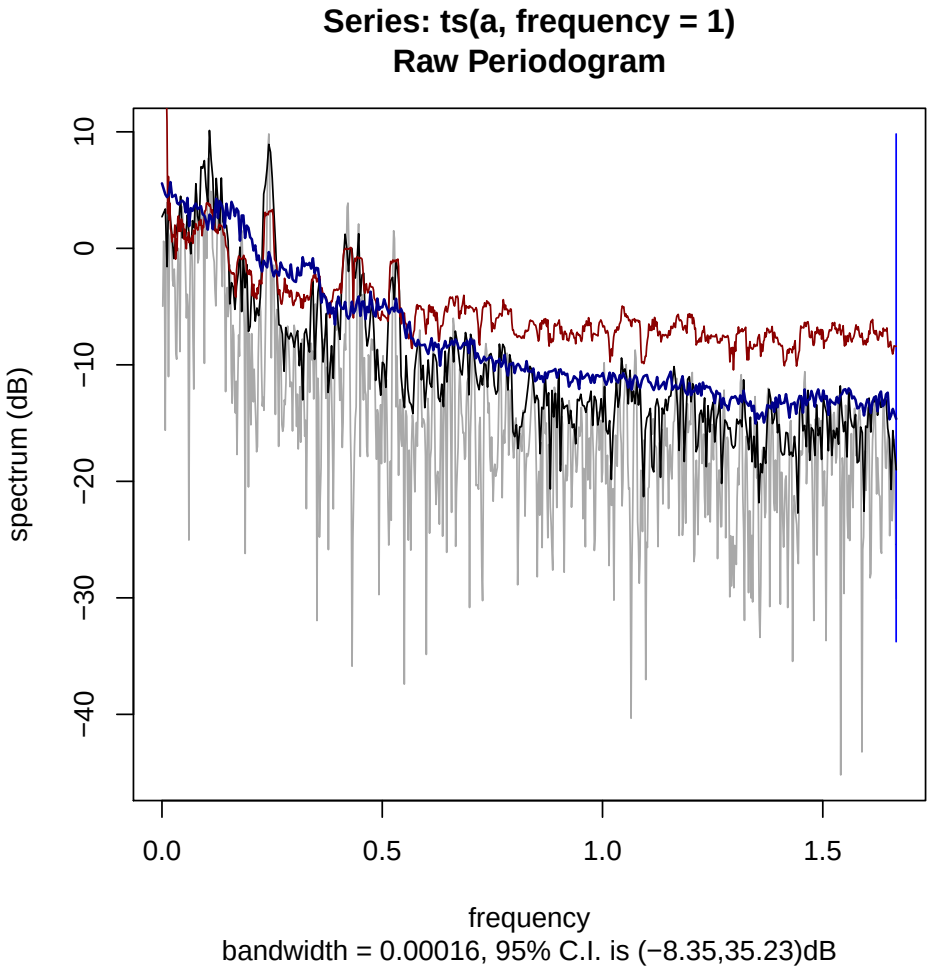
Figure 2: Pre-processed δO_{18} timeseries from the `RSEIS` package.

```
> all.equal(psdcore(a,1)$spec, psdcore(a,-1)$spec)

[1] TRUE
```

Returning to the δO_{18} data, we can now calculate the spectrum:

```
> Aspec <- pspectrum(a, -dt, 5, niter=1, plot=FALSE)
> Pspec <- psdcore(a, -dt, 5)
> pltf <- Mspec$freq
> nt <- 1:Mspec$numfreqs
> # multiply by Nyquist
> nyq <- frequency(ats)/2
> pltp <- dB(Mspec$spec[nt] * nyq)
> Xspec <- spec.pgram(ts(a, frequency=1), pad=1, taper=0.2, detrend=TRUE, demean=TRUE, plot=FALSE)
> Xspec$freq <- Xspec$freq * 2 * nyq
> Xspec$spec <- nyq * Xspec$spec
> plot(Xspec, log="dB", col="dark grey")
> plot(Pspec, log="dB", add=TRUE)
> lines(pltf, pltp, col="dark red")
> plot(Aspec, log="dB", add=TRUE, col="dark blue", lwd=1.5)
```



This is a situation where the adaptive method can be overly aggressive in taper application. Remember, it all depends on the spectral shape.

and enhance instead with an adaptive procedure:

2.2 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.

Included in `rlpSpec` is an option to compare the results with a naïve estimator–a 20% tapered periodogram– from within the spectrum calculator, `psdcore`. In R this estimator is equivalent to running:

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

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 2.2.

```
> ntap <- psdc$taper
> psdcore(magsat$clean, ntaper=ntap, refresh=TRUE, plotpsd=TRUE)
```

Figure 3: 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.

2.3 multitaper::spec.mtm

Adaptive sine multitaper option has nearly identical functionality: it calls source code of a Fortran equivalent to rlp, also written by R.L. Parker. By default it uses the Discrete Prolate Spheroidal Sequences (dpss) of Thomson [1982], so there can be many more knobs to turn.

Among the other functions included in `multitaper` is `spec.mtm`.

In fact, this function calls portions of the Fortran equivalent of `rlpSpec`

2.4 SDF::sapa

As of this writing, the package has no maintainer; lest we discuss deprecated and archived functions, we will not review its capabilities.

3 Assessing spectral properties

In a multitaper scheme, the computation of resolution and uncertainty (shown as blue lines in Figure 1 depends on the the number of tapers; hence, the methods internal to `plot.spec` are not appropriate.

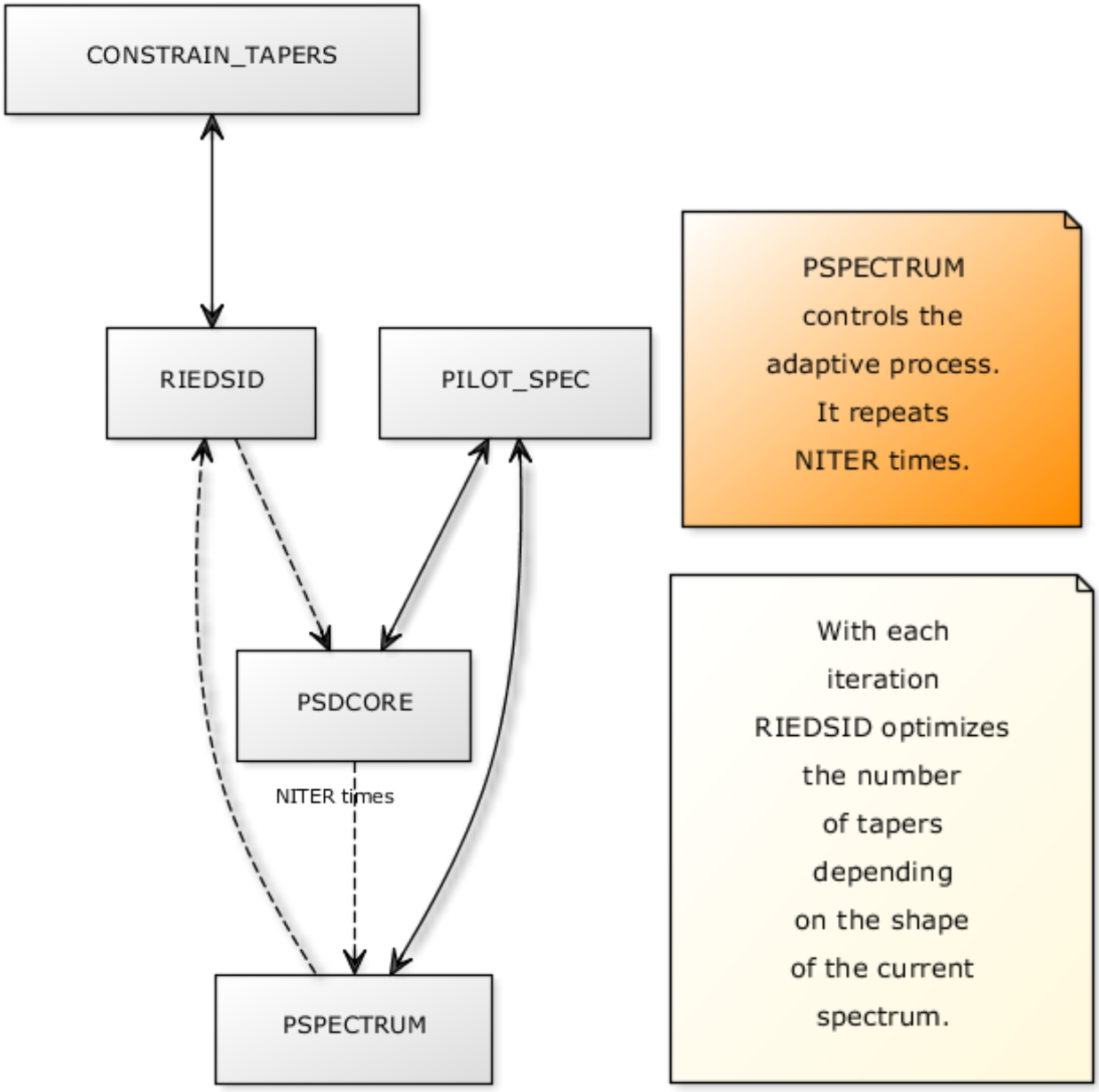


Figure 4: Simplified call graph for `rlpSpec`. The dashed lines show the circuit in which the spectra and its tapers make during the iterative process.

4 Call overview

References

D. C. Agnew. The time-domain behavior of power-law noises. *Geophysical Research Letters*, 19:333–336, 1992.

R. Langel, G. Ousley, J. Berbert, J. Murphy, and M. Settle. The MAGSAT mission. *Geophysical Research Letters*, 9(4):243–245, 1982.

J. M. Lees and J. Park. Multiple-taper spectral analysis: A stand-alone C-subroutine. *Computers & Geosciences*, 21(2):199–236, 1995.

D.B. Percival and A.T. Walden. *Spectral analysis for physical applications*. Cambridge University Press, 1993.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. URL `http://www.R-project.org/`. ISBN 3-900051-07-0.

K. Rahim and W. Burr. *multitaper: Multitaper Spectral Analysis*, 2012. URL `http://CRAN.R-project.org/package=multitaper`. R package version 1.0-2.

K. S. Riedel and A. Sidorenko. Minimum bias multiple taper spectral estimation. *IEEE Trans. SP*, 43(1): 188–195, 1995.

D. J. Thomson. Spectrum estimation and harmonic analysis. *Proceedings of the IEEE*, 70(9):1055–1096, 1982.