

Detrending the signal

Similar as in the "traditional approach"

```
library(astrochron)
```

```
## Welcome to astrochron v0.9 (2019-01-08)
```

```
signal_used=read.csv(paste0(DIRECTORY,"Case_3_Signal.csv"))
signal_detrended=lowpass(signal_used, fcut = 1/50, win = 0, addmean = F, genplot = F, verbose = F)
signal_detrended[,2]=signal_used[,2]-signal_detrended[,2]
```

TimeOpt

TimeOpt is an astronomical testing algorithm for depth-domain stratigraphic data. The function identifies the sedimentation rate that simultaneously optimizes eccentricity amplitude modulation of precession, and the concentration of spectral power at the astronomical frequencies (i.e. targetE and targetP). For each sedimentation rate that is investigated, the observed precession band amplitude envelope is extracted using a Taper bandpass filter (settings in flow, fhigh, roll) and the Hilbert transform. The fit of extracted precession envelope with the expected eccentricity periods is evaluated using a linear regression (r²_envelope). The concentration of power at the target astronomical frequencies is evaluated using a linear regression of the time-calibrated stratigraphic series onto sine and cosine terms in targetE and targetP. The fit is estimated by calculation of the correlation of the time-calibrated stratigraphic series to the time-calibrated stratigraphic series (r²_spectral). The final measure of fit (r²_opt) is determined as r²_envelope * r²_spectral.

The statistical significance of the r²_opt is determined via Monte Carlo simulation using timeOptSim. This function performs Monte Carlo AR1 simulations to compare the r²_opt obtained for our stratigraphic data with the distribution of r²_opt results that one obtains by looking at AR1 simulations without an astronomical signal.

```
res1=timeOpt(signal_detrended,sedmin=1,sedmax=10,numsed=250,linLog=1,fit=1,targetE=c(405,125,95),targetP=c(20,17),
,flow=1/24,fhigh=1/14,roll=1000,output=1, verbose = T, genplot = T)
```

```
##
## ----- TimeOpt: Assessment of Amplitude Modulation & Bundling-----
## * Number of data points in stratigraphic series: 2631
## * Stratigraphic series length (meters): 394.5
## * Sampling interval (meters): 0.15
##
## * Linear trend subtracted. m= -5.483685e-05 b= 5.289046
##
## **** WARNING: minimum sedimentation rate is too low for full signal recovery.
##      sedmin reset to 2.142857 cm/ka
##
##
## * PLEASE WAIT: Performing Optimization
##
## 0%    25%    50%    75%    100%
## =====
##
## * Maximum (spectral power r^2)= 0.04690877 at sedimentation rate of 4.474241 cm/ka
## * Maximum (envelope r^2)= 0.09247067 at sedimentation rate of 8.891014 cm/ka
## * Maximum (envelope r^2) x (spectral power r^2) = 0.00351658 at sedimentation rate of 4.474241 cm/ka
```

```
res2=timeOpt(signal_detrended,sedmin=1,sedmax=10,numsed=250,linLog=1,fit=1,targetE=c(405,125,95),targetP=c(20,17),
,flow=1/24,fhigh=1/14,roll=1000,output=2, verbose = F, genplot = F)
```

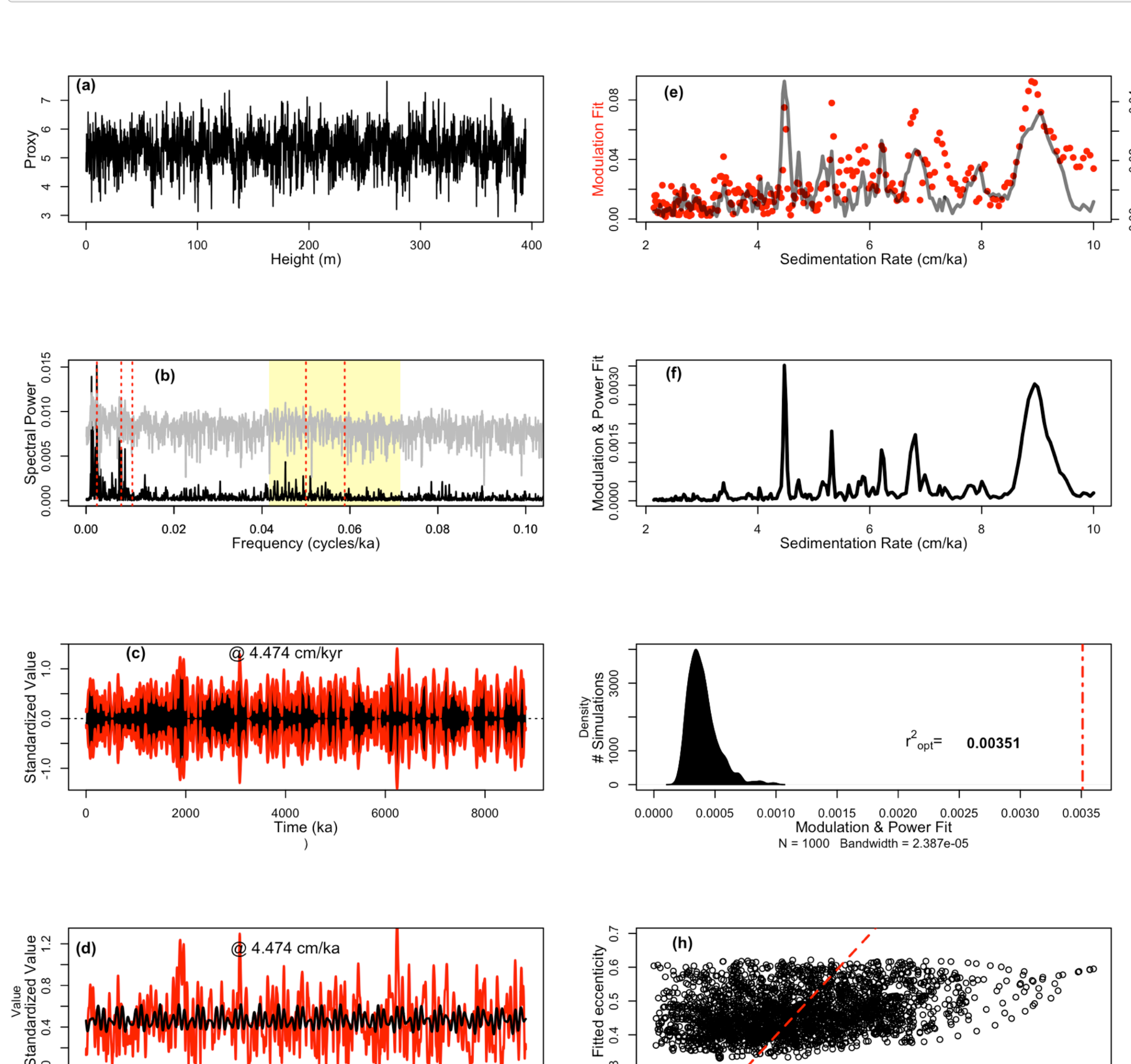
```
# We arrive to an optimal sedimentation rate of 4.474 cm/kyr. The next line of code carries out a Monte Carlo
simres=timeOptSim(signal_detrended,sedrate=4.474241,numsim = 1000, fit=1, targetE=c(405,125,95), targetP=c(20,17),
,flow=1/24, fhigh=1/14, roll=1000, output=2, genplot = F, verbose = F)
```

Plotting timeOpt results

This chunk of code creates a figure that is similar to Figures 3 and 5 in Meyers (2015). In more recent versions of astrochron, there is a built-in function to make such a Figure:

```
timeOptPlot(dat = signal_detrended, res1 = res1, res2 = res2, simres = simres, fit = 1, targetE=c(405,125,95), targetP=c(20,17), flow=1/24,
fhigh=1/14, roll=1000)
```

```
par(mfrow=c(4,2))
# plot 1: (A)
plot(signal_detrended[,1], signal_detrended[,2], col = "black", xlab = "", ylab = "", main = "", type="l", lwd=1.2)
mtext("Proxy",side=2,line=2,cex=0.8)
mtext("Height (m)",side=1,line=2,cex=0.8)
text(0,7.5,"(a)",cex=1.3,font=2)
# plot 2: (B)
plot(res1[, 1], res1[, 2], cex = 1.2, col = "red", xlab = "", ylab = "", main = "",pch=16)
text(2.5,0.085,"(e)",cex=1.3,font=2)
par(new = T)
plot(res1[, 1], res1[, 3], col = "#00000096", xlab = "", ylab = "", type = "l", axes = F, lwd = 3)
axis(4, ylim = c(0, max(res1[, 3])), lwd = 1, col = "black")
mtext("Modulation Fit",side=2,line=2,cex=0.8,col="red")
mtext("Sedimentation Rate (cm/ka)",side=1,line=2,cex=0.8)
mtext("Spectral Power Fit",side=4,line=2,cex=0.8)
# plot 3: (B)
fft = periodogram(data.frame(cbind(res2[, 1], res2[, 2])),output = 1, verbose = F, genplot = F)
fft = subset(fft, (fft[, 1] > 0))
plot(fft[, 1], fft[, 3], xlim = c(0, 0.1), type = "l",lwd=0, xlab = "", ylab = "", main = "")
rect(xleft=1/24,ybottom=-.005,xright=1/14,ytop=0.3,col="#FFFF00",border=NA)
par(new = TRUE)
plot(fft[, 1], fft[, 3], xlim = c(0, 0.1), type = "l",lwd=1.5, xlab = "", ylab = "", main = "",axes = F)
text(0.018,0.014,"(b)",cex=1.3,font=2)
par(new = TRUE)
plot(fft[, 1], log(fft[, 3]), xlim = c(0, 0.1), ylim=c(-17.1,-1),type = "l",lwd=1.5, yaxt = "n", col = "gray", xl
ab = "", ylab = "")
targetTot=c(405,125,95, 20, 17)
for (i in 1:length(targetTot)) {
  abline(v = 1/targetTot[i], col = "red", lty = 3,lwd=1.5)
}
mtext("Spectral Power",side=2,line=2,cex=0.8)
mtext("Frequency (cycles/ka)",side=1,line=2,cex=0.8)
# plot 4: (F)
plot(res1[, 1], res1[, 4], type="l", lwd=3, col = "black", xlab = "", ylab = "", main = "")
mtext("Modulation & Power Fit",side=2,line=2,cex=0.8)
mtext("Sedimentation Rate (cm/ka)",side=1,line=2,cex=0.8)
text(2.5,0.003,"(f)",cex=1.3,font=2)
# plot 5: (C)
plot(res2[,1], res2[,3], col = "black", cex = 0.5, xlab = "", ylab = "", main = "",type="l")
lines(res2[,1], res2[,4], col = "red",lwd=2)
lines(res2[,1], -1 * res2[,4], col = "red",lwd=2)
abline(h = 0, col = "black", lty = 3)
mtext("Standardized Value",side=2,line=2,cex=0.8)
mtext("Time (ka)",side=1,line=2,cex=0.8)
text(1000,1.3,"(c)",cex=1.3,font=2)
text(4000,1.3,"@ 4.474 cm/kyr",cex=1.3)
# plot 6: (G)
plot(density(simres$simres),main="",xlim=c(0,0.0036))
polygon(density(simres$simres),col="black")
abline(v=0.00351, lwd=2, lty=4,col="red")
mtext("# Simulations",side=2,line=2,cex=0.8)
mtext("Modulation & Power Fit",side=1,line=2,cex=0.8)
text(0.0005,6000,"(g)",cex=1.3,font=2)
text(0.002,1200,expression(paste("r^2_opt","[opt]","=")),cex=1.3,font=2,pos=4)
text(0.0025,1200,"0.00351",cex=1.2,font=2,pos=4)
# plot 7: (D)
plot(res2[,1],res2[,4], cex = 0.5, xlab = "Time (ka)", ylab = "Value", main = "", col = "red", type = "l",lwd=2,y
lim=c(0,1.3))
lines(res2[,1], res2[,5],lwd=2)
mtext("Standardized Value",side=2,line=2,cex=0.8)
mtext("Time (ka)",side=1,line=2,cex=0.8)
text(0,1.15,"(d)",cex=1.3,font=2)
text(4000,1.15,"@ 4.474 cm/ka",cex=1.3)
# plot 8: (E)
plot(res2[,4], res2[,5], main = "", xlab = "", ylab = "",ylim=c(0.3,0.7))
mtext("Fitted eccentricity",side=2,line=2,cex=0.8)
mtext("Data precession envelope",side=1,line=2,cex=0.8)
text(0,1,0.67,"(h)",cex=1.3,font=2)
abline(0,1,lty=2,lwd=2,col="red")
```



etimeOpt

etimeOpt applies the timeOpt technique in a sliding-window approach. It is useful to detect changing sedimentation rates throughout the record. However, in this case, etimeOpt is not able to resolve the oscillating sedimentation rate between 3 and 6 cm/kyr because this oscillation occurs with a temporal periodicity of 1200 kyr, which corresponds to a spatial periodicity of ~55 meters. This is about as wide as the size of one analysis window, which was chosen to be 50 meters so to resolve the 405-kyr cycle even at sedimentation rates up to 10 cm/kyr.

With the function "tracePeak", one can click on the optimal r²_opt values throughout the stratigraphic column to select optimal sedimentation rates, and reconstruct the evolution of sedimentation rate changes. This function is here commented out, because it requires user-action. As a work-around, a matrix is provided that reflects a possible tracing result.

This tracing result suggests sedimentation rates (4.5 - 5 cm/kyr) that are slightly higher than the average sedimentation rate that was used to build Case 3 (4.5 cm/kyr). Hence, in this case etimeOpt is very useful to get a first depth-to-time transition but provides a total duration that is too short. To come to the correct solution, one can filter out the 405-kyr variability from the initial time-calibrated signal, compare it to the raw proxy data, and do a 405-kyr cycle-counting. This approach would lead to the correct solution (i.e. 21.5 long 405-kyr eccentricity cycles in the black filtered signal in the last figure).

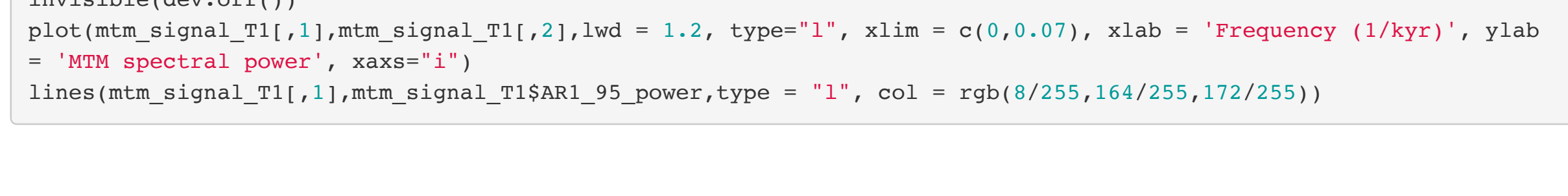
```
res3=eTimeOpt(signal_detrended,win=50,step=15,sedmin = 1, sedmax = 10, numsed = 100, linLog = 1, fit=1, targetE=c(405,125,95),
targetP=c(20,17), flow=1/24, fhigh=1/14, roll=1000, output=1, fitModPwr = F)
```

```
##
## ----- PERFORMING EVOLUTIVE TimeOpt ANALYSIS -----
## * Number of data points in stratigraphic series 1: 2631
## * Stratigraphic series length (space or time): 394.5
## * Sampling interval (space or time): 0.15
##
## * Number of data points per window: 334
## * Moving window size (space or time): 49.95
## * Window step points: 100
## * Window step (space or time): 15
## * Number of windows: 23
##
## * PLEASE WAIT: Performing Optimizations
##
## 0%    25%    50%    75%    100%
## =====
```

```
#sedrates=tracePeak(res3[3])
sedrates=matrix(nrow = 11, ncol = 2, data = c(27,69,99,114,134,189,221,237,266,312,346,4.83,4.97,5.16,5.22,5.20,5
.18,4.9,4.86,5.44,5.45,5.37))
agemodel=sedrate2time(sedrates, verbose = F)
```

```
signal_T1_detrended=tune(signal_detrended, agemodel, extrapolate = T, genplot = F, verbose = F)
signal_T1_detrended[,1]=signal_T1_detrended[,1]
plot(signal_T1_detrended,type="l")
axis(3, at = c(0,2000,4000,6000),signal_T1_detrended[length(signal_T1_detrended[,1]),1]))
```

```
signal_T1_detrended=linterp(signal_T1_detrended, verbose = F, genplot = F)
mtm_signal_T1=mtm(signal_T1_detrended,demean = T, ar1 = T, tbw = 2,output = 1, xmax = 0.07, genplot = F, verbose = F)
invisible(dev.off())
plot(mtm_signal_T1[,1],mtm_signal_T1[,2],lwd = 1.2, type="l", xlim = c(0,0.07), xlab = 'Frequency (1/kyr)', ylab = 'MTM spectral power',
xaxs="l")
lines(mtm_signal_T1[,1],mtm_signal_T1$AR1_95_power,type = "l", col = rgb(8/255,164/255,172/255))
```



```
filter_405=bandpass(signal_T1_detrended, flow = 1/440, fhigh = 1/300, xmax = 0.01, genplot = F, verbose = F)
plot(signal_T1_detrended,type="l", col = "grey", xlab = "Relative Time (kyr)", ylab = "Virtual proxy", xaxs = "i",
ylim = c(3,7.5), yaxs = "i")
lines(filter_405, col = "black", lwd = 2)
```

