Search for Milankovitch

Introduction

In this notebook we demonstrate hunt for Milankovitch cycles in the record of:

- Mix, A. C., J. Le, and N. J. Shackleton (1995a), Benthic foraminiferal stable isotope stratigraphy from Site 846: 0–1.8 Ma, Proc. Ocean Drill. Program Sci. Results, 138, 839–847.
- Shackleton, N. J. (1995), New data on the evolution of Pliocene climate variability, in Paleoclimate and Evolution, With Emphasis on Human Origins, edited by E. S. Vrba et al., pp. 242-248, Yale Univ. Press, New Haven, CT.

The data were aligned to the Benthic Stack of Lisiecki & Raymo (2005) using the HMM-Match algorithm (Khider et al, 2017).

We first load the data using the LiPD utilities.

```
library(lipdR)
library(geoChronR)
I = readLipd("ODP846.Lawrence.2006.lpd")
```

```
## [1] "reading: ODP846.Lawrence.2006.lpd"
```

```
d18O =  I$chronData[[1]]$model[[1]]$summaryTable[[1]]$d18O$values
t.med <- I$chronData[[1]]$model[[1]]$summaryTable[[1]]$median$values
L <- geoChronR::mapAgeEnsembleToPaleoData(I,which.pmt = 1)
```

```
## [1] "ODP846.Lawrence.2006"
## [1] "Looking for age ensemble...."
## [1] "Found it! Moving on..."
## [1] "Found it! Moving on..."
## [1] "getting depth from the paleodata table..."
## [1] "Found it! Moving on..."
```

```
age = geoChronR::selectData(L,"ageEnsemble",which.mt = 1)
```

```
## [1] "Found it! Moving on..."
```

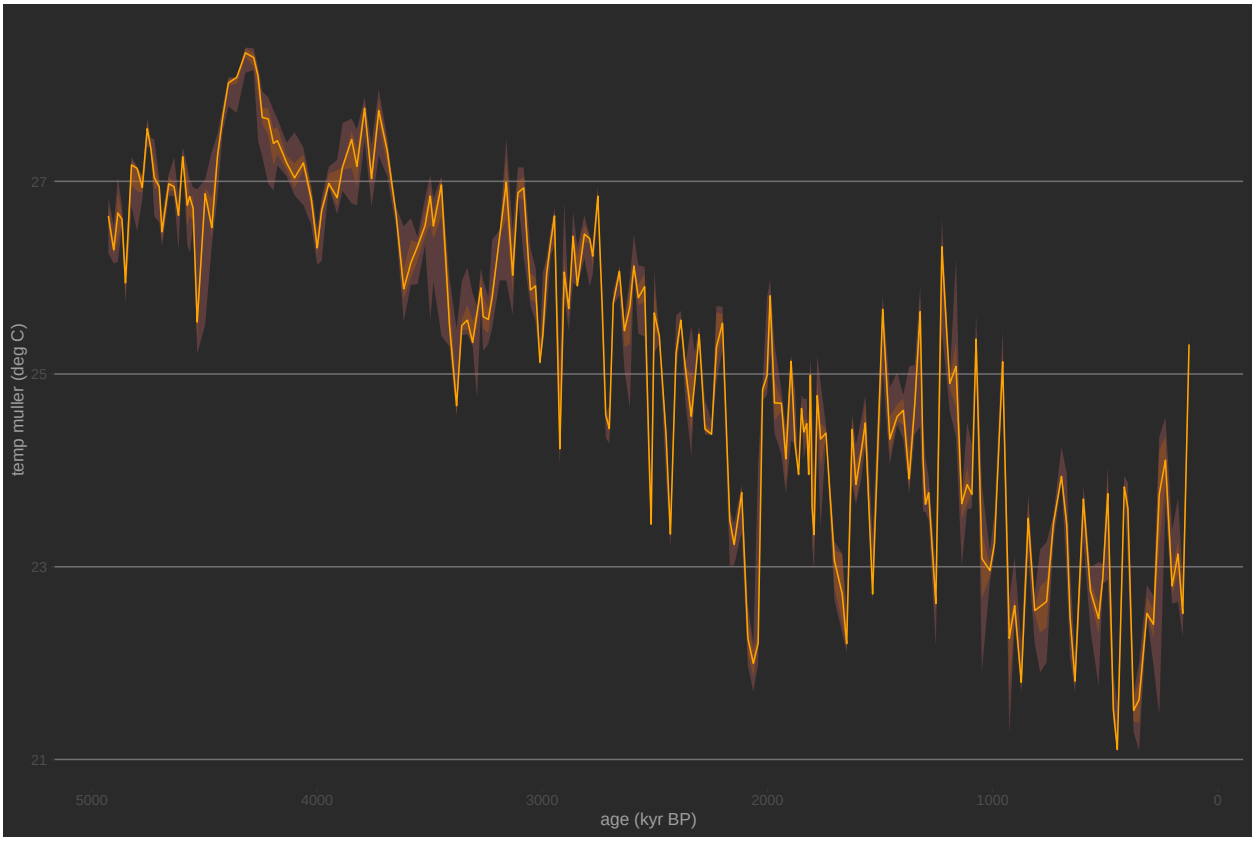
```
temp = geoChronR::selectData(L,"temp muller",which.mt = 1)
```

```
## [1] "Found it! Moving on..."
```

```
# define plotting theme
library(ggthemes)
```

Now let us take a first look at the median age model:

```
library(ggplot2)
#ggplot(df) + geom_line(aes(x=t/1000,y=d18O),colour="orange") + ggtitle("IODP 846 aligment to ProbStack
plotTimeseriesEnsRibbons(ggplot(),X=age,Y=temp,colorLow="orchid",colorHigh="darkorange3",lineColor="ora
```

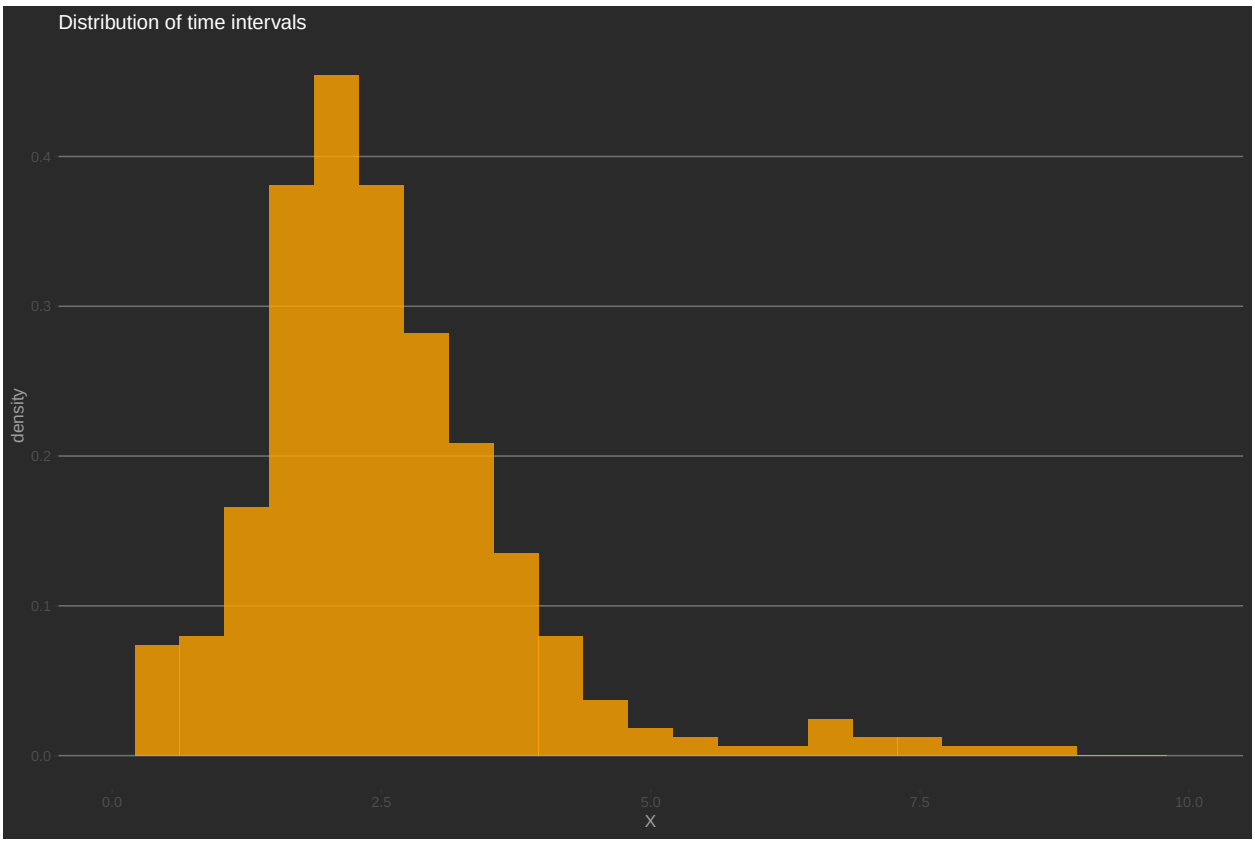


This paleoclimate record features:

- a long-term cooling trend (d18O gets more positive over time)
- some quasi-periodic oscillations (the legendary Pleistocene Ice Ages)
- nonstationary behavior, related to the well-known mid-Pleistocene transition from a "41k world" to a "100k world" somewhere around 0.8 Ma (Paillard, 2001).

To keep things simple and lower computational cost, let's focus on the last 1 Ma, and use the median age model. Now, a standard assumption of spectral analysis is that data are evenly spaced in time. In real-world paleo timeseries this is seldom the case. Let's look at the distribution of time increments in this particular core, as contrained by this tuned age model:

```
age.median = matrixStats::rowQuantiles(age$values,probs = 0.5)
temp.median = matrixStats::rowQuantiles(as.matrix(temp$values),probs = 0.5)
t <-age.median[age.median < 1000]
X <- temp.median[age.median < 1000]
X <- X - mean(X)
#dfs = dplyr::filter(df,t<=1000)
dt = diff(t)
ggplot() + xlim(c(0,10)) +
  geom_histogram(aes(x=dt,y = ..density..), bins = 25, ,alpha = 0.8, fill = "orange") + ggtitle("Distril
```



We see that over the past 1 Ma, the time increments (dt) are sharply peaked around 2 ka, but they range from 0 to about 7.5 ka. For now, let us assume that the time axis, albeit uneven, is well-known (no uncertainty).

Time-certain spectral analysis

From here there are two ways to proceed: 1) use methods that explictly deal with unevenly-spaced data, or 2) interpolate to a regular grid and apply standard methods. For 1, we could use the *Lomb-Scargle periodogram* or the lesser-known *nuspectral* package. Let's see what each method brings to the table.

Lomb-Scargle periodogram

This is a very standard method implemented in many packages. For a review, VanderPlas (2018). There are several ways to implement Lomb-Scargle. GeoChronR does this via the lomb package.

To establish significance, we have a few choices. We re-use Stephen Meyer's excellent astrochron package for this purpose, as it implements the methods described in Meyers, (2012).

```
library(lomb)
spec.ls <-  lomb::lsp(x=X,times=t,ofac=2,plot = F)
ls.df <- data.frame("freq" = spec.ls$scanned, "pwr" = spec.ls$power)
# estimate significance against a power-law fit
f.low <-  1e-3; f.high <- 0.1
plaw.ls <-  astrochron::pwrLawFit(ls.df, dof = 2, flow = f.low, fhigh = f.high, output = 1, genplot = F
```

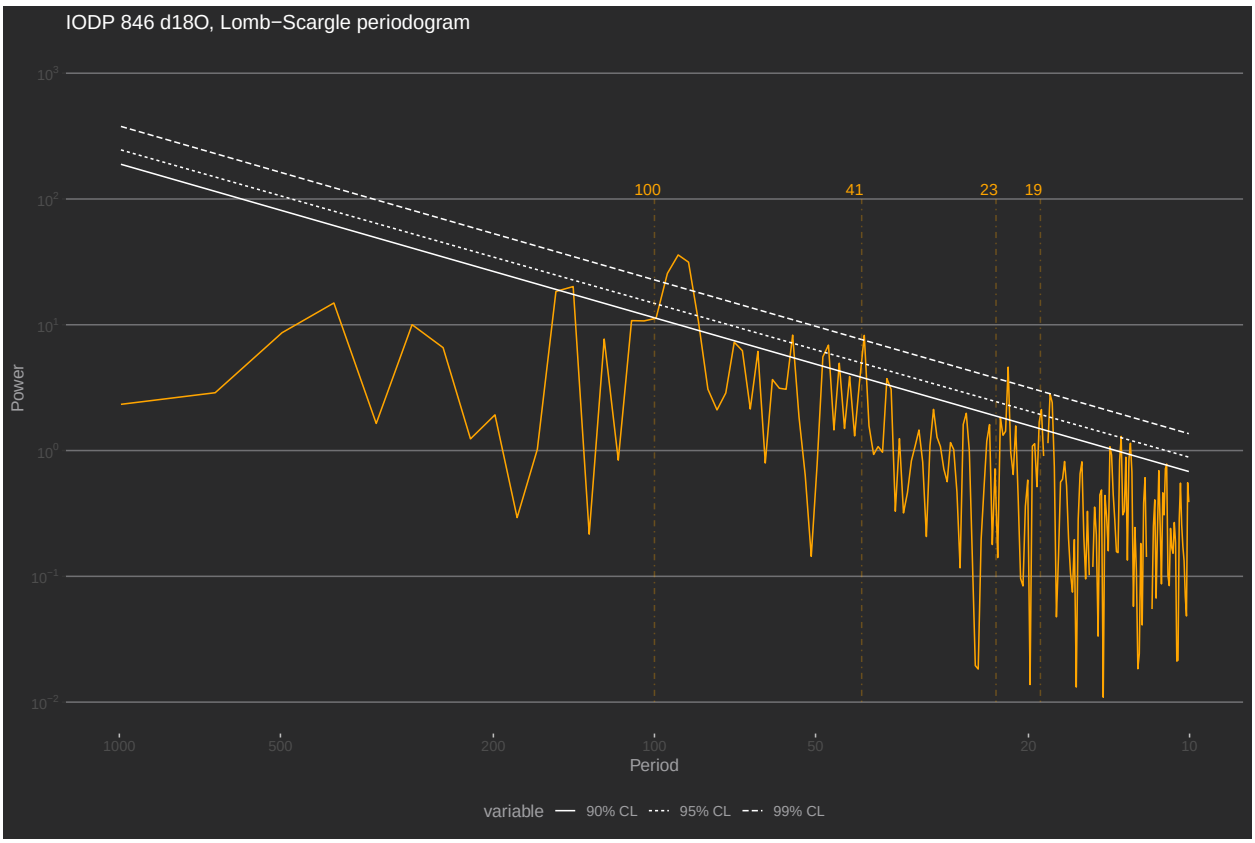
```
##
## ----- ESTIMATING Power Law (1/f) Fit -----
##  * Number of frequencies in spectrum: 391
```

```
##  * Minimum frequency: 0.001006844
##  * Maximum frequency: 0.1973414
##  * Frequency spacing: 0.0005034219
##  * Slope from log(power) vs. log(frequency) fit (m): -1.224017
##  * Y-intercept from log(power) vs. log(frequency) fit (b): -2.00534
##  * beta = 1.224017
##  * log(N) = -1.754658
##  * estimated bias = -0.2506816
```

```r
cl.df <- data.frame(plaw.ls[,union(1,c(5:7))]) # extract confidence limits
# rename columns to be less silly
names(cl.df)[1] <- "freq"
names(cl.df)[2] <- "90% CL"
names(cl.df)[3] <- "95% CL"
names(cl.df)[4] <- "99% CL"

# plot this
pticks = c(10, 20, 50, 100, 200, 500, 1000)
prange = c(10,1000)
yl = c(0.01,1000)
p.ls <- plotSpectrum(ls.df,cl.df,period_range=prange,period_ticks = pticks, ylims = yl) +
  ggtitle("IODP 846 d18O, Lomb-Scargle periodogram") +
  theme_hc(style = "darkunica") + theme(axis.ticks.x = element_line(color = "gray"))

# label periodicities of interest
p.ls <- PeriodAnnotate(p.ls, periods = c(19,23,41,100), ylims =c(0.01,100))
show(p.ls)
```



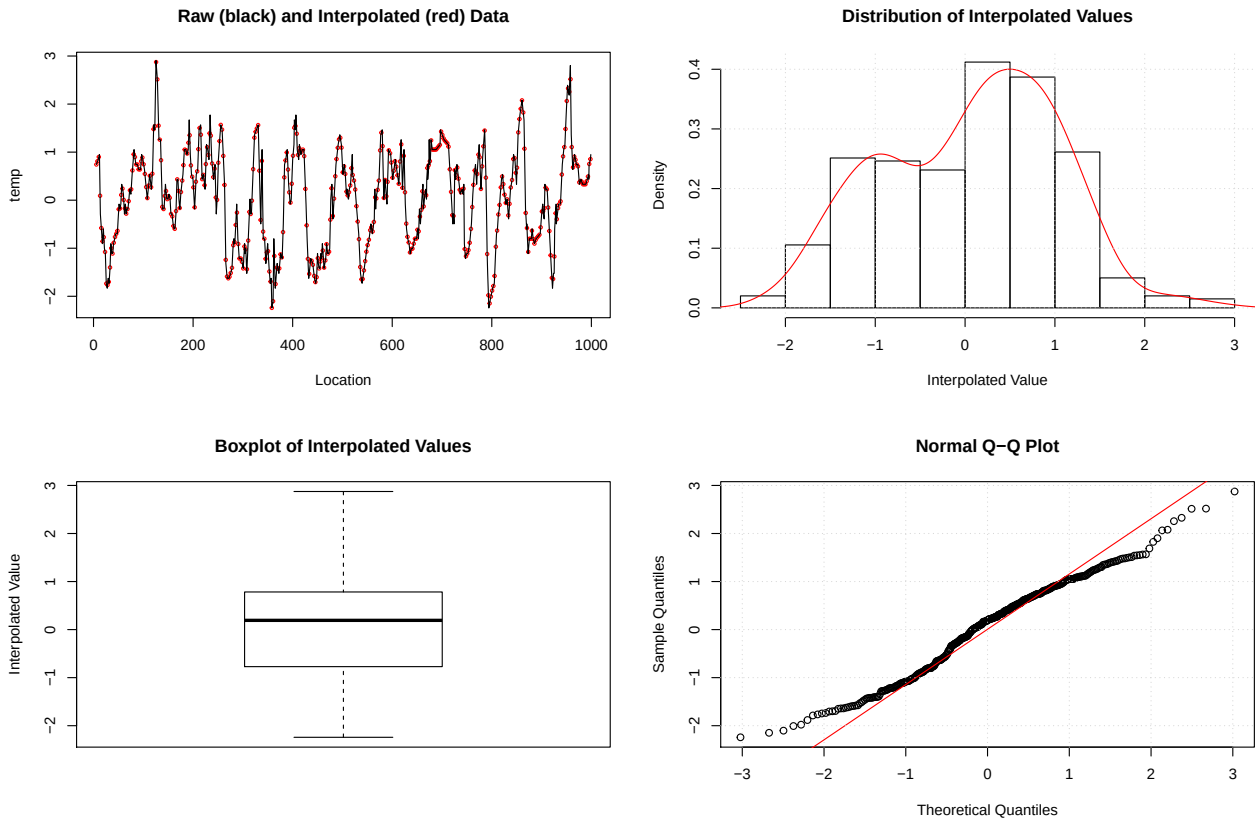
Where it is clearly seen that the data appear to contain periodic signals near, but not exactly at, the Milankotich frequencies. These frequencies (particularly the eccentricity and obliquity frequencies) rise above the various power law nulls, but we see hints of higher power at high-frequencies. We shall soon see that this is an artifact of Lomb-Scargle. Other methods smooth out that noise.

Multi-taper Method

The Lomb-Scargle periodogram is a decent way to deal with unevenly-spaced timeseries, but it is still a periodogram, i.e. one of the worst estimators known to humankind. In particular, it is inconsistent: the variance of each estimate goes to infinity as the number of observations increases. A much better estimator is Thomson's Multi-Taper Method [Thomson, 1982], which is consistent (the variance of each estimate goes to zero asymptotically, meaning that the more data you have, the better you know the spectrum). Formally, MTM optimizes the classic bias-variance tradeoff inherent to all kinds of statistical inference. It does so by minimizing leakage outside of a frequency band with half bandwidth equal to pf_R, where $f_R = 1/(N\Delta t)$ is the Rayleigh frequency, Δt is the sampling interval, N the number f measurements, and p is the so-called time-bandwidth product. p can only take a finite number of values, all multiples of $1/2$ between 2 and 4. A larger p means lower variance (i.e. less uncertainty about the power), but broader peaks (i.e. a lower spectral resolution), synonymous with more uncerainty about the exact location of the harmonic. So while MTM might not distinguish between closely-spaced harmonics, it is much less likely to identify spurious peaks, especially at high frequencies. In addition, a battery of formal tests have been devised with MTM, allowing under reasonably broad assumptions to ascertain the significance of spectral peaks. We show how to use this "hamornic F-test below"

A notable downside of MTM is that it only handles evenly-spaced data. Small potatoes! As we saw above, the data are not that far from evenly-spaced, so let's interpolate and see what we get. Conveniently, both the interpolation routine and MTM are available within the astrochron package.

```
library(astrochron)
dfs = data.frame(time=t,temp=X)
dti = 2.5  # interpolation interval, corresponding to the mode of the \Delta t distribution
dfe = linterp(dfs,dt=dti)
```



Most of the `astrochron` routines produce a diagnostic graphical output, which you can silence by turning the `genplot` flag to `FALSE`. However, it is instructive to take a peak at the top-left panel and see where the interpolation has made a difference. We see that the black line closely espouses the red measurements for most of the timeseries, reassuring us that this process did not introduce spurious excursions or oscillations.

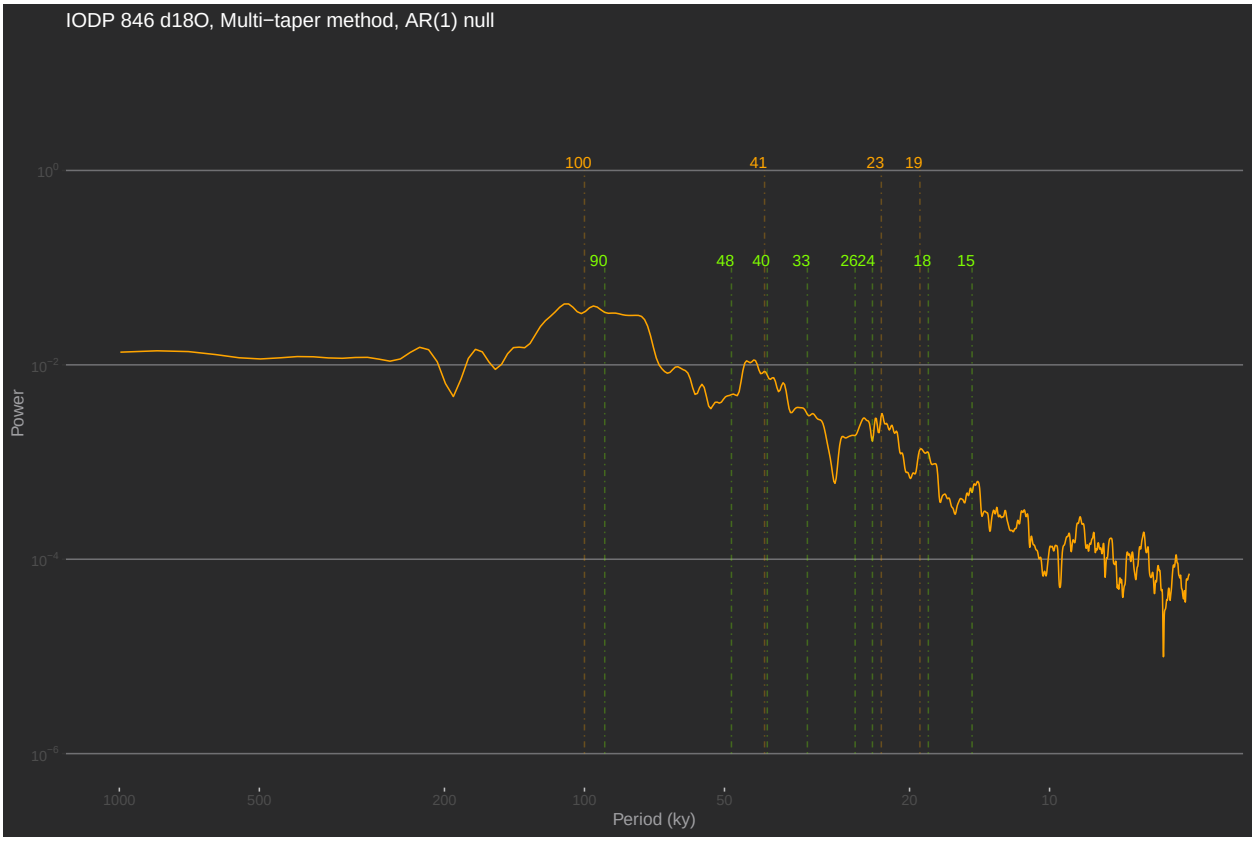
Now let's compute the spectrum using MTM on the equally-spaced data stored in `dfe`. We continue to manually label Milankovitch frequencies (in orange) and label in green (technically, "chartreuse") the periodicities idenitified as significant by the test. Here, we start with the default option of a test against an AR(1) background.

```
spec.mtm <- astrochron::mtm(dfe,tbw=3,padfac=5,ar1=TRUE,genplot = F,output=1, verbose = F, detrend=T)
sig.freq <- astrochron::mtm(dfe,tbw=3, padfac=5,ar1=TRUE,genplot = F,output=2, verbose = F, detrend=T)
mtm.df <- data.frame("freq" = spec.mtm$Frequency, "pwr" = spec.mtm$Power)

# plot this
prange = c(5,1000)

p.mtm <- plotSpectrum(mtm.df,period_range=prange,period_ticks = pticks, ylims = c(1e-6,10)) +
  ggtitle("IODP 846 d18O, Multi-taper method, AR(1) null") + xlab("Period (ky)") +
  theme_hc(style = "darkunica") + theme(axis.ticks.x = element_line(color = "gray"))

# label periodicities of interest
p.mtm <- PeriodAnnotate(p.mtm, periods = c(19,23,41,100), ylims = c(1e-6,1))
p.mtm <- PeriodAnnotate(p.mtm, periods = 1/sig.freq$Frequency,colour = "chartreuse",ylims = c(1e-6,.1))
show(p.mtm)
```



You may notice a few differences to the Lomb-Scargle periodogram. First, the high frequency part is much smoother, getting rid of a lot of high-frequency noise. There is also a clear power law behavior from periods of 5 to 100 ky, which in this log-log plotting convention manifests as a linear decrease. *Astrochron* implements several tests to detect harmonic (sinusoidal) components. Like all tests, they are heavily dependent on a null hypothesis. By default, *astrochron::mtm()* assumes that we test against an AR(1) model. Using the option `output = 2`, it will export the frequencies identified as "significant" by this procedure. In this case, it roughly identifies the Milankovitch frequencies we expected to find, plus many others. What should we make of that? The 100ka cycle is now labeled as 90 ka, though given the peak's width, one should not be foolish enough to consider them distinct. There is also evidence that this peak may be the result of ice ages clustering every 2 to 3 obliquity cycles (81 or 123 ka), averaging around 100 over the pleistocene Huybers & Wunsch (2005). Bottom line: with such spectral resolution, 94 and 100 are one and the same, and may really be a mixture of 80 and 120. Overall, this test identifies 8 periodicities as significant, compared to the 4 we'd expect.

It's helpful to take a step back and contemplate our null hypothesis of AR(1) background, and the real possibility that without adjustments, we might be underestimating the lag-1 autocorrelation, hence making the test too lenient. There are alternatives to that in *Astrochron*, using either the robust method of Mann & Lees (1996) or a less error-prone version by the author Meyers, (2012), called LOWSPEC. You might want to experiment with that.

Another factor to play with is the padding factor (`padfac`). To make a long story short, padding helps increase the spectral resolution at little to no cost (see here for an informal account). You should also try and see what happens when it varies. (note: we used the *Astrochron* default values here)

Most important of all is the choice of null (Vaughan et al, 2011). Past work has shown that the continuum of climate variability is well described by power laws Huybers & Curry (2006), so this should be a far better null against which to test the emergence of spectral peaks. Never fear! *Astrochron* has an app for that:

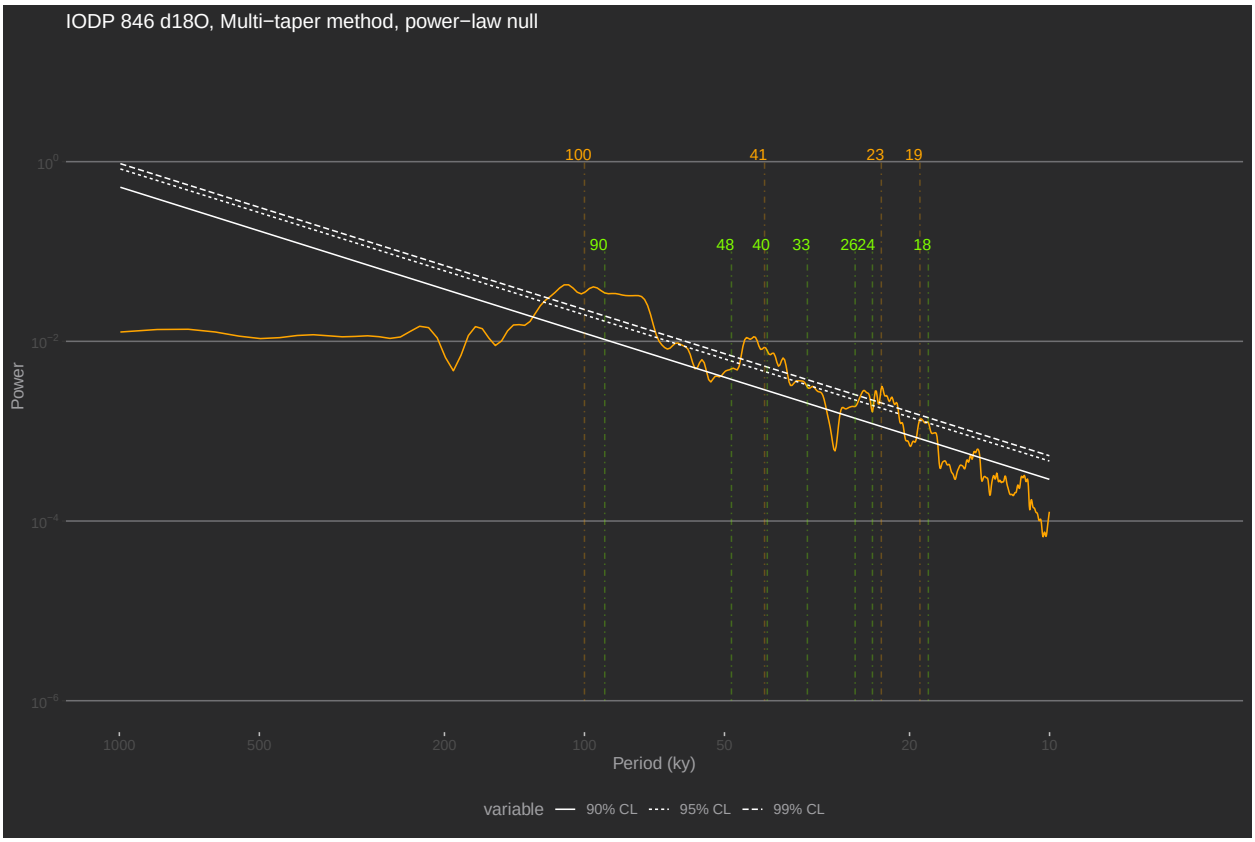
```
mtmPL.df <- astrochron::mtmPL(dfe,tbw=3,padfac=5,genplot = F,output=1, verbose = F, flow=f.low, fhigh=f
sig.freqPL <- astrochron::mtmPL(dfe,tbw=3,padfac=5,genplot = F,output=2, verbose = F, flow=f.low, fhigh=
```

```
mtm.df = mtmPL.df[c(1,2)]
names(mtm.df)[1] <- "freq"
names(mtm.df)[2] <- "pwr"

cl.df <- data.frame(mtmPL.df[,union(1,c(5:7))]) # extract confidence limits
# rename columns to be less silly
names(cl.df)[1] <- "freq"
names(cl.df)[2] <- "90% CL"
names(cl.df)[3] <- "95% CL"
names(cl.df)[4] <- "99% CL"


# plot it
p.mtmPL <- plotSpectrum(mtm.df,cl.df,period_range=prange,period_ticks = pticks, ylims = c(1e-6,10)) +
  ggtitle("IODP 846 d18O, Multi-taper method, power-law null") + xlab("Period (ky)") +
  theme_hc(style = "darkunica") + theme(axis.ticks.x = element_line(color = "gray"))

# label periodicities of interest
p.mtmPL <- PeriodAnnotate(p.mtmPL, periods = c(19,23,41,100), ylims = c(1e-6,1))
p.mtmPL <- PeriodAnnotate(p.mtmPL, periods = 1/sig.freqPL$Frequency,colour = "chartreuse",ylims = c(1e-
show(p.mtmPL)
```



Sure enough, this gets rid of a few cycles, but many remain. Taking the 95% level as a benchmak, we see that a few regions of the spectrum poke above it, but one should not be so foolish as to identify every single periodicity in these ranges to be equally significant. This is related to the multiple hypothesis problem. However, as pointed out by Meyers, (2012), even that misses the point: sedimentary processes (and many processes in other proxy archives) tend to smooth out the signal over the depth axis, making comparisons at

neighboring frequencies highly dependent. One solution is to use predictions made by a physical model about the frequency and relative amplitude of astronomical cycles Meyers & Sageman, 2007. However this may not be applicable to all spectral detection problems. We thus refrain from implementing "cookie-cutter" solutions in GeoChronR, to force the user to think deeply about the null hypothesis and the most sensible way to test it.

Weighted Wavelet Z-transform (*nuspectral*)

An interpolation-free alternative to MTM is the Weighted Wavelet Z-transform of Foster, (1996). The idea of spectral analysis is to decompose the signal on an orthogonal basis of sines and cosines. Data gaps cause this basis to lose its orthogonality, creating energy leakage. WWZ mitigates this problem using an optimization approach. Because it is wavelet based, it does not rely on interpolation or detrending. WWZ was adapted by Kirchner & Neal (2013), who employed basis rotations to mitigate the numerical instability that occurs in pathological cases with the original algorithm. A paleoclimate example of its use is given in Zhu et al (2019)).

The WWZ method has one adjustable parameter, a decay constant that balances the time resolution and frequency resolution of the wavelet analysis. The smaller this constant is, the sharper the peaks.

We wanted to offer GeoChronR users a better alternative to Lom-Scargle, but the WWZ code is relatively complex. We thus chose to incoporate the *nuspectral* method of Mathias et al (2004) based on a similar idea. We stress that it is not equivalent to WWZ . The main difference has to do with the approximation of the mother wavelet by a cubic polynomial with compact support. This speeds up computation at the cost of much accuracy. Mathias et al (2004) also describe implementations of complex wavelets in C (a compiled language, faster with loops), but we were not able to obtain any sensible results with those. If what you want is a

To learn more about *nuspectral* in an ideal setting, see [LINK to nuspectral vignette].

Let us see how it performs in on IODP846.

```
library(nuspectral)
nt = length(t)
tau = seq(min(t),max(t),length = max(nt %/% 2,5))
nuspec <-  nuspectral:::nuwavelet_psd(t,X,sigma=0.01,taus = tau)
nf <- length(nuspec$Frequency)
nus.df <- data.frame("freq" = nuspec$Frequency, "pwr" = nuspec$Power)

# power law fit
plaw.fit = astrochron::pwrLawFit(nus.df, dof = 2, flow = 0.001, fhigh = 0.1, output = 1, genplot = F)
```

```
##
## ----- ESTIMATING Power Law (1/f) Fit -----
##  * Number of frequencies in spectrum: 196
##  * Minimum frequency: 0.001092096
##  * Maximum frequency: 0.2140509
##  * Frequency spacing: 0.001092096
##  * Slope from log(power) vs. log(frequency) fit (m): -0.9326002
##  * Y-intercept from log(power) vs. log(frequency) fit (b): -1.354954
##  * beta = 0.9326002
##  * log(N) = -1.104273
##  * estimated bias = -0.2506816
```

```
cl.df <- data.frame(plaw.fit[,union(1,c(5:7))]) # extract confidence limits
# rename columns to be less silly
names(cl.df)[1] <- "freq"
names(cl.df)[2] <- "90% CL"
```

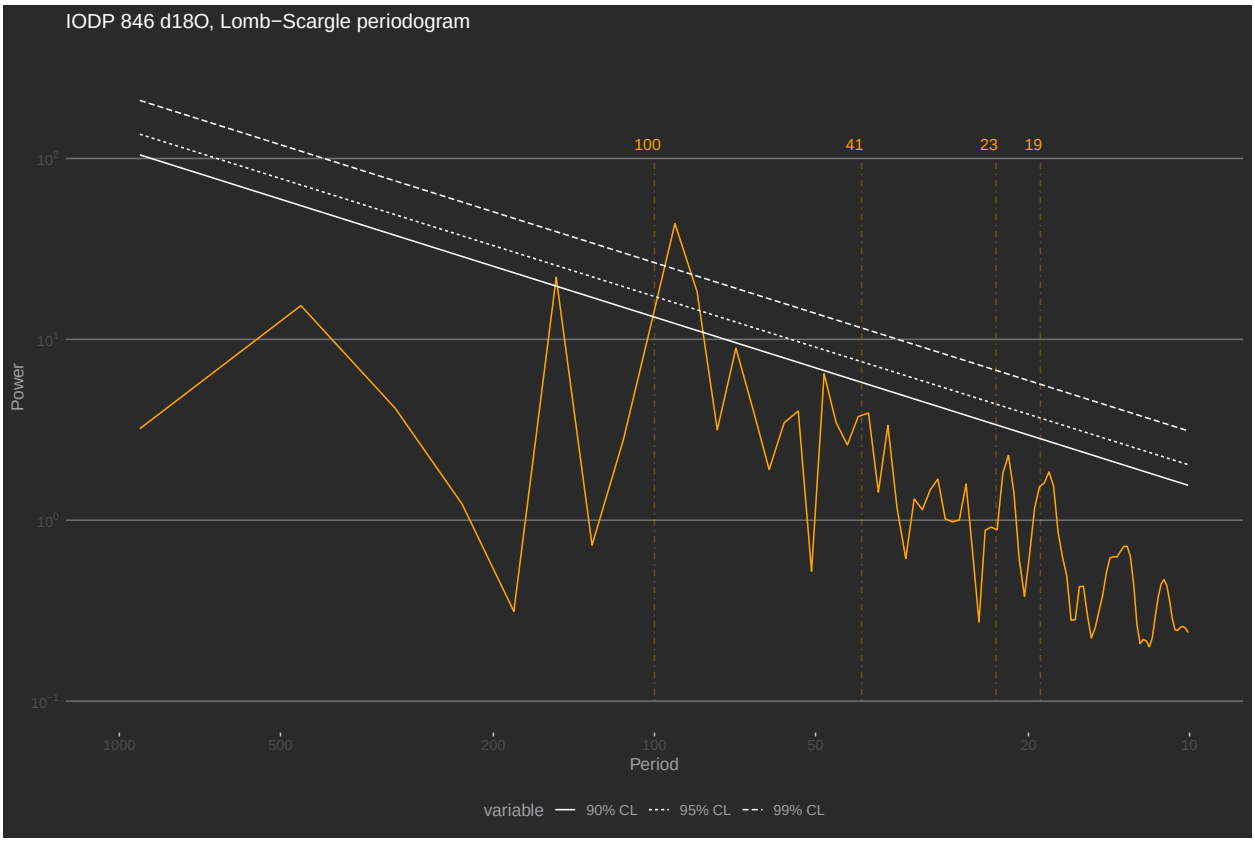
```
names(cl.df)[3] <- "95% CL"
names(cl.df)[4] <- "99% CL"
```

Compared to the previous methods, which were essentially instantaneous on this dataset, this took a few seconds. Keep that in mind for large ensembles, because if you start adding up 1000's of seconds, pretty soon you're waiting for an hour.

As before, we leverage *astrochron*'s capabilities to fit a power-law to the spectrum. The power law comes out with a slope around 1, as before.

```
p.nus <- plotSpectrum(nus.df,cl.df,period_range=c(10,1000),period_ticks = pticks, ylims = c(0.1,300)) +
  ggtitle("IODP 846 d18O, Lomb-Scargle periodogram") +
  theme_hc(style = "darkunica") + theme(axis.ticks.x = element_line(color = "gray"))

# label periodicities of interest
p.nus <- PeriodAnnotate(p.nus, periods = c(19,23,41,100),ylims = c(0.1,100))
show(p.nus)
```



Broadly speaking, we see evidence for the same periodicities, but this took some fiddling with the σ parameter ; the default (0.05) would have overly damped the peaks. We note, by the way, that this default value is not backed by good theoretical arguments or Monte Carlo simulations. That is undoubtedly a weakness of this method. Also notable is that the confidence limits affect peak detection very differently here; only the eccentricity cycle near 100k rises above the 95% and 99% levels; other cycles are summarily dismissed. This could be a result of assuming `dof = 2` at each frequency, which is most likely violated with wavelets. Internally, `nuwavelet_psd` does use variable DOFs, but this is not easy to export. You can tweak the range over which this power law is fit and it will not change much.

Readers interested in a more robust implementation of WWZ should consider Python implementation, available from the Pyleoclim package. Note that this can be done in R via the Reticulate package.

Time-uncertain spectral analysis

Now let's consider age uncertainties.

The GeoChronR approach to pretty much everything (particularly, to quantifying and propagatng uncertainties) is to leverage the power of ensembles. Here we will illustrate this with MTM. Let us repeat the previous analysis by looping over the 1,000 age realizations output by the tuning algorithm HMM-Match

```
time = age$values[age.median < 1000,]  # age ensemble for ~ last 1 Ma
values = temp$values[age.median < 1000]
mtm.ens.spec = geoChronR:::computeSpectraEns(time,values,max.ens = 1000, method = 'mtm',tbw=3, padfac=5)
```

```
##
  |
  |=                                                     |   2%
  |
  |==                                                    |   4%
  |
  |====                                                  |   6%
  |
  |=====                                                 |   8%
  |
  |======                                                |  10%
  |
  |========                                              |  12%
  |
  |=========                                             |  14%
  |
  |==========                                            |  16%
  |
  |============                                          |  18%
  |
  |=============                                         |  20%
  |
  |==============                                        |  22%
  |
  |================                                      |  24%
  |
  |=================                                     |  26%
  |
  |===================                                   |  28%
  |
  |====================                                  |  30%
  |
  |======================                                |  32%
  |
  |=======================                               |  34%
  |
  |========================                              |  36%
  |
  |==========================                            |  38%
  |
  |===========================                           |  40%
  |
  |============================                          |  42%
```

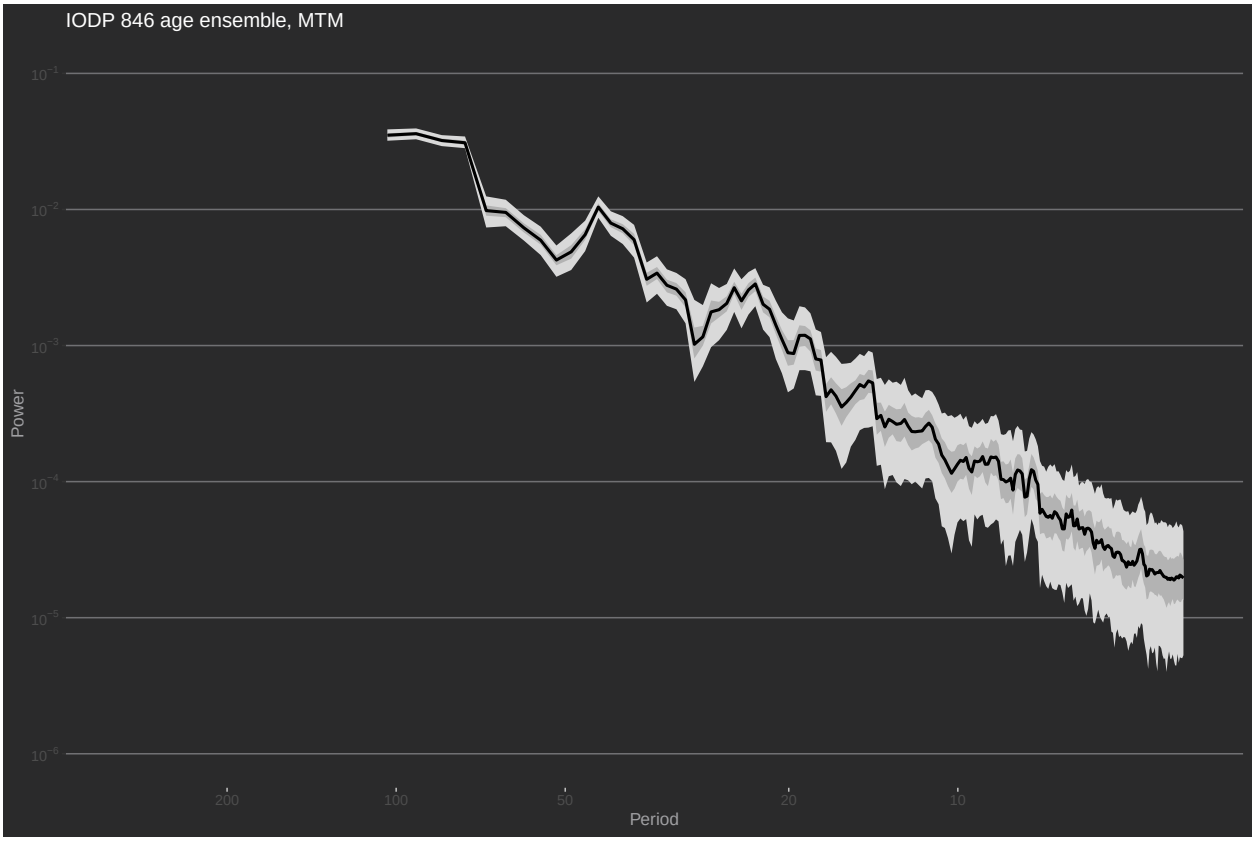
```
|
|============================                                     |  44%
|
|============================                                     |  46%
|
|=============================                                    |  48%
|
|=============================                                    |  50%
|
|==================================                               |  52%
|
|==================================                               |  54%
|
|===================================                              |  56%
|
|===================================                              |  58%
|
|====================================                             |  60%
|
|====================================                             |  62%
|
|=====================================                            |  64%
|
|=====================================                            |  66%
|
|======================================                           |  68%
|
|======================================                           |  70%
|
|=======================================                          |  72%
|
|=======================================                          |  74%
|
|========================================                         |  76%
|
|========================================                         |  78%
|
|=========================================                        |  80%
|
|=========================================                        |  82%
|
|==========================================                       |  84%
|
|==========================================                       |  86%
|
|===========================================                      |  88%
|
|===========================================                      |  90%
|
|============================================                     |  92%
|
|============================================                     |  94%
|
|=============================================                    |  96%
```

```
  |
  |======================================================================== |  98%
  |
  |======================================================================== | 100%
```

This took a minute or so, and the resulting output is close to 40 Mb. Not an issue for modern computers, but you can see why people weren't doing this in the 70's, even if they had wanted to. Now, let's plot the result:

```
p.mtm.ens <- plotSpectraEns(spec.ens = mtm.ens.spec) + ggtitle("IODP 846 age ensemble, MTM") +
  theme_hc(style = "darkunica") + theme(axis.ticks.x = element_line(color = "gray"))

# label periodicities of interest
p.mtm.ens <- PeriodAnnotate(p.mtm.ens, periods = c(19,23,41,100), ylims =c(0.01,100))
show(p.mtm.ens)
```



To conclude, we must emphasize that hunting for astronomical (or other) frequencies in paleoclimate timeseries takes extreme care, and cannot be answered in one function call. In this particular example where the data are nearly evenly spaced, and interpolation has a negligible impact, the best choice is MTM for two reasons: (1) because its minimizes spectral leakage, and (2) because the implementation used here (derived from *astrochron*) allows to flexibly apply different tests of significance. Not such luxury is afforded by the other two methods (Lomb-Scargle, nuspectral) at present, and nuspectral is desperately slow on this dataset.

Depending on their goals, readers might want to consider the work of Stephen Meyers: - Meyers, 2015 - Meyers & Malinverno 2018 - Crampon et al 2018