Multi-Taper Spectrum Estimation in R

Jonathan M. Lees
University of North Carolina, Chapel Hill
Department of Geological Sciences
CB #3315, Mitchell Hall
Chapel Hill, NC 27599-3315
email: jonathan.lees@unc.edu
ph: (919) 962-0695

April 1, 2011

1 Introduction

This vignette is intended to show how to use R to reproduce the results illustrated in *Lees and Park*(1995):

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

The traditional single taper analysis has the objectionable feature that portions of the time series are excluded from analysis as a trade-off for reducing spectral leakage in the frequency domain. Smooth spectrum estimates using Slepian tapers avoids this tradeoff.

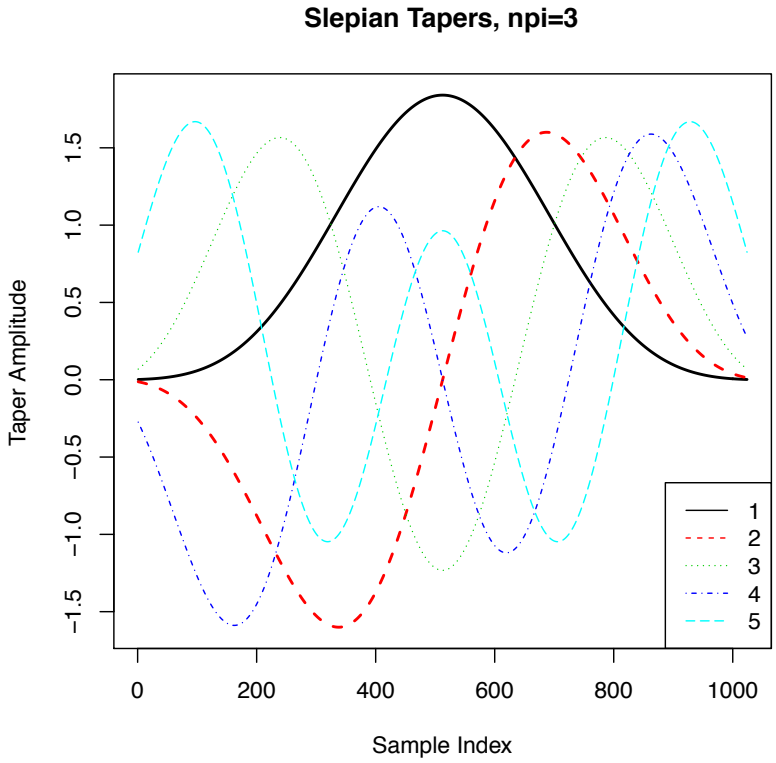
In this demonstration I present a portable subroutine for calculating a multitaper spectrum estimate for a geophysical or geological time series. Multitaper analysis arises as an extension of traditional taper analysis where time series (or auto-corelation functions) are tapered prior to Fourier transform to reduce bias due to leakage. Common single taper methods include applying Hann, 20% cosine or numerous other tapers, which reduce the effects

of spectral leakage. In the multitaper approach an orthonormal sequence of tapers is designed to minimize spectral leakage. The set of tapers and the associated eigenspectra can be combined to reduce the variance of the overall spectrum estimate. One of the main advantages of the multitaper spectrum estimates is that formal estimates of their statistical degrees of freedom and variance are simple consequences of the procedure. Alternatively, nonparametric jackknife estimates of variance in spectrum and coherence estimates are easily implemented. In addition, the algorithm includes an F-test which can be used to identify the location and confidence bounds of spectral peaks presumed to represent periodic, phase-coherent signals.

2 Slepian Tapers

The following code shows how to extract and plot the Slepian tapers. These tapers optimize the trade-off of leakage bias to loss of information content in the signals. Note that the first Slepian taper looks similar to a gaussian. The second weights the left and right halves with opposite sign. The others include variable amounts of information along the trace.

```
>   nwin = 5
>      npi = 3
>      npoints = 1024
>      sleps = get.slepians(npoints, nwin, npi)
> linwds = rep(1, times=nwin)
> linwds[1:2] = 2
>      matplot(sleps, type='l', xlab="Sample Index", ylab="Taper Amplitude", lw
>      legend('bottomright', legend=1:nwin, lty=1:nwin, col=1:nwin)
> title(main="Slepian Tapers, npi=3")
```



3 Power Spectrum Estimates

These codes are used here to estiamte the power spectra. The first is a simple single hanning taper applied at 10% at each end of the time series.

```
> singleTaper<-function(y, dt, tappercent=0.1 )
+      {
+        if(missing(tappercent)) tappercent=0.1
+        N= length(y)
+        fn = 1/(2*dt)
+        tapy = spec.taper(y, p=tappercent)
+        ##tapy = tapy-mean(tapy)
+
+        Y = fft(tapy)
+        Pyy = (Mod(Y)^2)/(N*N)
```

```
+        ##  Pyy = Y * Conj(Y)
+        n = floor(length(Pyy)/2)
+        Syy = Pyy[1:n]
+        f = (0:(length(Syy)-1))*fn/length(Syy)
+        #####    plot(f, Syy, type='l', xlab="frequency", ylab="Power Density"
+
+        invisible(list(f=f, syy=Syy))
+
+      }
```

The following code is a function that will plot the multitaper estimate and other estimates for comparison:

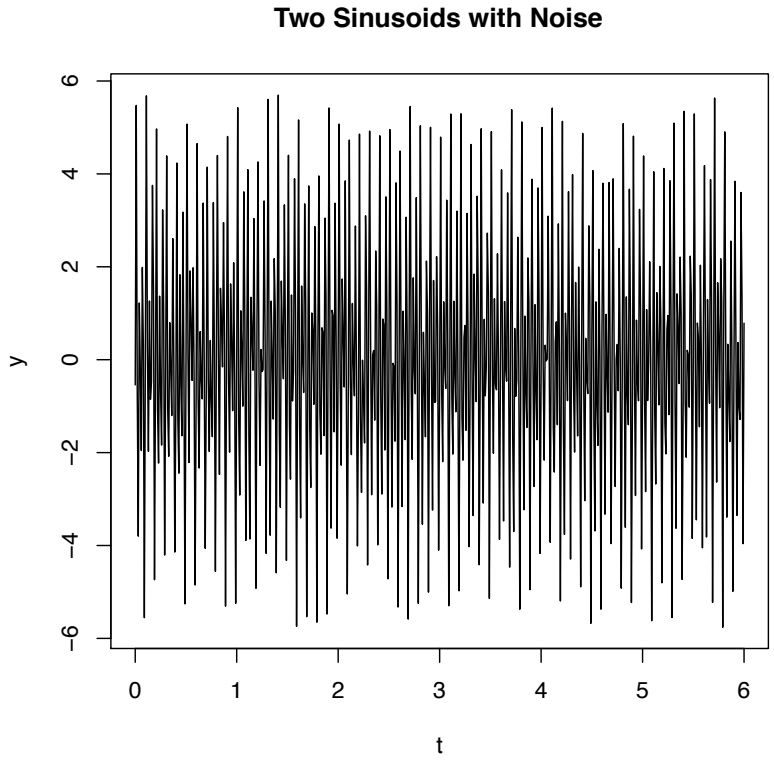
```
> DOLEESPARK<-function(y, dt, tappercent=0.1)
+   {
+
+     y = y-mean(y)
+
+     ####
+
+      nn=next2(length(y))
+
+     Mspec =   mtapspec(y, dt, klen=nn,  MTP=list(kind=2,nwin=5, npi=3,inorm=
+     f = Mspec$freq
+
+     amp1 = Mspec$spec[1:length(f)]
+     amp = 10*log10(amp1 )
+
+     ####
+     singy = singleTaper(y, dt, tappercent=tappercent)
+
+     stap1 = 10*log10(singy$syy )
+
+
+     ####
+     squig = list(y=y, dt=dt)
+     ZIM = autoreg(squig , numf=length(Mspec$freq) , pord = 80, PLOT=FALSE)
+
```

```
+     AutoR = 10*log10(ZIM$amp )
+
+     ####
+     frange = range(c( f ))
+     amprange  = range(c(amp, stap1, AutoR))
+
+     ####
+     plot(frange, amprange, type='n',ylab="Spectrum Amplitude", xlab="Frequenc
+
+     lines(f, amp)
+
+     lines(ZIM$freq,AutoR, col='red')
+     lines(singy$f,stap1, col='blue')
+     ####
+     title("Lees and Park, 1995")
+     legend("topright", legend=c("multitaper",  "Autoregressive", "Single-tap
+
+
+   }
```

4 Synthetic Example

This section shows how the spectrum estimates are made and how they compare with standard single taper estimates.

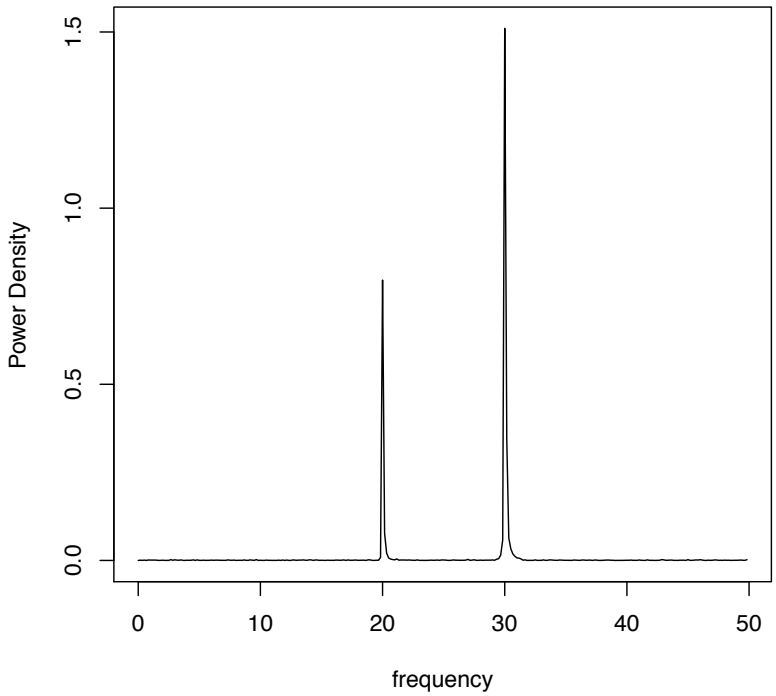
```
>   library(RSEIS)
>     f1 =20
> f2 = 30
> dt = 0.01
> t = seq(from=0, to=6, by=dt)
>   noise =  runif(length(t), 0, 2)
> y = 2*sin(2*pi*f1*t) + 3*sin(2*pi*f2*t) + noise
>   y = y-mean(y)
> par(mfrow=c(1,1))
> plot(t,y, type='l')
> title("Two Sinusoids with Noise")
```



Standard way to solve this problem with a single taper. Note that the scaling by N, the number of points in the time series, is one way to scale the power spectrum estimate. Other choices might include scaling by the variance in the time series.

```
> tappercent=.05
> N= length(y)
> fn = 1/(2*dt)
> tapy = spec.taper(y, p=tappercent)
>  ##tapy = tapy-mean(tapy)
>
> Y = fft(tapy)
> Pyy = (Mod(Y)^2)/(N*N)
>        ##  Pyy = Y * Conj(Y)
> n = floor(length(Pyy)/2)
> Syy = Pyy[1:n]
```

```
> fs = (0:(length(Syy)-1))*fn/length(Syy)
> plot(fs, Syy, type='l', xlab="frequency",
+      ylab="Power Density", log='')
```



The commands above can be combined into a function called "singleTaper".

For the multitaper method the function *mtapspec* is used. The time series, sample rate and length for output are parameters determined by the user. Here parameters associated with the MTM method are:

- kind=2, (use adwait)
- nwin=5, (number of tapers)
- npi=3, π-prolate tapers
- inorm=1, normalization

where *inorm* is the scaling, where 1 indicated scaling by N. Parameter *nwin* is the number of windows and *npi* is the order of $n\pi$-prolate tapers.

The *kind* parameter refers to either *hires* or *adwait* different ways of combining the *nwin* tapers to get the final estimate. See *Lees and Park*(1995) for details on these two approaches. Here the adaptive weighting scheme is illustrated.

The mtm estimate is attained by running:

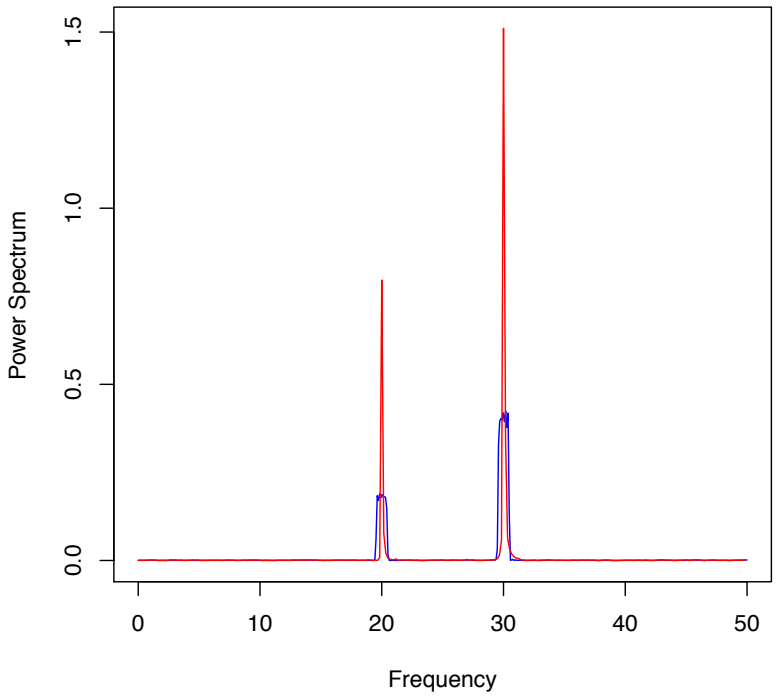
```
> Mspec=mtapspec(y,dt,klen=1024,
+   MTP=list(kind=2,nwin=5, npi=3, inorm=1))
> cat(names(Mspec), sep="\n")
dat
dt
spec
dof
Fv
Rspec
Ispec
freq
df
numfreqs
klen
mtm
```

- dat = input time series
- dt=delta-T
- spec=power spectrum
- dof=degrees of freedom
- Fv=F-values for ftest
- Rspec=matrix of real parts of tapered fft
- Ispec=matrix imaginary parts of tapered fft
- freq=vector of frequencies

- df=frequency interval
- numfreqs= number of frequencies calculated
- klen= length of fft
- mtm= parameters sent to program

We can then extract the relevant parts for plotting:

```
> f = Mspec$freq
>  amp1 = Mspec$spec[1:length(f)]
> Yrange  = range(c(amp1, Syy))
> plot(range(f), Yrange, type='n',
+ ylab="Power Spectrum", xlab="Frequency")
> lines(f, amp1, col="blue" )
> lines(fs, Syy, col='red')
```



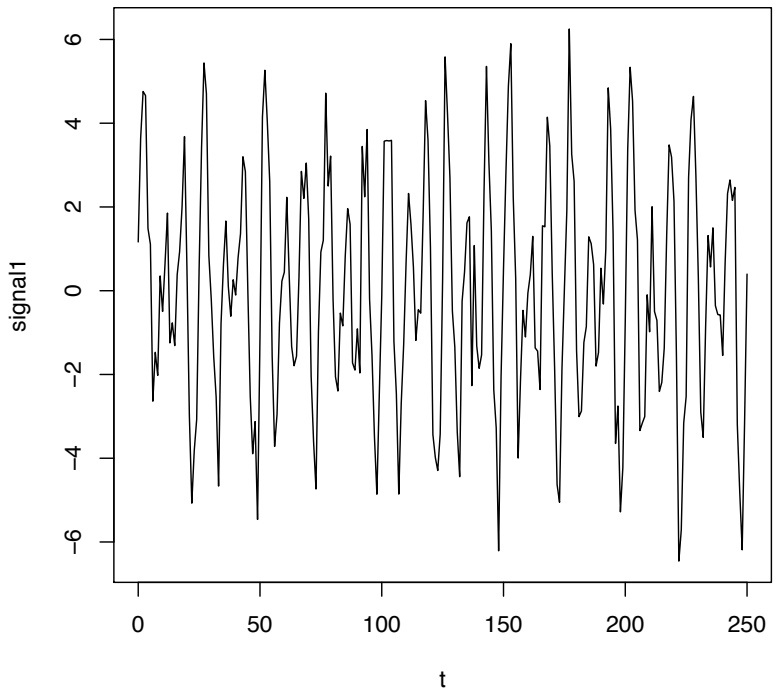
5 Lees and Park, 1995

In this section commands are presented that will reproduce figures in the *Lees and Park*(1995). These are similar to the examples above but they use the same data from the earlier publication.

5.1 Example 1

The first signal in *Lees and Park*,(1995) was created by adding 2 sinusoids with some noise.

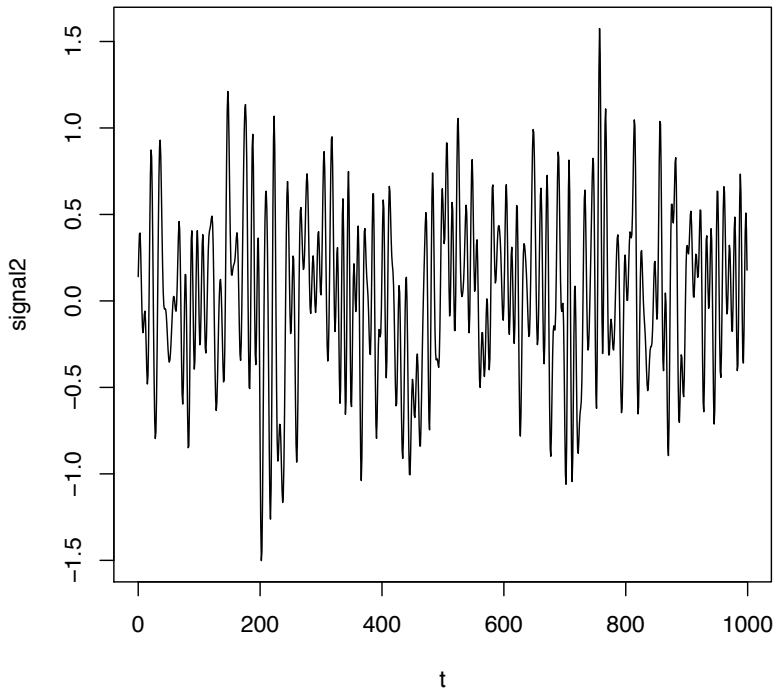
```
> signal1 = scan(file='signal1', skip=1)
> dt1 = 1
> t = seq(from=0, by=1, length=length(signal1))
> plot(t, signal1, type='l')
```



5.2 Example 2

The second example was created by summing 1000 realizations of a time series whose amplitudes are identical but whose phases vary randomly.

```
> signal2=scan(file='signal2', skip=1)
> dt2 = 1
> t = seq(from=0, by=dt2, length=length(signal2))
> plot(t, signal2, type='l')
```



5.3 MTM Comparison

We compare the mtm estimate with the single-taper and the auto-regressive estimate (Figure 1).

```
> DOLEESPARK(signal1, dt1, tappercent=0.05)
```

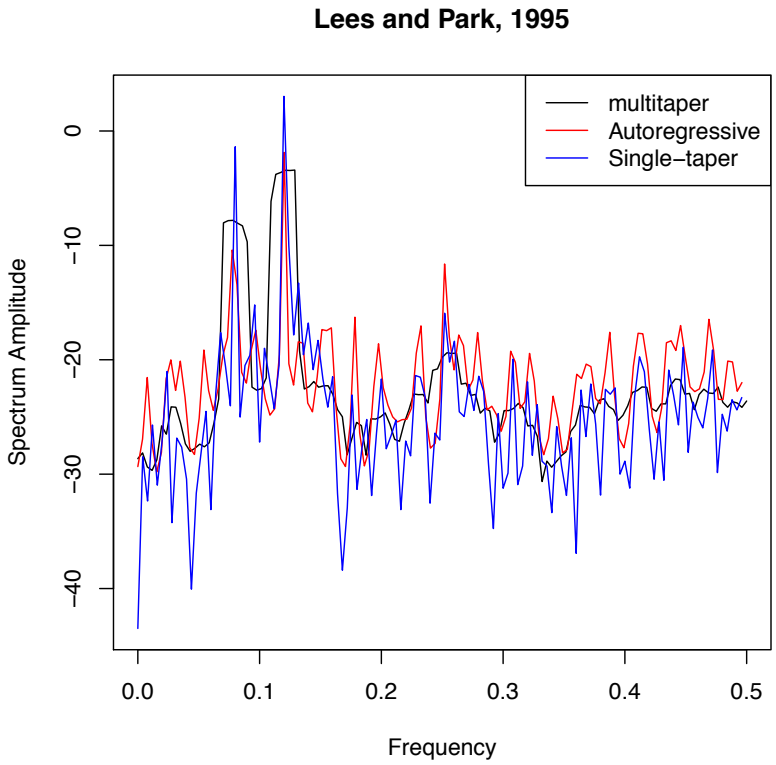


Figure 1: MTM example 1 comparison from Lees and Park 1995 paper

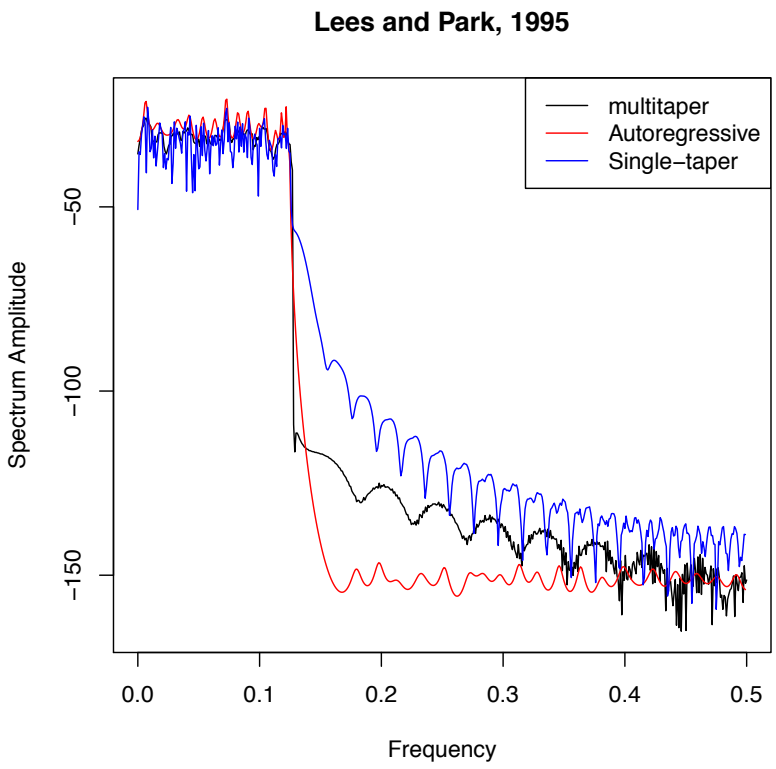


Figure 2: MTM example 2 comparison from Lees and Park 1995 paper

The second example (Figure 2).

```
> DOLEESPARK(signal2, dt2, tappercent=0.05)
```