Entire Range Test Complexity

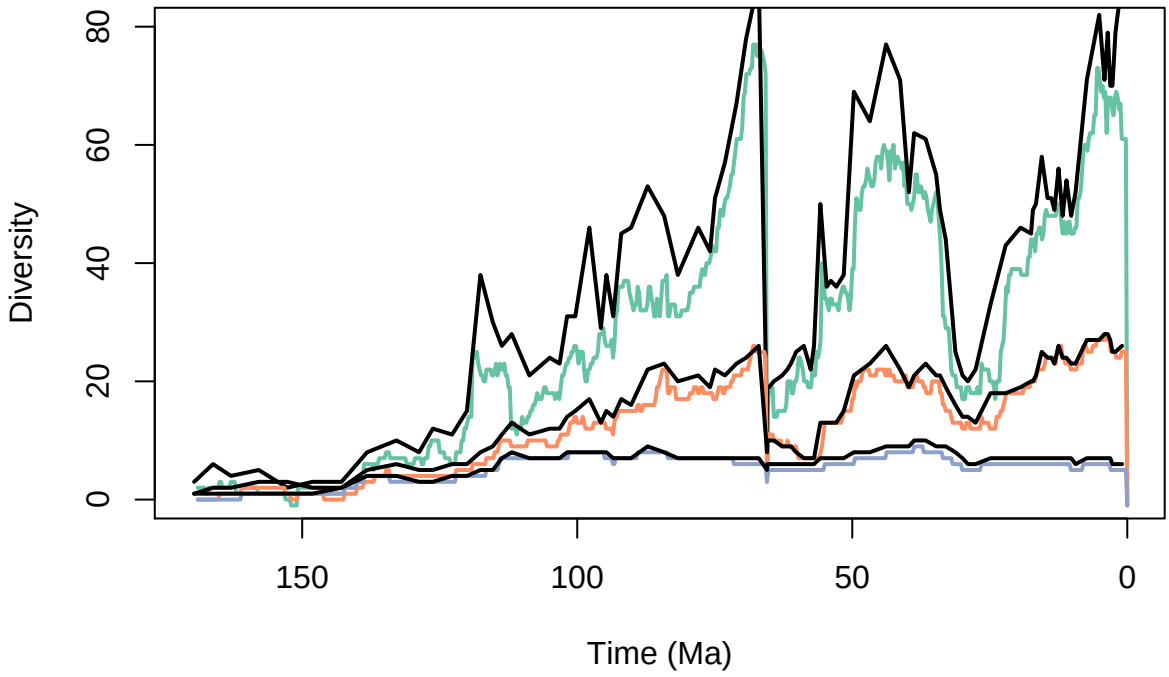
Fraass

7/4/2018

Several code-chunks below execute custom functions not displayed in this brief walkthrough of the calculations. All that code and the .r file used to generate this file avaliable on github.com/Fraass/Test-Complexity.

Initial Dataset

```
plot(0,0,
     type='n',
     xlim=c(170,0),
     #xlim=c(80,50),
     ylim=c(0,
            80),
     xlab='Time (Ma)',
     ylab="Diversity")
#axis(side=2,at=seq(0,800,by=5))
brewer.pal(n=8,name="Set2")->div.pallet
lines(time.div(test.complex.index,
               morph$origin,
               morph$extin,
               0.25),
      lwd=2,
      col=div.pallet[1])
lines(midstage,occurtot,col='black',lwd=2)
lines(time.div(1:length(genus.dataframe$genus),
               genus.dataframe$origin,
               genus.dataframe$extin,
               0.25),
      lwd=2,
      col=div.pallet[2])
lines(midstage,occurtot.genus,col='black',lwd=2)
lines(time.div(1:length(family.dataframe$genus),
               family.dataframe$origin,
               family.dataframe$extin,
               0.25),
      lwd=2,
      col=div.pallet[3])
lines(midstage,occurtot.family,col="black",lwd=2)
```

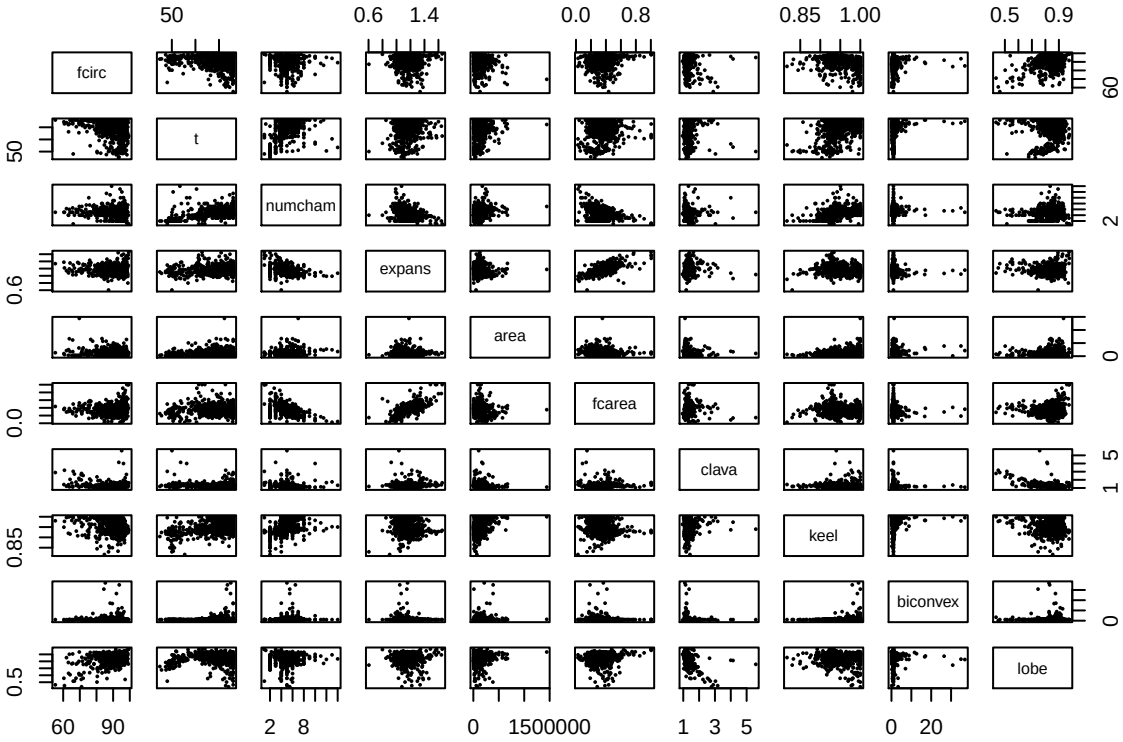


The difference between the 0.25 myr binned and zone scheme binned are pretty stark. In most timeperiods the zonal scheme inflates the number of individuals. That's not surprising. There are a few times that isn't the case, the later part of the Cretaceous, the Danian, the latter stages of the Paleogene, and the later portion of the Miocene. This is much less the case for the genus, which appears much more stable. While this seems obvious, it's an important difference, and suggests that binning schemes are much more important in the lower taxonomic levels.

A case certainly should be made that there are instances (the Danian here, for example) where the additional resolution is useful because of the increased amount of work devoted to understanding the recovery. Where doing a longer record however, the zone scheme is appropriate.

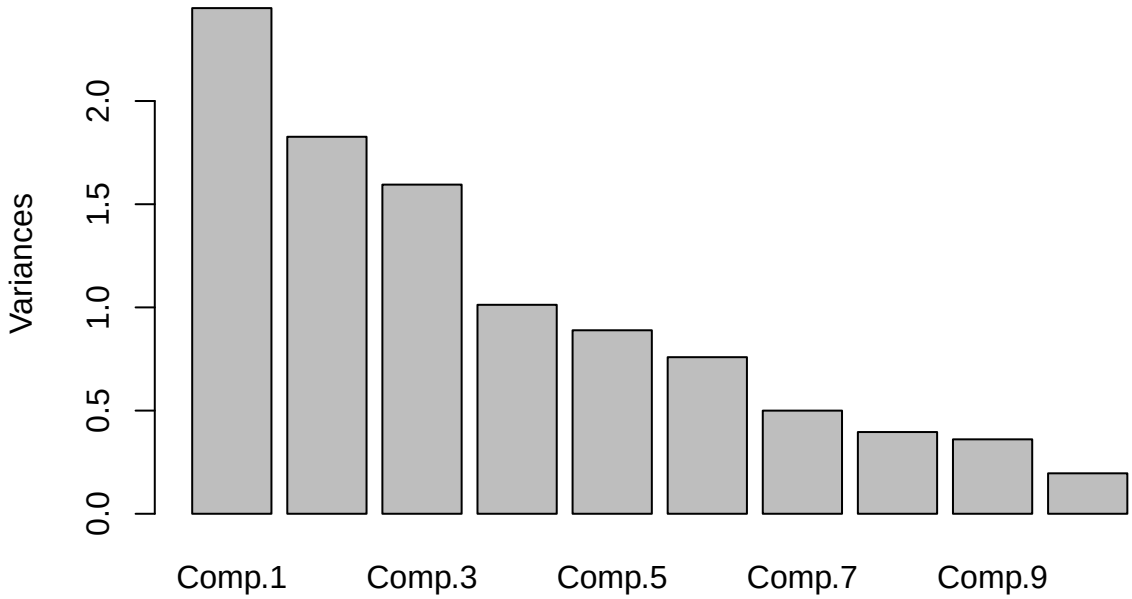
```
### This is the full list of characters and the
inc<-c(#"w",
  #"mtheta",
  #"lw",
  #"lh",
  #"ic1",
  #"ic2",
  #"ic3",
  "fcirc",
  "t",
  "numcham",
  "expans",
  #"height",
  #"length",
  #"fcangle",
  "area",
  "fcarea",
  "clava",
  #"chamwl",
  "keel",
  #"bidors",
  #"biven",
  "biconvex",
```

```
  "lobe"
  #"double"
  #"depth"
)
inc2<-c(#"w",
  "mtheta",
  "lw",
  "lh",
  #"ic1",
  #"ic2",
  #"ic3",
  "fcirc",
  "t",
  "numcham",
  "expans",
  #"height",
  #"length",
  "fcangle",
  "area",
  #"fcarea",
  "clava",
  "chamwl",
  "keel",
  #"bidors",
  #"biven",
  "biconvex",
  "lobe",
  "double"
  #"depth"
)
pairs(morph[,inc],pch=16,cex=.4)
```



Plot of the included morphometric parameters. `inc` is the set of morphological parameters used to for this study, while `inc2` is the complete dataset. In `inc2` several parameters are `#`'d out, those are used in calculations of other parameters. I use it for the PCA, along with `inc`.

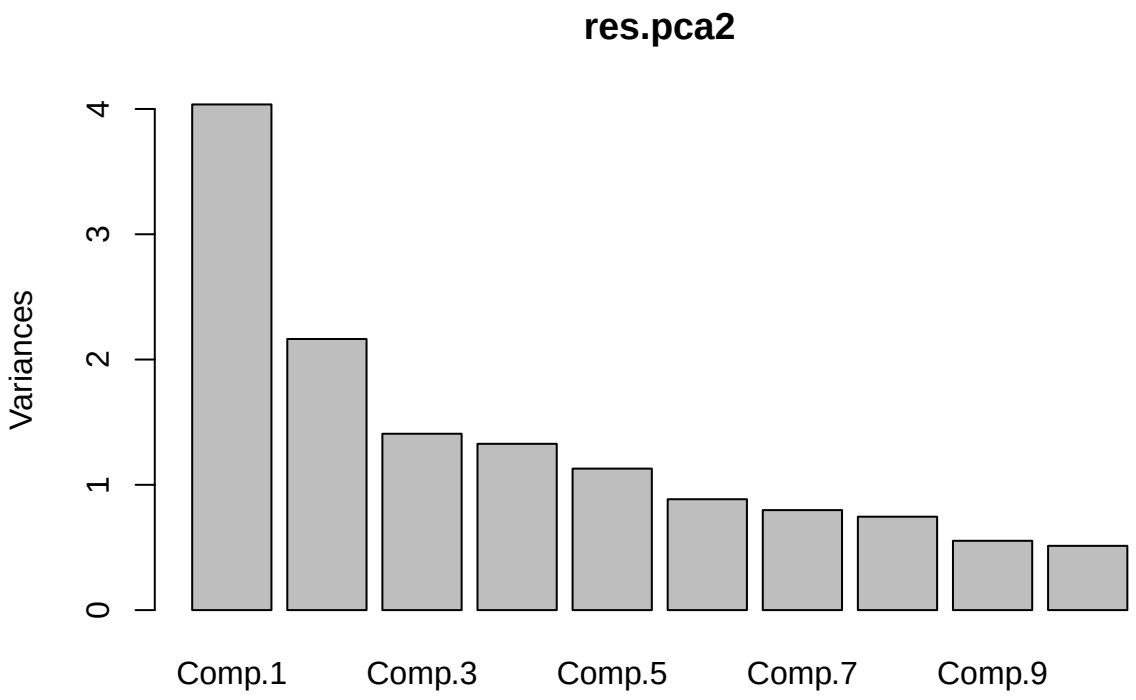
```
foram.dataframe->morph
exclude<-c(4) #this is the only set of measurements that is incomplete.
morph[1:length(morph[,1]) %w/o% exclude,]->morph
morph[which(is.na(rowSums(morph[,inc])) == F),13:36]->pca.in
decostand(pca.in[,inc],method='standardize',2)->pca.in
princomp(pca.in)->res.pca
plot(res.pca)
```




```
res.pca$sdev^2/sum(res.pca$sdev^2)->PoV
```

Results of the Principle Components Analysis (PCA). First, second, and third axes contain ~58.81% of the variance.

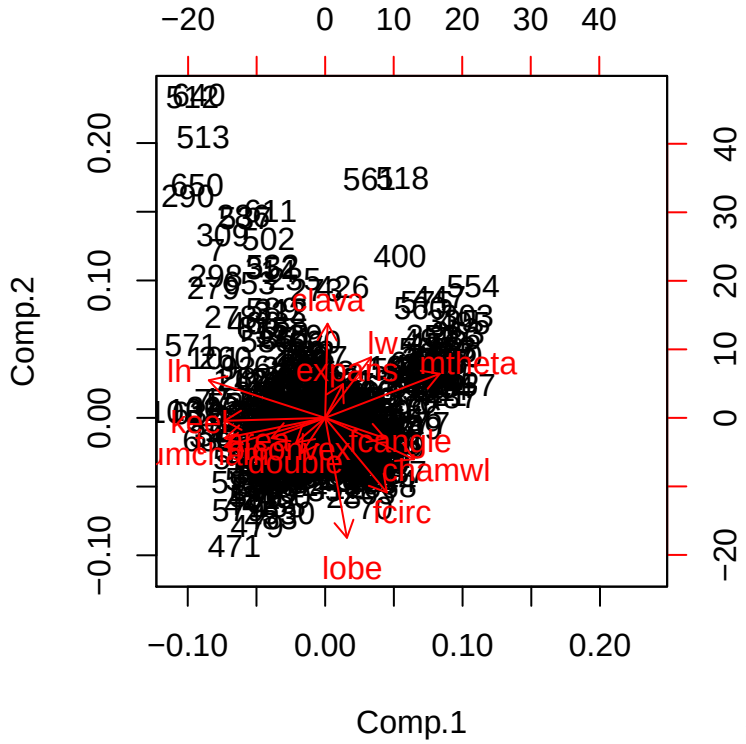
```
foram.dataframe->morph
exclude<-c(4)
morph[1:length(morph[,1]) %w/o% exclude,]->morph
morph[which(is.na(rowSums(morph[,inc2])) == F),13:36]->pca.in
decostand(pca.in[,inc2],method='standardize',2)->pca.in
princomp(pca.in)->res.pca2
plot(res.pca2)
```




```
res.pca2$sdev^2/sum(res.pca2$sdev^2)->PoV2
```

Results of a second Principle Components Analysis (PCA) which includes all morphometric parameters. First, second, and third axes contain ~50.8% of the variance.

```
biplot(res.pca2)
```



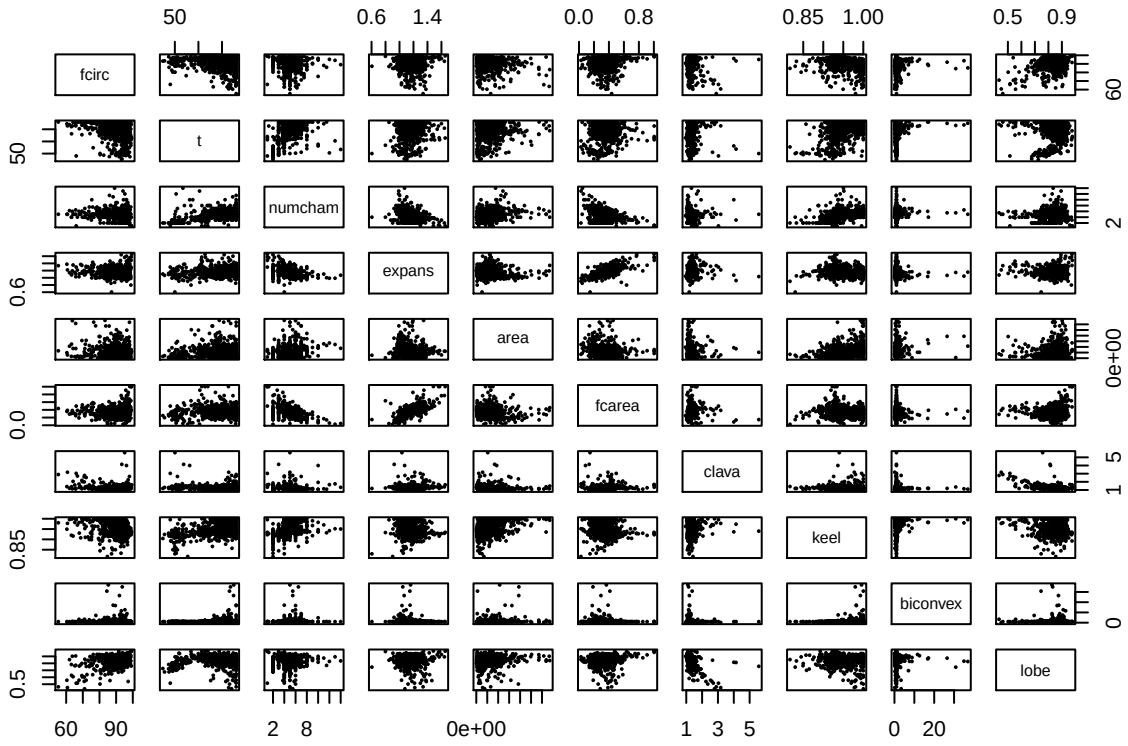
So, including targeted set lets us look at fewer components, with higher variance explained. The number of chamber parameters seem to dominate the first axis in the second kitchen-sink approach. That makes sense, since the number of chambers in the final whorl is dependent on the interchamber angle, which is then summarized by mtheta. So, mtheta, ic1:5 are all

covarying, while number of chamber is placed at the other side. Thus, multiple parameters are explaining the same property, and thus the 'kitchen-sink' approach is obviously fundamentally flawed. Axis two is size, (theres also some minor loading of size on A1). Essentially in the 'kitchen-sink' approach the analysis picks out big keeled things (ie., *Globorotalids*) v. others.

A primary concern is the lack of differentiation on the skree-plot in the 'focused' dataset. That dataset has variance spread out through the various axes. All of our measurements were designed to discriminate between the major groups, and be highly varying. Thus, because of the nature of our measruermnets the variance should be distributed because we did that on purpose.

Test Complexity

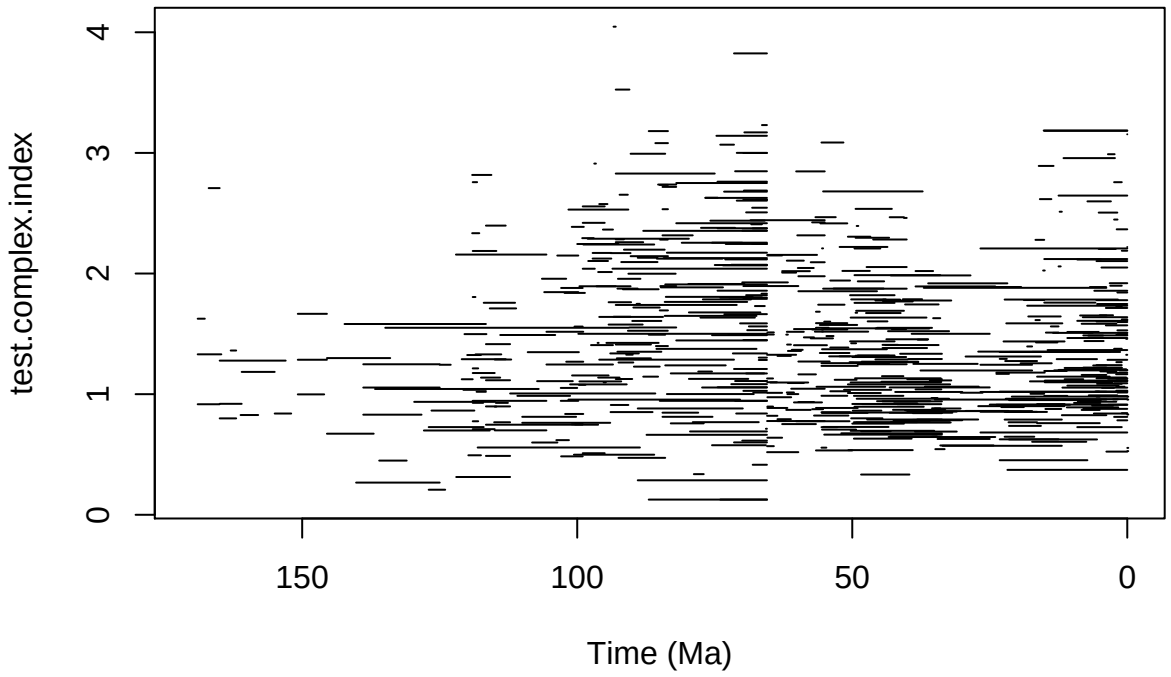
```
inc<-c(#"w",
  #"mtheta",
  #"lw",
  #"lh",
  #"ic1",
  #"ic2",
  #"ic3",
  "fcirc",
  "t",
  "numcham",
  "expans",
  #"height",
  #"length",
  #"fcangle",
  "area",
  "fcarea",
  "clava",
  #"chamwl",
  "keel",
  #"bidors",
  #"biven",
  "biconvex",
  "lobe"
  #"double"
  #"depth"
)
pairs(morph[,inc],pch=16,cex=.4)
```




```
which(morph$species == "holmdelensis")->h
which(morph$species == "monmouthensis")->m
build.tci<-morph
for(i in inc){
  {morph[h,i]+morph[m,i]}/2->simple.value
  build.tci[,i]<-morph[,i]-simple.value
}
```

Generating the predefined `simple.value` to assign as our 0 for each parameter.

```
for(i in inc){
  abs(build.tci[,i])->build.tci[,i]#taking the absolute value
  decostand(build.tci[,i],method='range',na.rm=TRUE)->build.tci[,i]
}
test.complex.index<-NA
for(i in 1:length(build.tci$species)){
  sum(build.tci[i,inc])->test.complex.index[i]
}
plot(0,0,
     type='n',
     xlim=c(170,0),
     ylim=c(min(test.complex.index,na.rm=T),
            max(test.complex.index,na.rm=T)),
     xlab='Time (Ma)',
     ylab="test.complex.index")
segments(build.tci$origin,
         test.complex.index,
         build.tci$extin,
         test.complex.index)
```

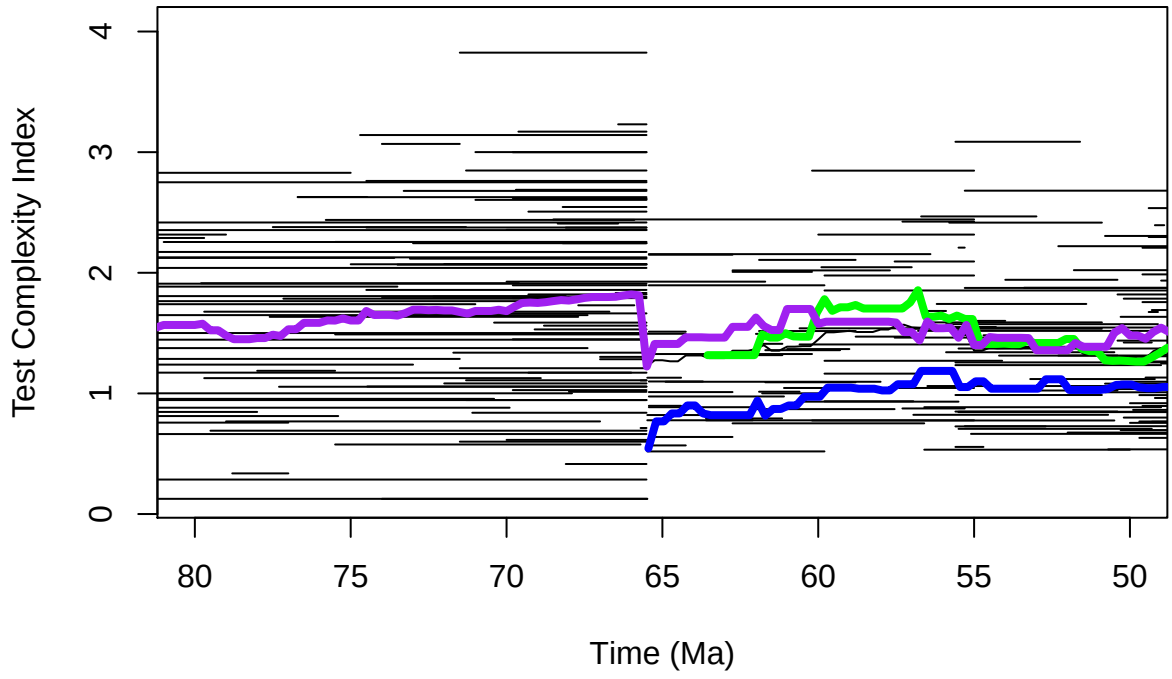



```
#highlighting significant macroevolutionary events
which(timeresolution.dataframe$sig.origin == 'y')->sig.origin
which(timeresolution.dataframe$sig.extin == 'y')->sig.extin
```

Test complexity is calculated by subtracting the predefined simple morphologies from all parameters. The absolute value is then taken, so that lowest value is now the 'simple morphologies' and divergence on either side is an increase. This is key, while several parameters are an increase in complexity (larger area is generally considered a 'more complex' trait in literature), a parameter like lobateness or clavateness are more complex in both directions. The 'simple' foraminifera has a globulose appearace, where the more complex forms are both digitate foraminifera and foraminifera with highly circular outlines. The mean, nor median, aren't the 'simple' morphology either, so this more arbitrary designation is a compromise.

```
photo.gen<-c('Morozovella',
             'Igorina',
             'Acarinina')
which(is.na(match(morph$genus,photo.gen))==FALSE)->photo.ind
c(photo.ind,which(morph$species == 'inconstans'))->photo.ind
#Spinose
spin.ind<-c("Globoturbotrotalia","Globorotalites","Catapsydrax","Paragloborotalia",
            "Subbotina", "Parasubbotina","Eoglobigerina")
which(is.na(match(morph$genus,spin.ind))==FALSE)->spin.ind
plot(0,0,
     type='n',

     #xlim=c(170,0),
     xlim=c(80,50),
     ylim=c(min(test.complex.index,na.rm=T),
            max(test.complex.index,na.rm=T)),
     xlab='Time (Ma)',
     ylab="Test Complexity Index")
segments(morph$origin,
         test.complex.index,
         morph$extin,
```

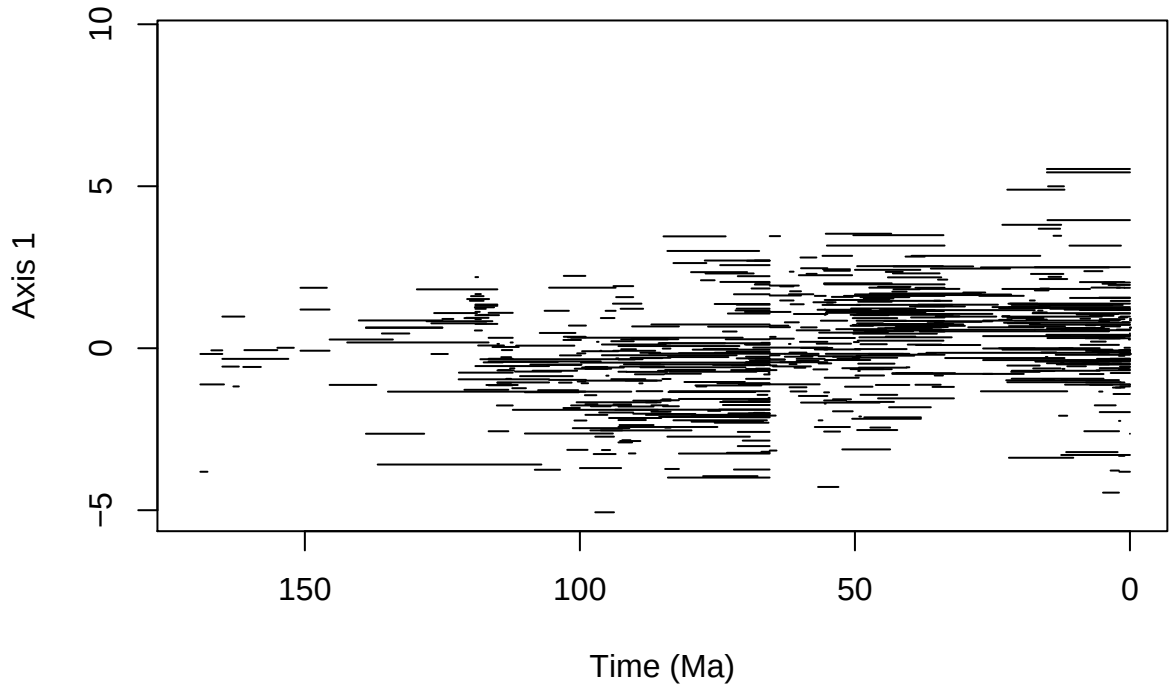
```
         test.complex.index)
#lines(time.mean(test.complex.index,morph$origin,morph$extin,0.25),lwd=2)
#[c(spin.ind,photo.ind)],

lines(time.mean(test.complex.index,morph$origin,morph$extin,0.25))
lines(time.mean(test.complex.index[photo.ind],
                morph$origin[photo.ind],
                morph$extin[photo.ind],0.25),
      lwd=4,
      col='green')
lines(time.mean(test.complex.index[spin.ind],
                morph$origin[spin.ind],
                morph$extin[spin.ind],0.25),
      lwd=4,
      col='blue')
lines(time.mean(test.complex.index[1:length(test.complex.index) %w/o% c(spin.ind,photo.ind)],
                morph$origin[1:length(test.complex.index) %w/o% c(spin.ind,photo.ind)],
                morph$extin[1:length(test.complex.index) %w/o% c(spin.ind,photo.ind)],0.25),
      lwd=4,
      col='purple')
```



Break down of trophic levels with running means.The `time.mean` function was written specifically for this type of analysis, and is just a basic running mean that calculates based on the first and last occurances, with a assignable resolution.

```
axis.of.interest<-1
plot(0,0,
     type='n',
     xlim=c(170,0),
     #xlim=c(80,50),
     ylim=c(min(res.pca$scores[,axis.of.interest],na.rm=T),
            max(res.pca$scores[,axis.of.interest],na.rm=T)+4),
     xlab='Time (Ma)',
```

```
     ylab="Axis 1")
segments(morph$origin,
         res.pca$scores[,axis.of.interest],
         morph$extin,
         res.pca$scores[,axis.of.interest])
```



This is striklingly similar to the plots of test complexity without the 'simple' values assigned or the absolute value taken. It is still possible to see roughly the same shape, and view the rough position of the mean values. There is a switch during the K/Pg where there's slightly higher values through the Cenozoic than the Cretaceous, but remember, this adds values that are complex in the positive direction to values that are complex in the negative direction.