Proposta C

Lei de Cope simulada por random walk

A função final a seguir demostra um princípio muito interessante: linhagens que surgem muito próximas do valor mínimo de um atributo (no exemplo de Gould é o tamanho da “concha” de foraminíferos), só tem uma direção possível mudança. No exemplo, se as espécies iniciais possuem um tamanho muito próximo do mínimo permitido por fatores físicos, a única direção possível é o aumento. Se as espécies tem a mesma chance de crescer ou diminuir, essa barreira de tamanho mínimo sempre reflete as espécies novas para tamanhos maiores. No entanto, isso não significa que há uma tendência evolutiva. Não há nenhuma pressão adaptativa que favoreça tamanhos maiores (pode ver o código!), mas mesmo assim a média dos tamanhos aumenta (simule: fullhouse(nlin=5,t=20,leftwall=0.15,isize=0.2)). O efeito é meramente uma consequência do aumento da variância nos tamanhos, enviesada por alguns poucos tamanhos maiores. Comprove isso observando a moda dos valores ao longo do tempo e o gráfico dos tamanhos mínimos. Por outro lado, se as espécies iniciais começam longe da barreira(fullhouse(nlin=5,t=20,leftwall=0.15,isize=1)) , o modelo prevê um aumento de variância simétrica ao redor do tamanho inicial: tamanhos maiores aumentando, tamanhos menores diminuindo e uma média mais estável. O mais interessante é que o princípio é geral. Se o atributo em questão for a complexidade iniciada próxima da barreira mínima, esperaríamos um aumento na complexidade (variância no sistema), sem que houvesse nenhuma tendência evolutiva para aumento de complexidade. Esperaríamos também que essa forma menos complexa se mantivesse como a mais comum e abundante, mesmo com a enviesada cauda a direita da distribuição, muito complexa, mas pouco comum... O sistemas devem portanto serem visto como uma figura completa (“Full House”), e não em função de uma minúscula ponta de cauda, onde se encontram nossas cabeças pensantes.

“(...) once only bacteria, but now petunias and people as well” Stephen Jay Gould, in Full House, p146.

Alguma bibliografia sobre o assunto:

Gould, S. J. 1988. Trends as changes in variance: a new slant on progress and directionality in evolution. Journal of Paleontology, 62(3):319-29.

Gould, S. J. 1996. Full House: The Spread of Excellence from Plato to Darwin. New York, NY. Three Rivers Press.

Stanley, S. M. 1973. An explanation for Cope's rule. Evolution, 27:1-26.

```
##Full House: Cope's Law com random walk, uma simulação
#Versão 1.0.0

#: Cria a função fullhouse, com os argumentos:
#nlin:Número de espécies iniciais; isize:Tamanho inicial das espécies;
x:Amplitude de variação de tamanho corpóreo;t: Intervalos de tempo;
leftwall: tamanho corpóreo mínimo.
fullhouse <- function(nlin, isize, x, t, leftwall){

#Determina os valores default para cada parâmentro.
  if(missing(nlin)){nlin <- 5}
  if(missing(isize)){isize <- 0.2}
  if(missing(x)){x <- 0.1 }
  if(missing(t)){t <- 20}
```

```
  if(missing(leftwall)){ leftwall <- 0.15}
#: Cria o vetor númerico "especie", contendo os nomes das espécies iniciais
de 1 até o nlin escolhido.
especie <- (1:nlin)

#: Cria o vetor "tamanho", contendo um tamanho inicial (isize) para cada
espécie (nlin).
tamanho <- rep(isize,nlin)

#: Cria o factor "tempo", que marca tempo 0 para todas as nlin espécies
iniciais. Cria todos os levels
#possíveis para esse vetor, para evitar problemas futuros na inclusão de
valores que não estavam no vetor previamente.
tempo <- factor(x=rep(0,nlin),levels=-1:t+1)

#: Cria o dataframe "fossil", que registra o tamanho de cada espécie num
dado tempo
fossil <- data.frame(especie, tamanho, tempo) #Cria o dataframe fossil, que
registra o tamanho de cada espécie num dado tempo

#: Cria um registro das espécies, que permitirá acertar os nomes de espécies
novas que forem criadas na simulação.
lista_especie <- unique(fossil[,1]) #

#: Cria um vetor com todos os tempos, que será usado na plotagem de
gráficos.
Tempo <- seq(0,t+1)

#:Abre o primeiro ciclo: cada t novo é um novo ciclo de simulação.
for (t in 0:t){

#: Abre o segundo ciclo: para um dado tempo, todas as espécies sao
submetidas ao tratamento abaixo:
  for (sp in fossil[fossil[,3] == t, 1] ){

#:Atribui probabilidades aos eventos 0, 1 e 2, e sorteia um desses valores.
    moeda1 <- sample(0:2,size=1,prob=c(0.69,0.29,0.02))
#: Se o valor sorteado for 0: PERSISTÊNCIA
    if (moeda1 == 0){
#: À ultima linha do dataframe fossil, acrescente um vetor com o mesmo
número da espécie, o mesmo tamanho e o tempo
#do ciclo seguinte (t+1)
      fossil[nrow(fossil)+1,] <- c((fossil[fossil[,1] == sp & fossil[,3] ==
t,c(1,2)]), t+1)
    }

#: Se o valor sorteado fo 1: CLADOGÊNESE
if (moeda1 ==1){

#: Executa o evento abaixo 2 vezes, pois na cladogenese duas espécies são
```

```
geradas.
#Sorteia 1 valor de 2 valores, com a mesma probabilidade.
      for(i in 0:1){moeda2 <- sample(0:1,size=1,prob=NULL)
#: Se o valor sorteado for 0....
                    if (moeda2 == 0){
#: Some x ao tamanho da espécie em questão. Guarde isso no arquivo
tamanho_novo1.
                      tamanho_novo1<- (fossil[fossil[,1] == sp & fossil[,3]
== t, 2])+x
#: À ultima linha do dataframe fossil, acrescente um vetor com o valor de
espécie nova (ultima espécie+1), o o tamanho_novo1, e o tempo do ciclo
seguinte (t+1).
                      fossil[nrow(fossil)+1,] <- c(max(lista_especie)+1,
tamanho_novo1, t+1)
#: Atualize a lista de espécies, para incluir a espécie nova.
                      lista_especie <- unique(fossil[,1])
                    }
#: Se o valor sorteado for 1...
                    if (moeda2 == 1){
#: Subtraia x do tamanho da espécie em questão. Guarde isso no arquivo
tamanho_novo2.
                      tamanho_novo2 <- (fossil[fossil[,1] == sp & fossil[,3]
== t, 2])-x
#: Se o tamanho_novo2 ficar menor do que o valor de leftwall...
                      if (tamanho_novo2<leftwall){
#: Atribua o valor de leftwall à tamanho_novo2.
                        tamanho_novo2 <- leftwall
                      }
#: À ultima linha do dataframe fossil, acrescente um vetor com o valor de
espécie nova (ultima espécie+1), o o tamanho_novo1, e o tempo do ciclo
seguinte (t+1).
                      fossil[nrow(fossil)+1,] <- c(max(lista_especie)+1,
tamanho_novo2, t+1)
                      lista_especie <- unique(fossil[,1])
                    }
      }
    }
  }
}



#:Início dos gráficos: abre um dispositivo gráfico.
x11()

#: Parâmetros: Três gráficos em uma coluna, com margens especificadas, com
eixo esquerdo e inferior, com tickmarks
#pequenas, tamanho de fonte ajustada, tamanho dos valoes do eixo, sempre
horizontais.
par(mfrow=c(2,2),mar=c(10,5,5,5),bty="L",tcl=-0.2,cex=0.8,cex.axis=0.7,las=1
)
```

```
#: Plota o a média dos tamanhos das espécies de cada um dos tempos. Dá nome
aos eixos, remove o eixo X e define pontos como pontos sólidos.
plot(x=Tempo, y=tapply(X=fossil[,2], INDEX=fossil[,3],mean),ylab="Tamanho",
xlab="Tempo",xaxt="n",pch=16)

#: Redefine o eixo X removido do primeiro gráfico. define que será de 0 até
t+2
axis(side=1, at=seq(0,t+2))
#: Traça a linha que une cada um dos pontos do gráfico.
segments(x0=seq(0,t), y0=tapply(X=fossil[,2], INDEX=fossil[,3],mean)[-
(t+2)], x1=seq(1,t+1), y1=tapply(X=fossil[,2], INDEX=fossil[,3],mean)[-1])

#:Diz para plotar em cima do gráfico anterior
par(new=TRUE)
#Um plot vazio, para colocar a legenda
plot(x=Tempo, y=tapply(X=fossil[,2], INDEX=fossil[,3],mean),ylab="",
xlab="",axes=FALSE, pch=16)

#: Coloca a legenda, como o ponto igual ao do gráfico
legend(x=t+2 ,y=median(tapply(X=fossil[,2], INDEX=fossil[,3],mean)), pch=16,
legend="Média",bty="n",xpd=TRUE)

#: Inicia a plotagem do segundo gráfico:
#Plota o gráfico dos valores mínimos de cada tempo, define as legendas dos
eixos, retira os dois eixos, e define os
#limites de valores de y.
plot(x=Tempo, y=tapply(X=fossil[,2], INDEX=fossil[,3],max),ylab="Tamanho",
xlab="Tempo",xaxt="n", yaxt="n",ylim=c(leftwall,max(fossil[,2]) ),pch=1)

#: Redefine o eixo X, com os valores de tempo.
axis(side=1, at=seq(0,t+2))

#: Redefine o eixo y, e acerca a distribuição dos valores.
axis(side=2, at=seq(0, max(fossil[,2]),by=0.2))

#: Traça a linha que une cada um dos pontos de máximos.
segments(x0=0:t, y0=tapply(X=fossil[,2], INDEX=fossil[,3],max)[-(t+2)],
x1=0:t+1, y1=tapply(X=fossil[,2], INDEX=fossil[,3],max)[-1])

#: Define que o gráfico a seguir será plotado em cima do anterior.
par(new=T)

#: Plota o gráfico dos valores mínimos de cada tempo, retira os eixos,
define os limites do eixo y, tranforma os pontos em
#triangulos,tira as labels dos eixos.
plot(tapply(X=fossil[,2], INDEX=fossil[,3],min), axes=FALSE,
ylim=c(leftwall, max(fossil[,2]) ), pch=17,ylab="", xlab="")

#Plota as duas legendas do mesmo gráfico, na margem do gráfico.
legend(x=t+2 ,y=max(fossil[,2]), pch=1, legend="Máximo",bty="n",xpd=TRUE)
```

```
legend(x=t+2 ,y=(max(fossil[,2])/6)*4, pch=17,
legend="Mínimo",bty="n",xpd=TRUE)

#Liga os pontos da plotagem de mínimos.
segments(x0=0:t+1, y0=tapply(X=fossil[,2], INDEX=fossil[,3],min)[-(t+2)],
x1=0:t+2, y1=tapply(X=fossil[,2], INDEX=fossil[,3],min)[-1])

#Cria objeto Moda vazio, para guardar as modas
Moda <-c()

#Acha moda de cada tempo (for), tabelando tempo e tamanho, perguntando quem
é o maior velor(+frequente),
#Isso é usado para indice de posição para saber qual é o tamanho. sort e
unique alinham a posição da coluna
#tamanho com o que resultou do table.
for(i in seq(1,t+2)){
  Moda[i] <-
sort(unique(fossil[,2]))[which.max(table(fossil[,2],fossil[,3])[,i]) ]
}

#Plota a moda pelo tempo, coloca legenda no eixo y e define a escala do eixo
y
plot(Tempo, Moda, pch=15,xaxt="n",
ylab="Tamanho",ylim=c(leftwall,max(fossil[,2]) ))

#plota o eixo x, que foi suprimido do gráfico
axis(side=1, at=seq(0,t+2))

#Permite plotagem sobreposta
par(new=TRUE)

#Plota o memos gráfico, para permitir colocar a legenda
plot(Tempo,Moda,ylab="", xlab="",axes=FALSE,
pch=15,ylim=c(leftwall,max(fossil[,2]) ))

#Plota a legenda do gráfico Moda
legend(x=t+2 ,y=median(c(leftwall,max(fossil[,2]))), pch=15,
legend="Moda",bty="n",xpd=TRUE)

#Começa a gerar o valor de tamanho das primieras ocorrências das espécies
#Cria objetos vazios para o ciclo
tempo_primeiro <- c()
tamanho_primeiro <- c()
#Para cada espécie, recupera qual o primeiro tempo que surgiu e usa isso
para achar o primiero tamanho.
for(sp in lista_especie){
  primeiro_t = unique(fossil[fossil$especie == sp, 3])[1]
  tempo_primeiro[sp] <-primeiro_t #o -1 é só pro tempo começar em 0
  tamanho_primeiro[sp]<-fossil[fossil$especie == sp & fossil$tempo ==
primeiro_t, 2]
}
```

```
#Arruma o tempo, para começar em 0.
tempo_primeiro_ok <- tempo_primeiro-1

#Plota o gráfico de primeiras ocorrências, com eixo x corrigido.
plot(tempo_primeiro_ok, tamanho_primeiro, bty="L",
pch=16,xaxt="n",xlab="Tempo",ylab="Tamanho")
axis(side=1, at=seq(0,t+2))
return(title(main="Tamanho na primeira ocorrência"))
}

###FIM

fullhouse()
```

```
fullhouse                package:unknown                R Documentation

~~Simulação de macroevolução~~

Description:

     ~~ fullhouse simula variação de algum parâmetro (Tamanho) ao longo do
tempo, sob random walk. Ao passar de um tempo para outro, cada espécie pode
sofrer: (1)Cladogênese,com duas novas espécies, (2) nada (se mantem igual)
ou (3) Extinção. A função retorna 4 gráficos de medidas/tempo (média,
Máximo/Mínimo, Moda, Tamanho na primeira ocorrência).

Usage:

     fullhouse (nlin, isize, x, t, leftwall)

Arguments:

 nlin: Numeric. Número de linhagens/espécies iniciais

 isize: Numeric. Tamanho inicial

 x: Integer. amplitude de variação do parâmentro em cada evento de mudança

 t: Integer. intervalos de tempo

leftwall: Numeric. valor mínimo do parâmetro

Details:

     ~~ If necessary, more details than the description above ~~

Value:

     fullhouse retorna 4 gráficos
     Média: Média dos valores do parâmetro (tamanho) das espécies, para cada
```

```
tempo
     Máximo/Mínimo: O valor máximo e mínimo, para cada tempo
     Moda: o valor mais frequente, para cada tempo
     Tamanho na primeira ocorrência: Plot do tamanho de cada espécie ao
surgir.

Warning:

     Simulações com mais t > 25 costumam ficar muito lentas! 20 é um bom
número, mas não aumente muito o número de linhagens!


Author(s):

   Guilherme Gainett
   e-mail:ggainett@gmail.com

References:

Gould, S. J. 1988. Trends as changes in variance: a new slant on progress
and directionality in evolution. Journal of Paleontology, 62(3):319-29.

Gould, S. J. 1996. Full House: The Spread of Excellence from Plato to
Darwin. New York, NY. Three Rivers Press.

Stanley, S. M. 1973. An explanation for Cope's rule. Evolution, 27:1-26.



Examples:

     #Espécies iniciais próximas à barreira: tendência?
     nlin=5,t=20,leftwall=0.15,isize=0.2
     #Espécies iniciais longe da barreira
     nlin=5,t=20,leftwall=0.15,isize=1
```

Código Full House