`rlpSpec`: Adaptive sine multitaper power spectral density estimation

Andrew J. Barbour and Robert L. Parker

February 4, 2013

Abstract

The purpose of this vignette is to provide an overview of some of the features included in `rlpSpec`, designed to compute estimates of power spectral density (PSD) for a univariate series in a sophisticated manner, 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 based on the spectra from the previous iteration. Assuming the adaptive procedure is convergent, this method produces power spectra with significantly lower spectral variance relative to results from naïve estimators. Sine tapers exhibit excellent leakage suppression characteristics, so bias effects are also reduced. Resolution and uncertainty vary with the number of tapers, which means we do not need to resort to either (1) windowing methods, which inherently degrade resolution at low-frequency (e.g. Welch's method); or (2) smoothing kernels, which can badly distort important features without careful tuning (e.g. the Daniell kernel in `stats::spectrum`). In this regards `rlpSpec` is best suited for data having a mix of large dynamic range and strong features, such as power spectra of ambient seismometer records.

Contents

1 Quick start: A minimal example.

First, we load the package into the namespace:

```
> library(rlpSpec)
```

We need a series 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 gimbaled, airborne magnetometer. The sampling interval is once every kilometer (km), so the will 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$raw)
> psdc <- pspectrum(magsat$clean)
```

Each `pspectrum` command calculates a pilot PSD, followed by `niter` iterations of refinement. With each iteration the number of tapers is adjusted based the proposed optimal number from Riedel and Sidorenko (1995), which depends on spectral shape; we use quadratically weighted spectral derivatives to estimate this shape.

In general, spectral variance is reduced with sequential refinements[1], but is not necessarily guaranteed to converge. Note that in the example the sampling frequency of both series is 1 km^{-1}, the assumed value.

Figure 1 compares the spectra for the `raw` and `clean` samples[2]. We expect the MAGSAT data to be linear in the space of linear-frequencies and logarithmic-power; we see a clear improvement in spectral shape between the two series, simply because the large outliers have been removed. The PSD of the clean series shows a very red spectrum typical of geophysical processes (Agnew, 1992), and a rolloff in signal for 10 kilometer wavelengths and longer; whereas, the PSD for the raw series looks somewhat unrealistic at higher wavelengths.

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

```
> plot(psdc, log="dB", main="Raw and Clean MAGSAT power spectral density",
+        lwd=3, ci.col=NA, ylim=c(0,32), yaxs="i")
> plot(psdr, log="dB", add=TRUE, lwd=3, lty=5)
> text(c(0.25,0.34), c(11,24), c("Clean","Raw"), cex=1)
```

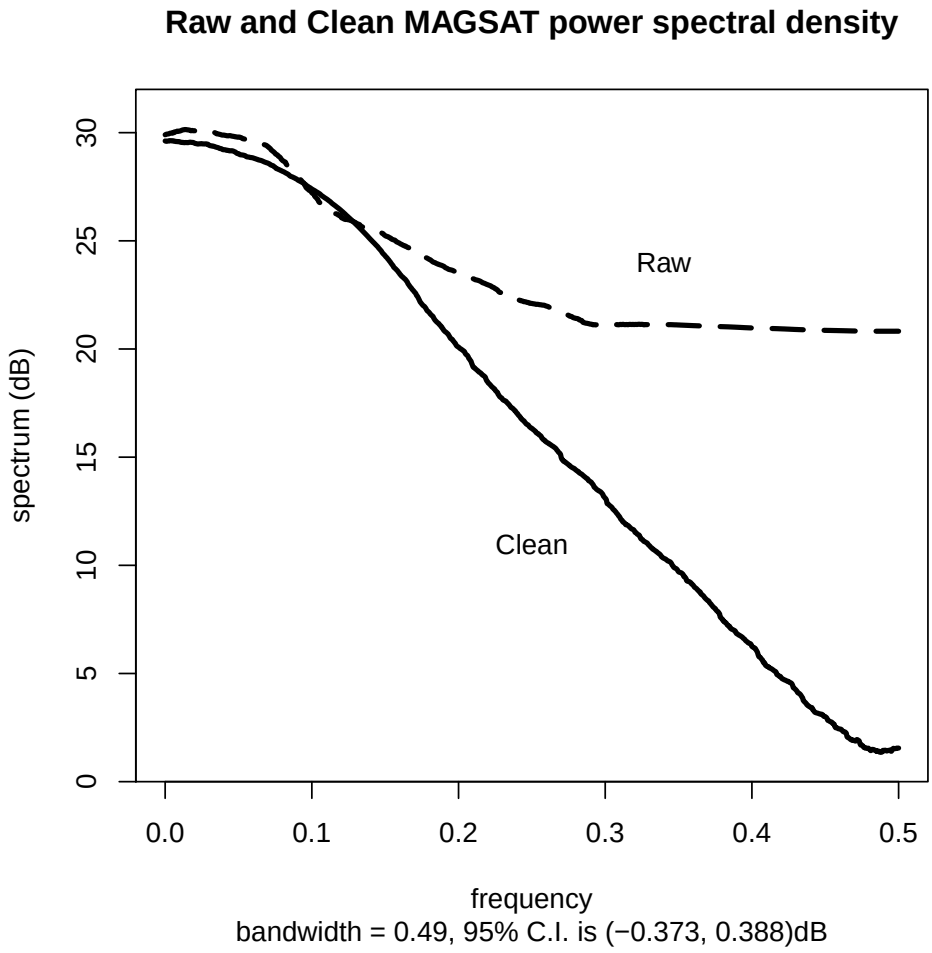


Figure 1: Power spectral density estimates for the raw and cleaned `MAGSAT` data bundled with `rlpSpec`. Note that because the class is `'spec'` we have utilized existing methods in the `stats` namespace.

Table 1: A comparison of power spectral density estimators in R, excluding extensions which only estimate raw-periodograms. Normalizations are shown as either "single" or "double" for either single- or double-sided spectra, and "various" if there are multiple, optional normalizations. A (∗) denotes the default for a function having an option for either single or double.

Function	Namespace	Sine m.t.?	Adaptive?	Norm.	Reference
`mtapspec`	`RSEIS`	Yes	No	various	Lees and Park (1995)
`pspectrum`	`rlpSpec`	Yes	Yes	single	Parker and Barbour (2013)
`spectrum`	`stats`	No	No	double	R Core Team (2012)
`spec.mtm`	`multitaper`	Yes	Yes	double	Rahim and Burr (2012)
`SDF`	`sapa`	Yes	No	single*	Percival and Walden (1993)

rlpSpec compare with other methods? Unfortunately the suite of extensions with similar functionality is relatively limited, but hopefully we have summarized most, if not all, the available functions in Table 1.

We compare results from rlpSpec with those from a few of the methods in Table 1, using the same data: the cleaned MAGSAT series.

2.1 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.

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 may be found with the following settings:

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

However, within psdcore the comparison is made with the logical arguments detrend and demean passed to spec.pgram; they are, by default, both TRUE.

As a matter of bookkeeping and good practice, we should consider the working environment accessed by rlpSpec functions. To ensure psdcore does not access any inappropriate information leftover from the previous calculations, we can set refresh=TRUE. We can then re-calculate the multitaper PSD and the raw periodogram with plotpsd=TRUE; these results are shown in Figure 2.

2.2 RSEIS::mtapspec

In RSEIS the spectrum estimation tool is mtapspec, which calls the program of Lees and Park (1995). There are numerous optional tuning parameters, including flags for normalization and taper averaging. For our purpose the correct normalization for mtapspec is found by using MTP=list(kind=2, inorm=3) and scaling the results by 2 (to convert double-sided spectra to single-sided spectra).

We assume mtapspec doesn't remove a mean and trend from the input series. We can do this easily with the prewhiten methods[3].

```
> require(RSEIS)
> dt=1 # km
> ats <- prewhiten(ts(magsat$clean, deltat=dt), plot=FALSE)
```

and if we set AR.max higher than zero, the program would've fit an auto-regressive (AR) model to the data. In Figure 3 we show the AR fitting method, and note that while we set AR.max relatively high, only and AR(6) model was fit significantly.

We didn't necessarily need to deal with the sampling information since it is just 1 per km; but, supposing the sampling information was based on an interval, we could have used a negative value for X.frq, with which psdcore would interpret as an interval (instead of a frequency). A quick example highlights the equivalency:

```
> a <- rnorm(32)
> all.equal(psdcore(a,1)$spec, psdcore(a,-1)$spec)

[1] TRUE
```

Returning the the RSEIS comparison, we first estimate the PSD from mtapspec with 10 tapers:

[1] Messages given by pspectrum with "Ave. S.V.R." (not shown) are in reference to the average spectral-variance reduction, found from double-differenced spectra at each stage, relative to the pilot estimate.

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

[3] Although the name implies parametric fitting, the default operation of prewhiten is to assume a linear model of the form $f(x) = \alpha x + \beta + \epsilon$ and fit using ordinary linear least squares.

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

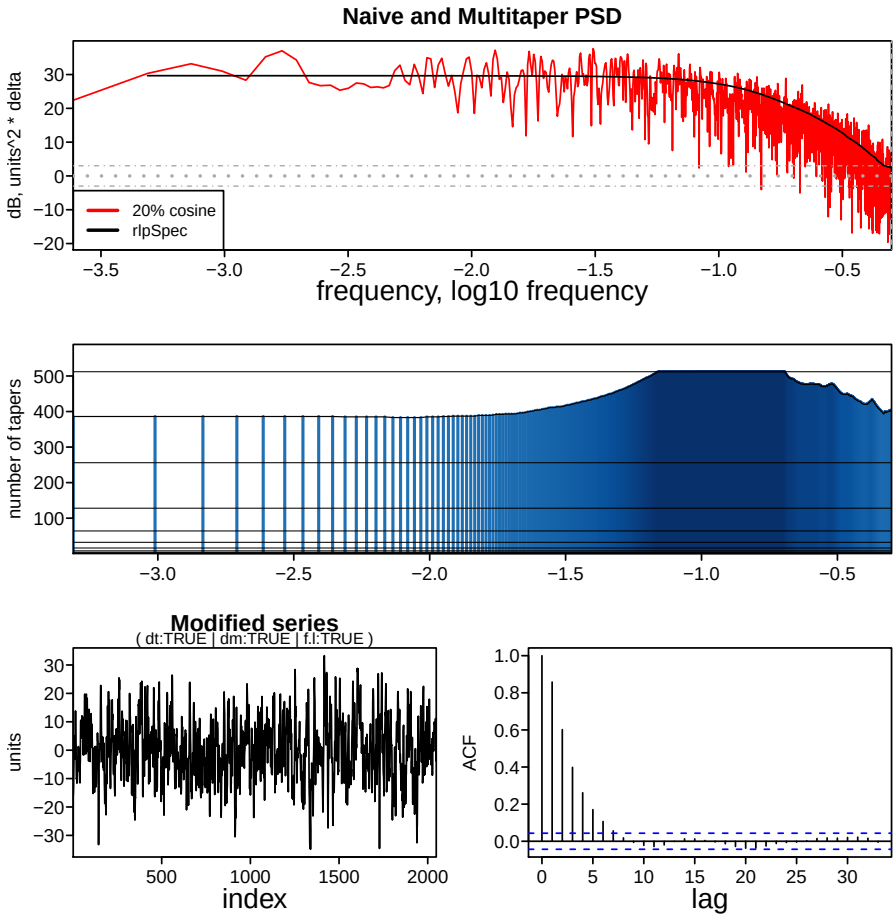


Figure 2: Top: Comparison between naïve and multitaper PSD estimators for the clean `MAGSAT` data. The frequency axis is in units of $\log_{10}$ km^{-1}, and power axis is in decibels. Middle: The number of tapers applied as a function of frequency from the `plot.tapers` method. Bottom: The spatial series used to estimate the PSDs and a subset of the full autocorrelation function.

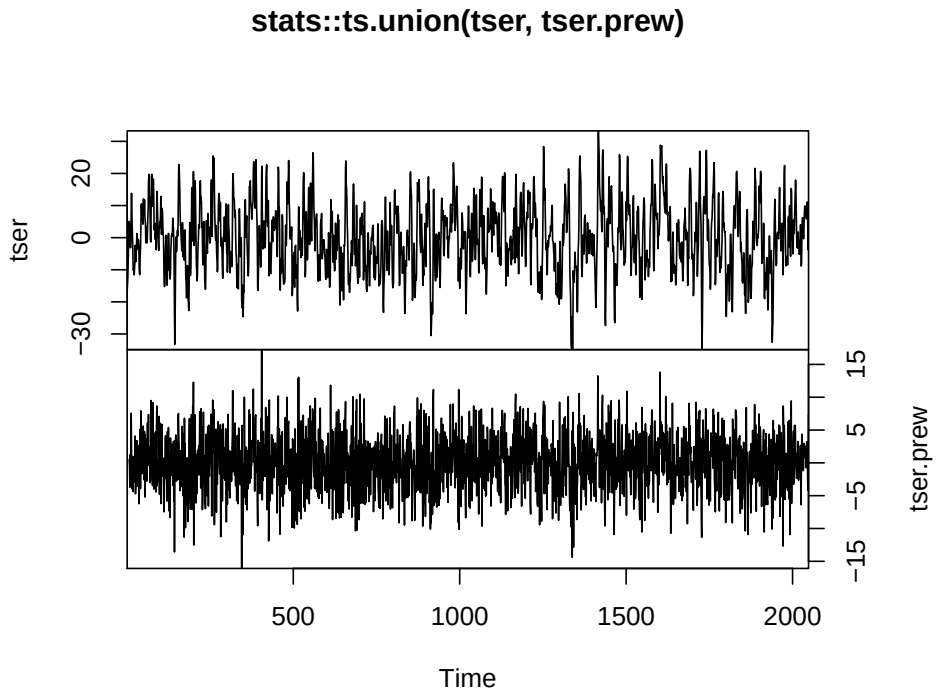
```
> tapinit <- 10
> Mspec <- mtapspec(ats, deltat(ats), MTP=list(kind=2, inorm=3, nwin=tapinit, npi=0))
```

where `nwin` is the number of tapers taken and `npi` is, from the documentation, the "number of Pi-prolate functions" (we leave it out for the sake of comparison). Note that the object returned is not of class `spec`:

```
> str(Mspec)

List of 12
 $ dat     : ts [1:2048, 1] -16.23 -14.56 -12.02 -7.21 -3.13 ...
  ..- attr(*, "dimnames")=List of 2
```

```
> atsar <- prewhiten(ats, AR.max=100, verbose=FALSE)
```




```
> plot(psdcore(atsar,ntaper=10), log="dB", main="PSD of MAGSAT innovations")
```

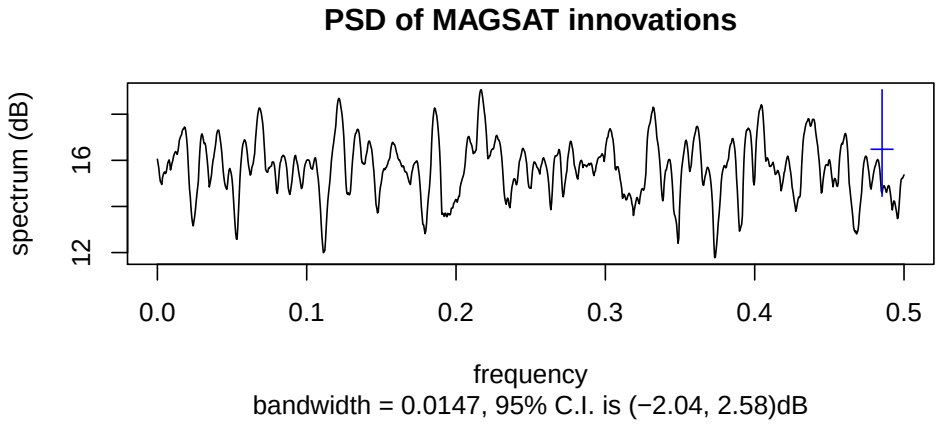


Figure 3: Top: Pre-whitening of a series assuming an AR model. Bottom: Power spectral density estimates of the innovations (model residuals): the spectrum is relatively flat, as expected.

```
  .. ..$ : NULL
  .. ..$ : NULL
  ..- attr(*, "tsp")= num [1:3] 1 2048 1
 $ dt      : num 1
 $ spec    : num [1:4096] 528 557 600 595 615 ...
 $ dof     : num [1:4096] 20 20 20 20 20 20 20 20 20 20 ...
 $ Fv      : num [1:4096] 4.45e-20 4.78e-02 5.36e-01 1.54 1.15 ...
 $ Rspec   : num [1:2049, 1:10] 1.86e-07 -9.32e+01 6.05e+02 1.16e+03 -2.97e+02 ...
 $ Ispec   : num [1:2049, 1:10] 0 -227 -569 665 1157 ...
 $ freq    : num [1:2049] 0 0.000244 0.000488 0.000732 0.000977 ...
 $ df      : num 0.000244
 $ numfreqs: num 2049
 $ klen    : num 4096
 $ mtm     :List of 4
  ..$ kind : num 2
  ..$ nwin : num 10
  ..$ npi  : num 0
  ..$ inorm: num 3
```

We will calculate the comparative spectra from

1. `spectrum` (20% cosine taper),
2. `psdcore` (with fixed tapers), and
3. `pspectrum` (allowing adaptive taper refinement)

and we will need to correct for normalization factors, as necessary, with `normalize`. Note that by default the normalization is set within `pspectrum` (with `normalize`) once the adaptive procedure is finished.

```
> Xspec <- spec.pgram(ats, pad=1, taper=0.2, detr=TRUE, dem=TRUE, plot=FALSE)
> Pspec <- psdcore(ats, dt, tapinit)
> Aspec <- pspectrum(ats, dt, tapinit, niter=3)
> # Correct for double-sidedness of spectrum and mtapspec results
> class(Mspec)

[1] "list"

> Mspec <- normalize(Mspec, dt, "spectrum")
> nt <- 1:Mspec$numfreqs
> mspec <- Mspec$spec[nt]
> class(Xspec)

[1] "spec"

> Xspec <- normalize(Xspec, dt, "spectrum")
```

These estimates are shown on the same scale in Figure 4.

Because we did not specify the length of the FFT in `mtapspec` we end up with different length spectra. So, to form some statistical measure of the results, we can interpolate PSD levels onto the `rlpSpec`-based frequencies (or reciprocally):

```
> require(signal)
> pltpi <- interp1(pltf, pltp, Pspec$freq)
```

```
> require(RColorBrewer)
> cols <- c("dark grey", brewer.pal(8, "Set1")[c(5:4,2)])
> lwds <- c(1,2,2,5)
> plot(Xspec, log="dB", ylim=40*c(-0.4,1), ci.col=NA,
+        col=cols[1], lwd=lwds[1], main="PSD comparisons")
> pltf <- Mspec$freq
> lines(pltf, pltp <- dB(mspec), col=cols[2], lwd=lwds[2])
> plot(Pspec, log="dB",  add=TRUE, col=cols[3], lwd=lwds[3])
> plot(Aspec, log="dB", add=TRUE, col=cols[4], lwd=lwds[4])
> legend("topright",
+             c("spec.pgram","RSEIS::mtapspec","psdcore","pspectrum"),
+             title="Estimator", lwd=3, cex=1.1, col=cols)
```

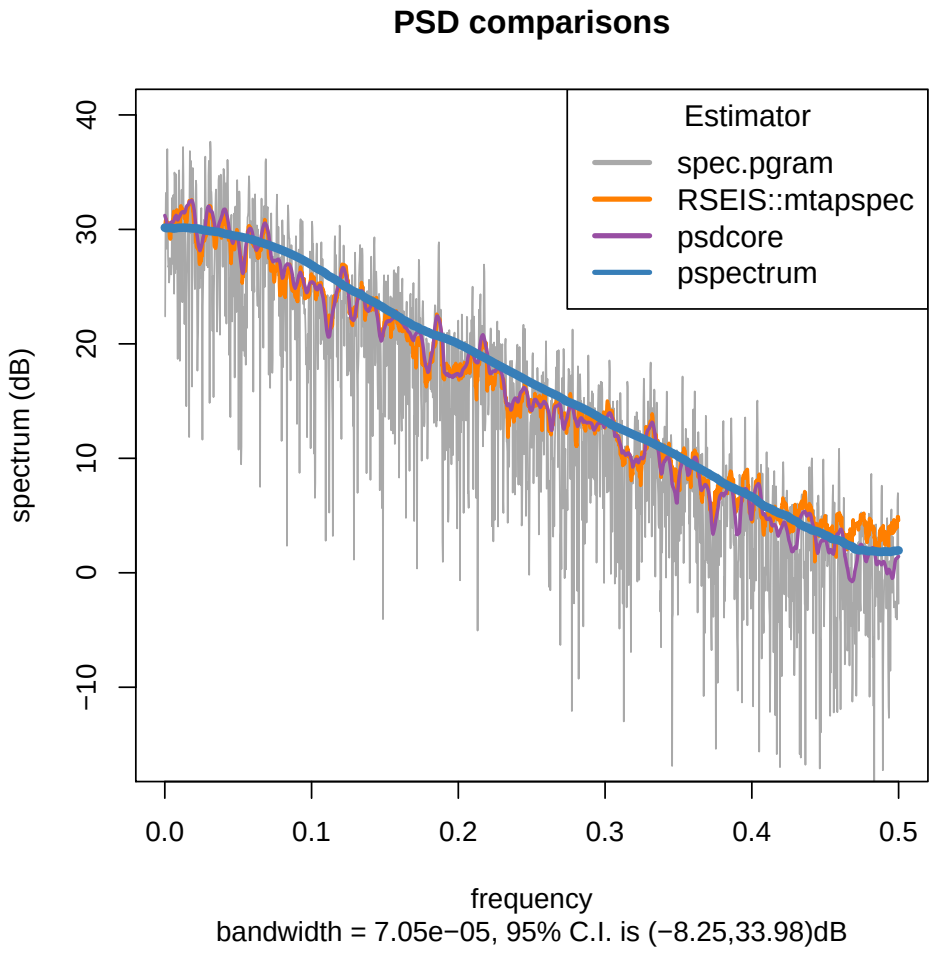


Figure 4: Comparisons of estimations of MAGSAT power spectral densities.

We regress the spectral values from `mtapspec` against the `psdcore` results because we have used them to produce uniformly tapered spectra with an equal number of sine tapers.

```
> df <- data.frame(x=dB(Pspec$spec), y=pltpi, tap=unclass(Aspec$taper))
> summary(dflm <- lm(y ~ x + 0, df))

Call:
lm(formula = y ~ x + 0, data = df)

Residuals:
    Min      1Q  Median      3Q     Max
-3.0996 -0.3012  0.2784  0.8471  4.5689

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
x 0.989507   0.001921     515   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.126 on 1024 degrees of freedom
Multiple R-squared: 0.9962,        Adjusted R-squared: 0.9962
F-statistic: 2.652e+05 on 1 and 1024 DF,  p-value: < 2.2e-16

> df$res <- residuals(dflm)
```

Create `ggplot2` objects for plotting purposes:

```
> require(ggplot2)
> g1 <- ggplot(df, aes(x=x, y=y)) +
+     geom_abline(intercept=0, slope=1, size=2, color="salmon")+
+     geom_point(aes(color=dB(y/x))) +
+     geom_smooth(colour="black", formula = y ~ x + 0, method="lm",
+                               se=TRUE, fullrange=TRUE) +
+     scale_colour_gradient2(mid="light grey") +
+     theme_bw() +
+     ggtitle("Regression of mtapspec against psdcore")
> g2 <- ggplot(df, aes(x=x, y=res)) +
+     geom_abline(intercept=0, slope=0, size=2, color="salmon") +
+     geom_point(aes(color=tap)) +
+     theme_bw() +
+     ggtitle("Regression residuals, colored by optimized tapers")
```

and show the regression and residuals in Figure 5. The structure visible in the residuals at low power levels may be from curvature bias in the `mtapspec` results, which manifests at short wavelengths in Figure 4.

2.3 `multitaper::spec.mtm`

The function with the highest similarity to `rlpSpec` is `spec.mtm` in the `multitaper` package: it uses the sine multitapers, and can adaptively refine the spectrum. In fact, this function calls source code of a Fortran equivalent to `rlpSpec` authored by R.L. Parker (2013) to do these operations.

There are some notable differences, though. By default `spec.mtm` uses the Discrete Prolate Spheroidal Sequences (dpss) of Thomson (1982), which can have very good spectral leakage suppression (assuming the number of tapers used is appropriate for the desired resolution, which varies inversely with the time-bandwidth product). Spectral analyses using dpss can have superior results if the series is relatively short

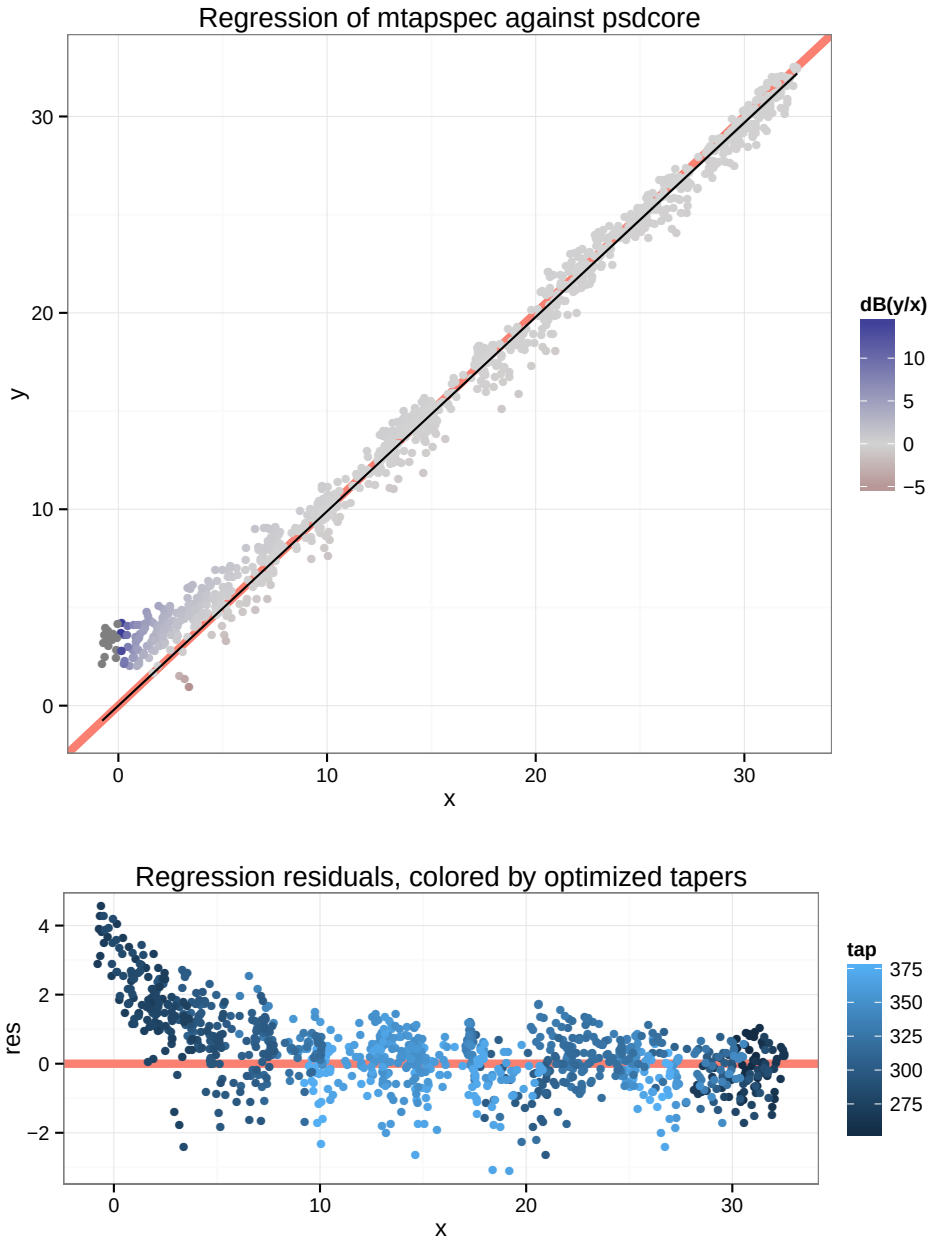


Figure 5: Regression of MAGSAT PSD estimates: `mtapspec` against `psdcore`.

(e.g. $N < 1000$), or has inherent spectra with sharply changing features or deep wells. Improper usage of the dpss, however, can lead to severe bias. Thus, considerable care should be given to parameter choices, which translates practicably to having many more knobs to turn.

2.4 `sapa::SDF`

As of this writing, the package has no maintainer; lest we end up discussing deprecated and archived functions, we will not compare it to `rlpSpec`.

3 Assessing spectral properties

3.1 Spectral uncertainties

It is important to place bounds on the uncertainties associated with a spectral estimate. In a multitaper algorithm the uncertainty is distributed as a χ^2_ν variate where ν is the number of degrees of freedom, which is twice the number of tapers applied. A proxy for this is simply $1/\sqrt{\nu - 1}$. Using $\nu = 2 * K$ we can approximate the distribution of uncertainties from the tapers alone; however, a more rigorous estimate comes from evaluating the appropriate distribution for a coverage probability (e.g. $p = 0.95$). Among other calculations, `spectral_properties` returns the χ^2_ν based confidence intervals for $p = 0.95$, as well as the approximate uncertainties.

To illustrate, we plot the uncertainties for an integer sequence[4] of tapers $[0, 50]$, shown in Figure 6. The benefits of having more than just a few tapers becomes obvious, though the spectral uncertainty is asymptotically decreasing with taper numbers and yields only slight improvements with logarithmic number of tapers.

Returning to the MAGSAT spectra, let us compare the `rlpSpec` spectra with spectral uncertainty as bounded polygons. First calculate the uncertainty polygon data:

```
> spp <- spectral_properties(Pspec$taper, db.ci=TRUE)
> spa <- spectral_properties(Aspec$taper, db.ci=TRUE)
> str(spa)

'data.frame':        1025 obs. of  8 variables:
 $ taper            : int  256 255 254 254 254 254 254 255 256 255 ...
 $ stderr.chi.upper : num  -0.518 -0.519 -0.52 -0.52 -0.52 ...
 $ stderr.chi.lower : num  0.547 0.548 0.549 0.549 0.549 ...
 $ stderr.chi.median: num  0.187 0.187 0.187 0.187 0.187 ...
 $ stderr.chi.approx: num  0.188 0.188 0.189 0.189 0.189 ...
 $ resolution       : num  0.501 0.5 0.498 0.498 0.498 ...
 $ dof              : num  512 510 508 508 508 508 508 510 512 510 ...
 $ bw               : num  0.251 0.25 0.249 0.249 0.249 ...

> create_poly <- function(x, y, dy){
+   xx <- c(x, rev(x))
+   yy <- c(y+dy, rev(y-dy))
+   return(data.frame(xx=xx, yy=yy))
+ }
> pspp <- create_poly(Pspec$freq, dB(Pspec$spec), spp$stderr.chi.approx)
> psppu <- create_poly(Pspec$freq, dB(Pspec$spec), spp$stderr.chi.upper)
> pspa <- create_poly(Aspec$freq, dB(Aspec$spec), spa$stderr.chi.approx)
> pspau <- create_poly(Aspec$freq, dB(Aspec$spec), spa$stderr.chi.upper)
```

```
> sp <- spectral_properties(as.tapers(1:50), p=0.95, db.ci=TRUE)
> plot(stderr.chi.upper ~ taper, sp, type="s",
+        ylim=c(-10,20), yaxs="i", xaxs="i",
+        xlab=expression("number of tapers ("* nu/2 *")"), ylab="dB",
+        main="Spectral uncertainties")
> mtext("(additive factor)", line=.3)
> lines(stderr.chi.lower ~ taper, sp, type="s")
> lines(stderr.chi.median ~ taper, sp, type="s", lwd=2)
> lines(stderr.chi.approx ~ taper, sp, type="s", col="red",lwd=2)
> # to reach 3 db width confidence interval at p=.95
> abline(v=33, lty=3)
> legend("topright", c(expression("Based on "* chi^2 *"(p,"*nu*") and (1-p,"*nu*")"),
+                      expression(""* chi^2 *"(p=0.5,"*nu*")"), "approximation"),
+                      lwd=c(1,3,3), col=c("black","black","red"),
+                      bg="white")
```

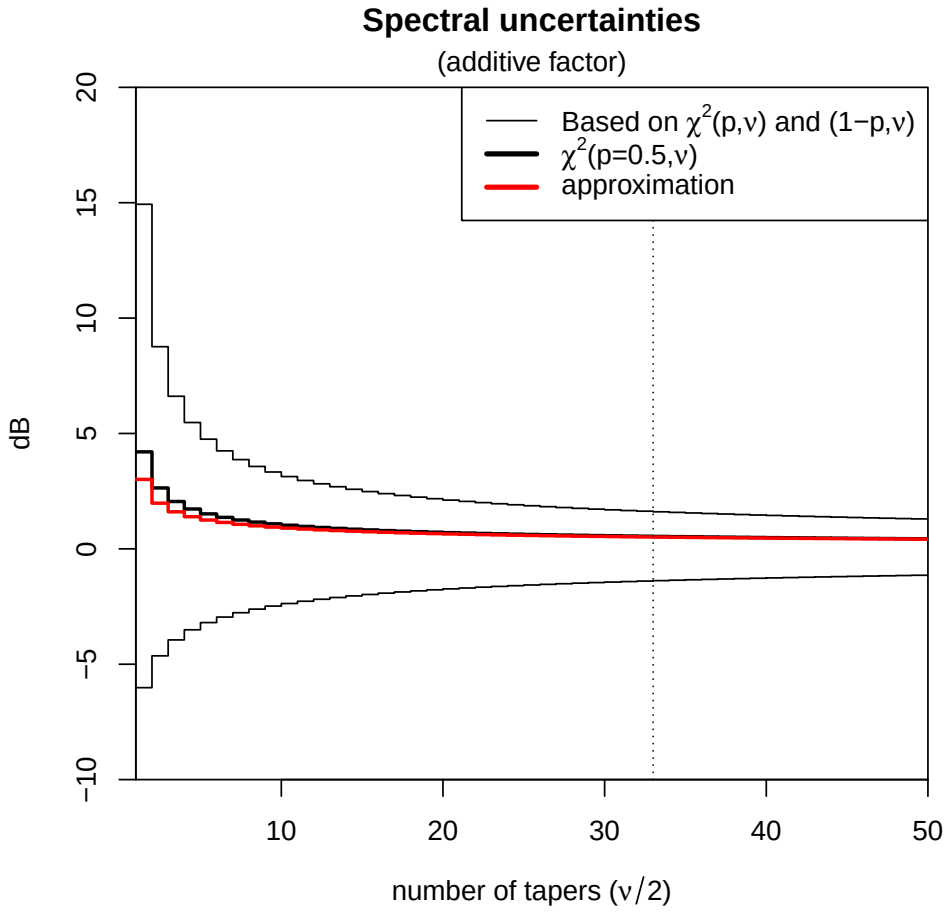


Figure 6: Additive spectral uncertainties by number of tapers. These quantized curves are found by evaluating the χ^2_ν distribution, where ν is the number of degrees of freedom (two per taper). The black lines show uncertainties for a coverage probability of 0.95. The thick, red line shows an approximation to these uncertainties based on $1/\sqrt{\nu - 1}$, which is accurate to within a few percent in most cases.

```
> plot(c(0,0.5),c(-8,35),col="white",
+        main="MAGSAT Spectral Uncertainty (p > 0.95)",
+        ylab="", xlab="spatial frequency, 1/km", yaxt="n", frame.plot=FALSE)
> lines(c(2,1,1,2)*0.01,c(5,5,8.01,8.01)-8)
> text(.05, -1.4, "3.01 dB")
> polygon(psppu$xx, (psppu$yy), col="light grey", border="black", lwd=0.5)
> polygon(pspp$xx, (pspp$yy), col="dark grey", border=NA)
> text(0.15, 6, "With adaptive\ntaper refinement", cex=1.2)
> polygon(pspau$xx, (pspau$yy)-10, col="light grey", border="black", lwd=0.5)
> polygon(pspa$xx, (pspa$yy)-10, col="dark grey", border=NA)
> text(0.35, 22, "Uniform tapering", cex=1.2)
```

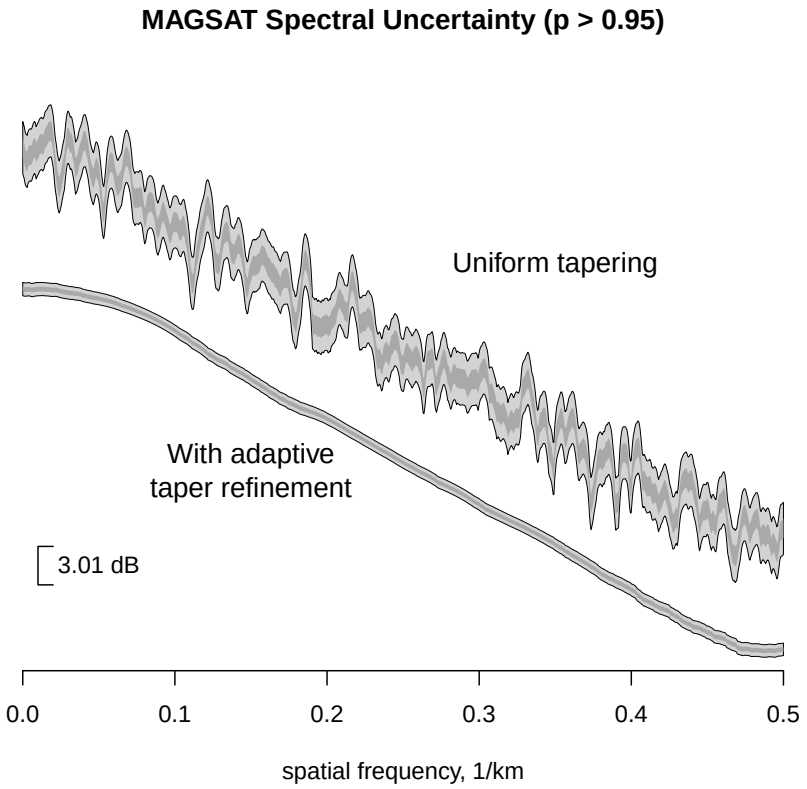


Figure 7: MAGSAT Spectral uncertainties with and without adaptive taper optimization. The filled regions encompass the upper limit and approximate values of spectral uncertainty .

and plot the comparison, shown in Figure 7.

3.2 Spectral resolution

There is an inherent tradeoff between the number of tapers applied and the spectral resolution (effectively, the spectral bandwidth). In general, the greater the number of tapers applied, the lower the spectral resolution. We can use the information returned from `spectral_properties` to visualize the actual differences in resolution for the MAGSAT PSD estimates; these are shown in Figure 8.

```
> frq <- Aspec$freq
> psd <- dB(Aspec$spec)
> kopt <- unclass(Aspec$taper)
> spres <- spa$resolution
> pltdf <- data.frame(frq=frq, p=psd, relp=psd + dB(spres * frq), col="light grey")
> pltdf$col[pltdf$relp<0] <- "light blue"
> plot(relp ~ frq, pltdf, type="h",ylim=18*c(-1,1), col=pltdf$col,
+        xaxs="i", yaxs="i", main="MAGSAT Spectral Resolution",
+        ylab="dB", xlab="spatial frequency, 1/km")
> abline(h=0, lwd=1)
> lines(relp ~ frq, pltdf, lwd=2)
```

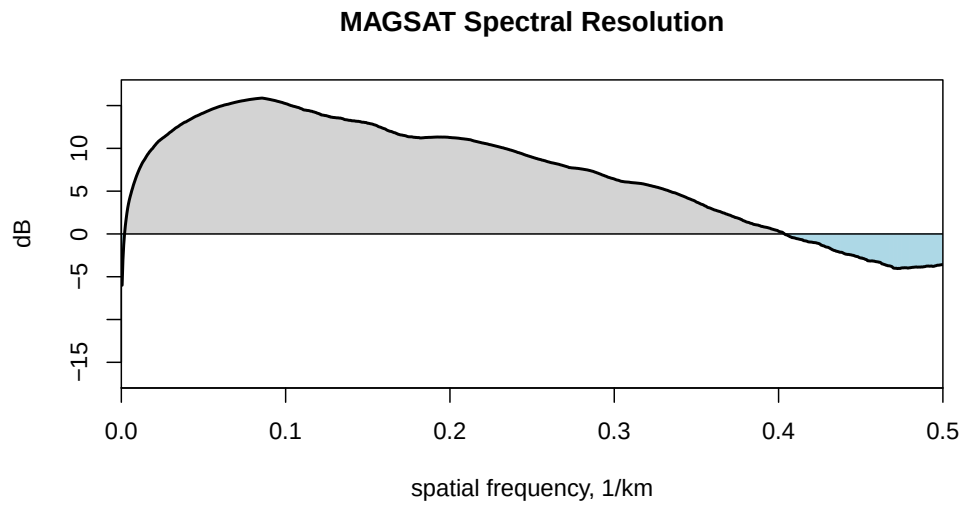


Figure 8: Resolution limits for MAGSAT PSD (`niter=3` adapts).

3.3 Visualizing the adaptive history

One might be curious to study how the uncertainties change with each iteration. `pspectrum` saves an array of "historical" data in its working environment. Specifically, it saves the frequencies, spectral values, and number of tapers at each stage of the adaptive procedure, accessible with `get_adapt_history`. To ensure a

[4] Note the χ^2_ν distribution is defined for non-negative, non-integer degrees of freedom, but we cannot apply fractions of tapers.

fresh calculation and to add a few more iterations to visualize, we repeat the adaptive spectral analysis, and then bring the stage history into the `.GlobalEnv` environment:

```
> pspectrum(ats, niter=6)
> str(AH <- get_adapt_history())

List of 3
 $ freq    : num [1:1025] 0 0.000488 0.000977 0.001465 0.001953 ...
 $ stg_kopt:List of 7
  ..$ :Class 'tapers'  int [1:1025] 9 9 9 9 9 9 9 9 9 9 ...
  ..$ :Class 'tapers'  int [1:1025] 48 49 50 51 50 49 48 47 47 48 ...
  ..$ :Class 'tapers'  int [1:1025] 129 130 131 132 133 134 133 132 131 130 ...
  ..$ :Class 'tapers'  int [1:1025] 230 231 231 231 231 232 231 231 230 230 ...
  ..$ :Class 'tapers'  int [1:1025] 275 276 276 276 276 276 276 276 276 276 ...
  ..$ :Class 'tapers'  int [1:1025] 324 324 324 324 324 324 324 324 324 324 ...
  ..$ :Class 'tapers'  int [1:1025] 330 330 330 330 330 330 330 330 330 330 ...
 $ stg_psd :List of 7
  ..$ : num [1:1025] 1144 1228 1318 1363 1377 ...
  ..$ : num [1:1025] 1455 1475 1494 1503 1509 ...
  ..$ : num [1:1025] 1441 1439 1437 1429 1432 ...
  ..$ : num [1:1025] 1075 1085 1095 1095 1095 ...
  ..$ : num [1:1025] 915 913 912 914 915 ...
  ..$ : num [1:1025] 939 939 940 941 940 ...
  ..$ : num [1:1025] 944 944 943 943 942 ...
```

Followed by some trivial manipulation:

```
> Freqs <- (AH$freq)
> Dat <- AH$stg_psd
> numd <- length(Freqs)
> numit <- length(Dat)
> StgPsd <- dB(matrix(unlist(Dat), ncol=numit))
> Dat <- AH$stg_kopt
> StgTap <- matrix(unlist(Dat), ncol=numit)
> rm(Dat, AH)
```

We can plot these easily with `matplot` or other tools. We show the adaptive history in Figure 9.

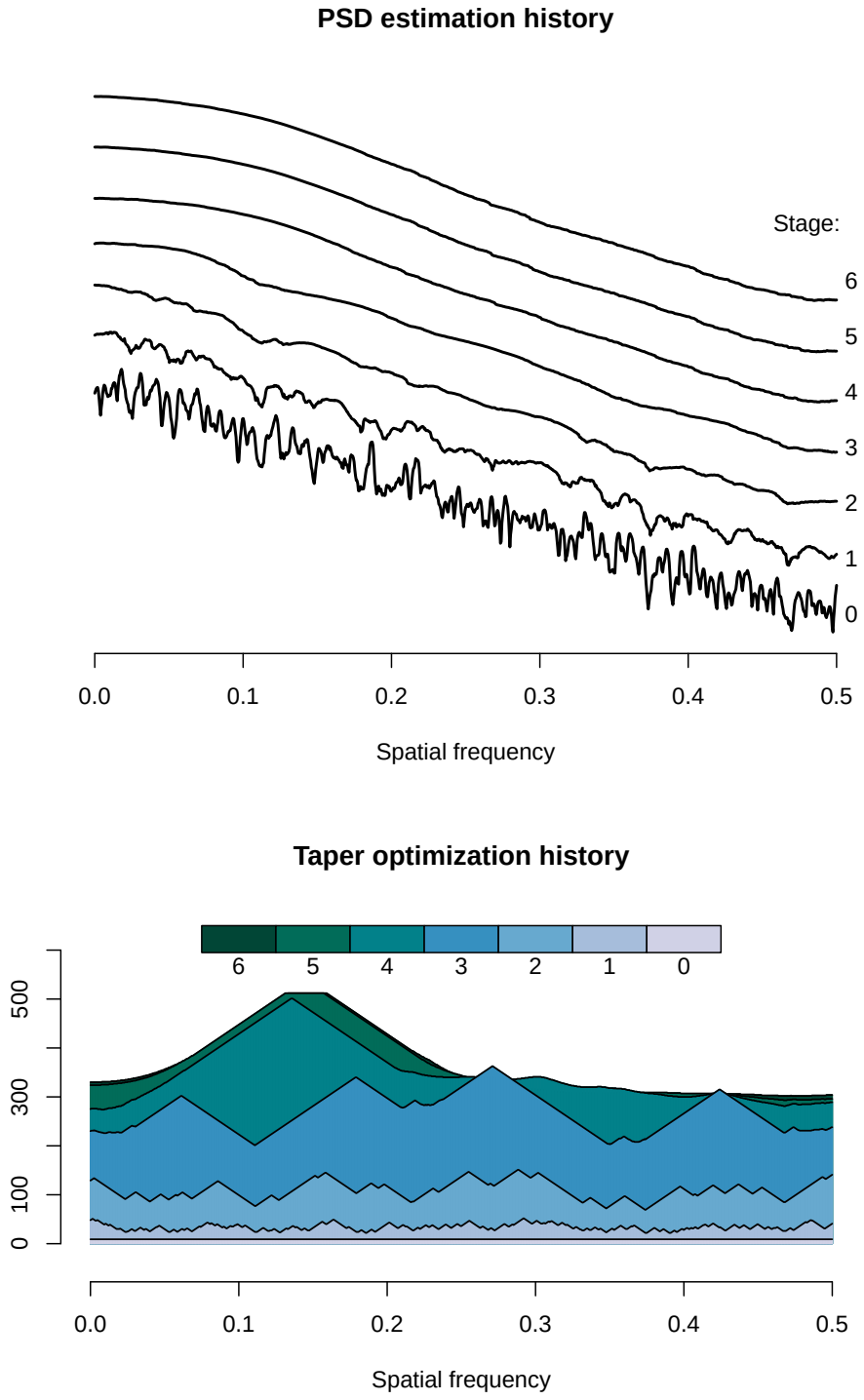


Figure 9: Adaptive spectral estimation history. Top: Sequential PSD series for each stage of the adaptive method, offset by a few decibels for visualization purposes. Bottom: Filled polygons showing the number of tapers at each stage.

4 Call overview

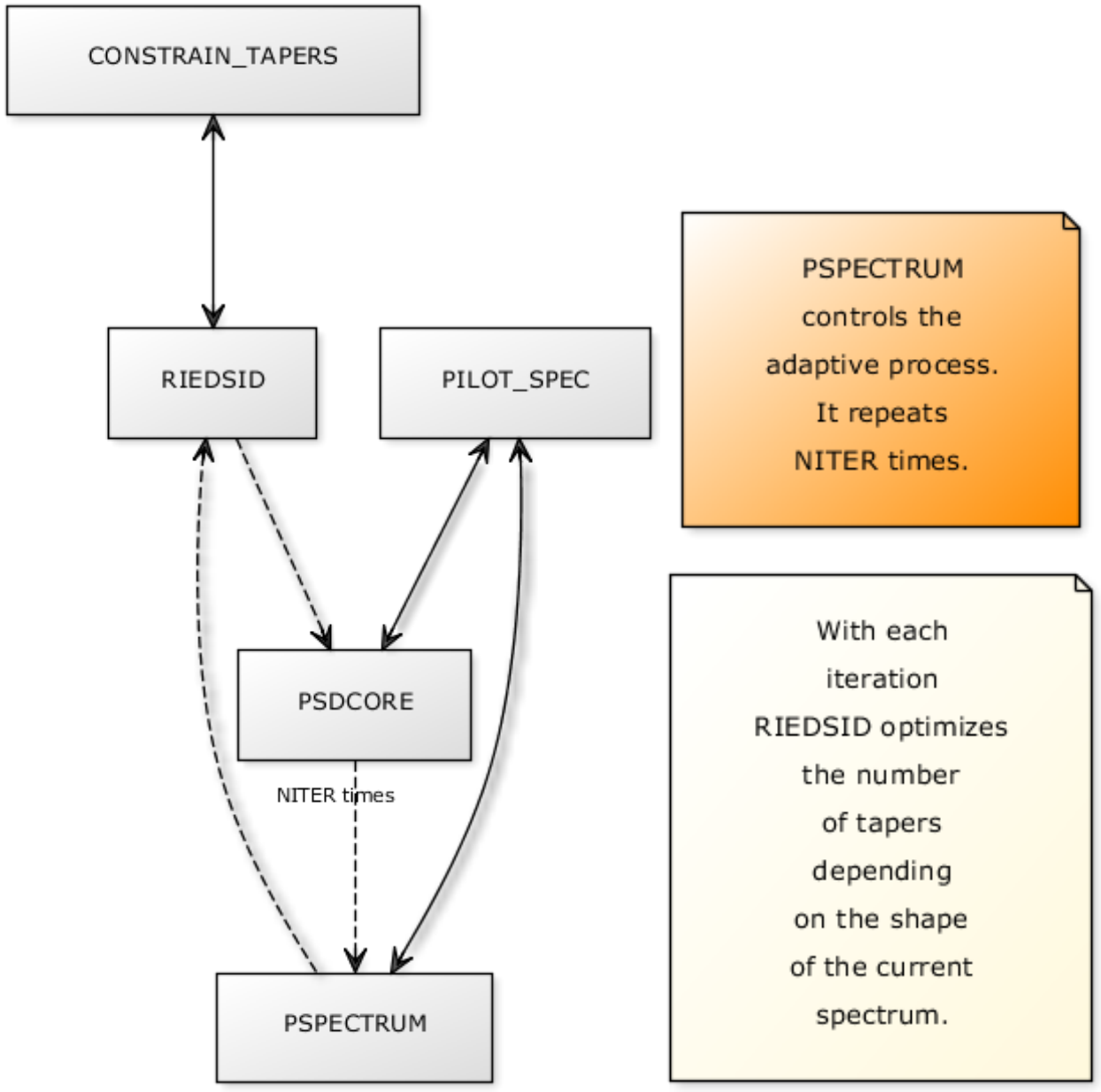


Figure 10: Simplified call graph for `rlpSpec`. The dashed lines show a simplified circuit in which the spectra and its tapers make during the iterative process.

References

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

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

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

Parker, R. L. (2013). PSD. `http://igppweb.ucsd.edu/%7Eparker/Software/`. *Maintained software (last accessed 30 Jan 2013)*.

Parker, R. L. and Barbour, A. J. (2013). *rlpSpec: Adaptive, sine-multitaper power spectral density estimation*. R package version 0.2-0.

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

R Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Rahim, K. and Burr, W. (2012). *multitaper: Multitaper Spectral Analysis*. R package version 1.0-2.

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

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